Blind Video Deblurring

Sandro Borin Neto, MSc Student, IST

Abstract— Blind image deblurring (referred to as "blind", due to the lack of knowledge, or partial knowledge, about the blurring filter) is a problem of great scientific and technological interest, with diverse applications (such as photography, biomedicine, surveillance, astronomy, etc.). The goal of this work is to extend a recently proposed, efficient method, which achieves state-of-the-art results, to make it applicable to video; for this goal, the three-dimensional structure of the video data is exploited. The method is expected to be able to deal with object motion, without the need for its explicit estimation. The blind image deblurring method implemented in this work was extended to a frame-based blind video deblurring method, and a volume-based blind video deblurring method. Experiments were performed using the blind image deblurring method (with and without noise) for different types of blurring filters. For the video methods, two types of experiments were performed using different types of blurs: progressive and 3D blurring.

Index Terms— Blind Image Deblurring; Blind Video Deblurring; Alternating Direction Method of Multipliers; Frame-based Blind Video Deblurring; Volume-based Blind Video Deblurring.

I. INTRODUCTION

Image deblurring is an ill-posed inverse problem, in which an image is degraded by a linear degradation model in addition to noise. Currently, two types of deblurring methods exist: Blind Image Deblurring (BID) and Non-blind Image deblurring (NBID). For NBID, the blurring operator is known, and for BID the blurring operator is not known. This blurring operator is typically ill-conditioned: small disturbances (such as noise) greatly affect the image estimation [1] [2]. This work focuses on BID, so estimating the blurring operator is also performed in addition to the image estimate, giving BID a greater degree of applicability compared to NBID methods. Although not having unicity in the solution (due to ill-posed problem) [3], a solution can be chosen based on prior information [4]. Most BID methods, in order to obtain reasonable results, restrict the type of deblurring filters used. These restrictions might be strong (for example, parametric models [5]) or weak (using priors for example [6]). A recent solution [7] on which this work is based makes weak assumptions on the blurring operator (such as having limited support) and managed to achieve good results, although being necessary to establish stopping criteria and parameter adjustments [8] [9].

The main difference between video deblurring and image deblurring is the addition of a time component. The presence of this component adds a new layer of information, such as motion, which is inexistent in image deblurring. Motion in a video sequence is a new source of blur that can be handled using several methods, such as block-matching [10], optical-flow [11], or removed frame by frame, such as in [12].

The linear degradation model applied to a single image can be extended to a group of images, and on this group of images it is possible to handle each image as an independent BID problem (frame-based). An alternative to this is to consider that the group of images is a space-time volume (introduced by Jähne [13]) and on this volume it is possible to apply the same linear degradation model stated before, where the temporal component is now explored in contrast to the frame-based solution. This work explores both approaches.

II. BLIND IMAGE DEBLURRING

The method presented in this chapter is an extension of the BID method in [7] [14]. In particular, ADMM is used as an optimization method for estimating both image and filter, allowing restricting the filter in having a fixed support and non-negative entries.

A. Observation Model

The linear degradation of an image is modeled by:

$$ y = Ax + n, \quad (1) $$

where $y \in \mathbb{R}^n$, $x \in \mathbb{R}^m$ and $n \in \mathbb{R}^n$ are lexicographically ordered vectors which contains the pixels of the degraded image, the original image we want to estimate and the additive noise respectively [15]. Matrix $A \in \mathbb{R}^{m \times n}$ represents the blurring degradation: if $n = m$ then the degradation is considered cyclic and $A = H \in \mathbb{R}^{n \times n}$ is a square matrix that represents the cyclic convolution with the blurring filter $h$:

$$ y = Hx + n. \quad (2) $$

Equation (2) may be written in function of filter $h$:

$$ y = Xh + n, \quad (3) $$

where $A = X \in \mathbb{R}^{m \times n}$ is the convolution with the image in vector $x$. Most models that use rapid deconvolution assume periodic boundary conditions, taking advantage of the efficiency of the Fast Fourier Transform (FFT) in this scenario. However, these models display artifacts on their boundaries as
a consequence [16]. To assume unknown boundary conditions is more natural, as it simulates the real behavior of a lens system, where the image is obtained from the center of the lens. Considering this, in order to be able to produce a blurred image with \( \sqrt{n} \times \sqrt{n} \) pixels it is necessary to have \( (\sqrt{n} + 2l) \times (\sqrt{n} + 2l) \) pixels from the original image and a blurring filter with \( (2l + 1) \times (2l + 1) \) pixels. In this case, the blurring operator is

\[
A = MH \in \mathbb{R}^{n \times (\sqrt{n} + 2l)}^2
\]

and is separated in a product between a cyclic convolution performed by matrix \( H \in \mathbb{R}^{(\sqrt{n} + 2l) \times (\sqrt{n} + 2l)}^2 \) and the mask matrix \( M \in \mathbb{R}^{n^2 \times (\sqrt{n} + 2l)}^2 \) which excludes pixels at the borders (where the cyclic convolution is invalid [15]):

\[
y = MHx + n = MXh + n. \tag{4}
\]

### B. Cost Function

Estimation of both filter \( h \) and image \( x \) requires a cost function that must be minimized:

\[
C(x, h) = \frac{1}{2} \| y - MHx \|_2^2 + \lambda \sum_i \| D^{(i)} x \|_2^q + i_S^+(h),
\]

where \( i_S^+(h) \) is the indicator function of the set \( S^+ \):

\[
i_S^+(h) = \begin{cases} 0, & h \in S^+ \\ \infty, & h \notin S^+.
\end{cases}
\]

### C. BID Algorithm in [15] [17]

The purpose of BID algorithm is to minimize the cost function of (5), for both the image \( x \) and filter \( h \) (Algorithm 1 [17]). At the beginning of the algorithm, \( \lambda \) is set to have a high value, diminishing progressively in each cycle.

![Figure 1 – Four directional filters used by the regularization term.](image)

The variable \( q \in [0,1] \) is the sparsity parameter and it is chosen to be equal to 0.5. Variable \( \lambda \) is the regularization parameter and it is set to be greater than zero.

### D. ADMM algorithm in [15] [17]

The minimizations of the cost function (5) that are performed in steps 2 and 3 of Algorithm 1 can also be performed by minimizing the equivalent problem:

\[
\min_{x \in \mathbb{R}^n} \sum_{j=1}^J g^{(j)}(G^{(j)}x), \tag{7}
\]

where \( z \in \mathbb{R}^d \) is the vector to be optimized, \( G^{(j)} \in \mathbb{R}^{p \times d} \) are arbitrary matrices and \( g^{(j)} : \mathbb{R}^p \to \mathbb{R} \). Equation (7) may be rewritten as:

\[
\min_{u \in \mathbb{R}^n} \sum_{j=1}^J g^{(j)}(u^{(j)}), \tag{8}
\]

subject to \( u^{(j)} = G^{(j)}x \) for \( j = 1, \ldots, J \), where \( u^{(j)} \) are the splitting variables. Algorithm 2 has the steps of generic ADMM algorithm used to solve the minimization problem of (8). The proximity operator in step 4 can be defined as:

\[
\text{prox}(y)_{f(x)} = \arg \min_{x \in \mathbb{R}^n} \frac{1}{2} \| y - x \|_2^2 + f(x). \tag{9}
\]
Step 4 thus is given by:
\[ u_{k+1}^{(j)} = \arg\min_u \frac{1}{2}\|u_{k+1}^{(j)} - G^{(j)}z_{k+1} - d_k^{(j)}\|^2 + g^{(j)}/\mu^{(j)} \, . \] (10)

E. Image Estimate

The image estimate update in step (2) of Algorithm 1 can be performed by minimizing the cost function:
\[ C(x, h) = \frac{1}{2}\|y - MHx\|^2_2 + \lambda\sum_i \left[\|D^{(i)}x\|^2_2\right]^q, \] (11)
or, using the notation in (8), as:
\[ \min_{u \in \mathbb{R}^n} \frac{1}{2}\|y - Mu^{(1)}\|^2_2 + \lambda\sum_i \left[\|u^{(2)}\|^2_2\right]^q, \] (12)
subject to \( u^{(1)} = Hz, \ u^{(2)} = Dz, \)
where the splitting variables are:
\[ u^{(1)} = G^{(1)}z = Hx, \] (13)
\[ u^{(2)} = G^{(2)}z = Dx, \] (14)
and their respective \( g^{(j)} \) functions are:
\[ g^{(1)}(u^{(1)}) = \frac{1}{2}\|y - Mu^{(1)}\|^2, \] (15)
\[ g^{(2)}(u^{(2)}) = \lambda\sum_i \left[\|u^{(2)}\|^2_2\right]^q, \] (16)
where \( u^{(i)} \in \mathbb{R}^4 \) is the vector with the output of four directional filters at pixel \( i \) and \( D \in \mathbb{R}^{m \times m} \) is a matrix that contains the stacking by lines of \( D^{(i)} \). We can write steps 2 and 3 of Algorithm 2 as [18] [19] [20] [21]:
\[ z_{k+1} = \psi_{k+1} = (\mu^{(1)}H^TH + \mu^{(2)}D^TD)^{-1}\psi_{k+1}, \] (17)
\[ \psi_{k+1} = [\mu^{(1)}H^TU_k^{(1)} + d_k^{(1)}] + \mu^{(2)}DU_k^{(2)} + d_k^{(2)}]. \] (18)
Both \( H^T \) and \( D^T \) are easily computed using the conjugate of the Fast Fourier Transform (FFT) as they are diagonal in this domain. \( H^TH \) and \( D^TD \) are also computed in the Fourier domain. The inverse operation in (17) is also performed in the Fourier domain.

Step 4 of Algorithm 2 is related with updating the splitting variables in (13) and (14). The first variable is updated as:
\[ u_{k+1}^{(1)} = (M^TM + \mu^{(1)})^{-1}[M^Ty + \mu^{(1)}(Hx_{k+1} - d_k^{(1)})], \] (19)
where \( M^TM \) is a binary matrix that sets to zero all unobserved boundary pixels, \( M^Ty \) creates an extended version of the degraded image \( y \), with its boundary pixels equal to zero. Both these operations are easily computed in the space domain. The inverse operation in (19) is performed in the space domain.

For the second splitting variable, the proximity operator in (10) has a closed form solution [22] for \( q = 0.5 \) which can be computed based on the roots of a 3rd order polynomial, choosing the solution which minimizes (20):
\[ \text{shrink}(S^{(2)}_i, \lambda, q) = \text{prox}_{\lambda|t|^q}(S^{(2)}_i) = \min_i \frac{1}{2}\|S^{(2)}_i - t_i\|^2 + \lambda|t|^q, \] (20)
\[ c^3 - abs(S^{(2)}_i)\lambda + q = 0 \] (21)
where \( S^{(2)}_i \in \mathbb{R}^4 \) a vector obtained from the output of \( Dx - d^{(2)} \) at pixel \( i \).

Finally, step 5 updates variables \( d_k^{(j)} \):
\[ d_{k+1}^{(1)} = d_k^{(1)} - (Hx_{k+1} - u_{k+1}^{(1)}), \] (22)
\[ d_{k+1}^{(2)} = d_k^{(2)} - (Dx_{k+1} - u_{k+1}^{(2)}), \] (23)
where \( Hx_{k+1} \) and \( Dx_{k+1} \) are performed in the Fourier domain.

F. Filter Estimate

Updating the filter estimate follows a similar procedure as updating the image estimate. Step 3 in Algorithm 1 is performed by minimizing the cost function:
\[ C(x, h) = \frac{1}{2}\|y - MXh\|^2_2 + \lambda_s + (h), \] (24)
or, using the notation in (8), as:
\[ \min_{u \in \mathbb{R}^n} \frac{1}{2}\|y - Mu^{(1)}\|^2_2 + \lambda_s + (u^{(2)}), \] (25)
subject to \( u^{(1)} = Xz, \ u^{(2)} = Ix \).

The splitting variables are defined, for \( f = 2 \), as:
\[ u^{(1)} = G^{(1)}z = Xh, \] (26)
\[ u^{(2)} = G^{(2)}z = h, \] (27)
and their respective functions are:
\[ g^{(1)}(u^{(1)}) = \frac{1}{2}\|y - Mu^{(1)}\|^2_2 \] (28)
\[ g^{(2)}(u^{(2)}) = \lambda_s + (u^{(2)}). \] (29)

Step 2 in Algorithm 2 updates the filter estimate and has a closed form solution [15]:
\[ z_{k+1} = h_{k+1} = (\mu^{(1)}X^TX + \mu^{(2)}I)^{-1}\psi_{k+1}, \] (30)
\[ \psi_{k+1} = [\mu^{(1)}X^Tu_k^{(1)} + d_k^{(1)}] + \mu^{(2)}I(u_k^{(2)} + d_k^{(2)}), \] (31)
where \( X^T \) is computed using the conjugate of FFT. \( X^TX \) and the inverse operation in (29) are both performed in the Fourier domain and \( I \in \mathbb{R}^{m \times m} \) is an identity matrix.

Step 4 is related with updating the splitting variables. The first variable is updated as:
\[ u_{k+1}^{(1)} = (M^T M + \mu^{(1)})^{-1} \{ M^T y + \mu^{(1)} (Xh_{k+1} - d_k^{(1)}) \} \]
and the second splitting variable as:
\[ u_{k+1}^{(2)} = P_{S^+}(h_{k+1} - d_k^{(2)}) \]
where \( P_{S^+}(w) \) is the projection of pixels \( w \) in the domain of \( S^+ \).

Finally, step 5 updates variables \( d_k^{(j)} \):
\[ d_k^{(1)} = d_k^{(1)} - (Xh_{k+1} - u_{k+1}^{(1)}) \]
\[ d_k^{(2)} = d_k^{(2)} - (h - u_{k+1}^{(2)}) \]
where \( Xh_{k+1} \) is performed in the Fourier domain.

### III. Blind Video Deblurring

#### A. Frame-based Blind Video Deblurring
This method of deblurring is a direct extension of BID presented in II, applied to each frame of a video. The estimation of each frame is considered an independent inverse problem, where the temporal component of the video is not explored.

#### B. Volume-based Blind Video Deblurring
The observation model introduced in II is applicable in this type of video deblurring, with some key differences. The degradation model is applied, not to an image, but to a video with volume \( V \). Considering this, the linear degradation is modeled by:
\[ y = Ax + n, \]
where \( y \in \mathbb{R}^n \), \( x \in \mathbb{R}^m \) and \( n \in \mathbb{R}^n \) are lexicographically ordered vectors which contains the pixels of the degraded video, the original video we want to estimate and the additive noise respectively. Matrix \( A \in \mathbb{R}^{m \times n} \) represents the blurring degradation: if \( n = m \) then the degradation is considered cyclic and \( A = H \in \mathbb{R}^{n \times n} \) is a square matrix that represents the cyclic convolution with the blurring filter \( h \):
\[ y = Hx + n. \]
Equation (37) may be written in function of filter \( h \):
\[ y = Xh + n. \]
where \( X = x \in \mathbb{R}^{n \times m} \) is the convolution with the video in vector \( x \). As before, we assume unknown boundary conditions as they represent a more realistic approach. In this case, the 3D blurring operator is \( A = MH \in \mathbb{R}^{n \times (\sqrt{7}+2)^2} \) and is separated in a product between a cyclic convolution performed by matrix \( H \in \mathbb{R}^{(\sqrt{7}+2)^2 \times (\sqrt{7}+2)^2} \) and the 3D mask matrix \( M \in \mathbb{R}^{n^2 \times (\sqrt{7}+2)^2} \) which excludes pixels at the borders of the volume (where the cyclic convolution is invalid):
\[ y = MHx + n = MXh + n. \]

The cost function for this type of video deblurring is similar to (5), but the variables represent 3D signals and the regularization term \( D^{(i)} \in \mathbb{R}^{4n^2} \) computes the output of 14 directional 3D regularization filters at pixel \( i \). The base filter favors sparse planes (instead of lines), and it is displayed in figure 2. All others are built from spatial rotations of this filter.

![Figure 2 - Base regularization filter used in 3D blind video deblurring: a) frame 1 and b) frame 2.](image)

The application of the Algorithm 1 and Algorithm 2 is identical as presented in II, considering that all variables are space-time volumes where the temporal component is explored, unlike the frame-based method. Another difference between this method and the frame-based method is that the regularization parameter \( \lambda \) is the same for all video frames, where in the frame-based method each frame has its own \( \lambda \).

![Figure 3 - BID-ADMM graphical representation.](image)

### IV. Experimental Results

#### A. Blind Image Deblurring
The BID method in [15] [17] was implemented and experiments (with and without noise) were performed with “cameraman” image (256x256 resolution), with four types of delubring filters of size 9x9 (15x15 support): square, circular, linear motion and non-linear motion (see fig. 4). For the parameters in all experiments \( q = 0.5, \lambda_0 = 0.5, \; r = 2, \; \mu^{(1)} = 0.01 \) and \( \mu^{(2)} = 0.1 \) were used.
The estimation process for the image and the filter is represented in figure 3, and all the process is controlled by a stopping criteria. The chosen stopping criteria was a fixed number of iterations equal to 40. The image estimate is performed by the “find_y” block and the filter estimate is performed by the “find_h” block, both limited also by a fixed number of iterations: 20 and 15 respectively. The chosen metric for evaluating image quality was the improved signal-to-noise-ratio (ISNR). For measuring the amount of noise level, blurred signal-to-noise-ratio (BSNR) was used.

![Image Filters](image1.png)  
**Figure 4 – Deblurring filters for blind image deblurring:** a) square, b) circular, c) linear motion and d) non-linear motion.

![Image Estimates](image2.png)  
**Figure 5 – Image estimates for blind image deblurring, using a square filter:** a) original image, b) blurred image, c) image estimate without noise and d) image estimate with noise.

![Image Filters](image3.png)  
**Figure 6 – Filter estimates for blind image deblurring, using a square filter:** a) original filter, b) filter estimate without noise and c) filter estimate with noise.

Table 1 shows the results of all experiments performed in BID. After analyzing the results, we can safely conclude that the BID method was successfully implemented and the results are comparable with the ones in [15] [16]. Both image estimates (for example, figure 5) and filter estimates (for example, figure 6) have good quality and even the presence of noise did not change the results significantly, although in the filter estimate in figure 6 we notice some differences between b) and c). The presence of noise had a small influence on the elapsed time as well.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Square</td>
<td>∞</td>
<td>13.34</td>
<td>111</td>
</tr>
<tr>
<td>Circular</td>
<td>∞</td>
<td>9.38</td>
<td>103</td>
</tr>
<tr>
<td>Linear</td>
<td>∞</td>
<td>9.05</td>
<td>77</td>
</tr>
<tr>
<td>Non-linear</td>
<td>∞</td>
<td>11.14</td>
<td>78</td>
</tr>
</tbody>
</table>

Table 1 – Experimental results for blind image deblurring.

B. Blind Video Deblurring

Both methods of blind video deblurring (frame-based and volume-based) were implemented and for each method, two experiments were performed using different types of blur: progressive blur and 3D blur. A sample of 20 frames from the video sequence “foreman” with 288x288 resolution was used. For the parameters in all experiments, \( q = 0.5, \lambda_0 = 0.5, \ r = 2, \mu^{(1)} = 0.01 \) and \( \mu^{(2)} = 0.1 \) were used and the value of ISNR was used to evaluate the video quality.

Progressive blur consists in blurring groups of frames with the same blurring filter, to simulate an out of focus lens. For this experiment, the original video with 20 frames was divided into 4 groups of 5 frames. Each group of frames was blurred with a circular filter of different diameter: 3, 5, 9 and 13 respectively. 3D blur consists in applying a tridimensional blurring filter on the space-time volume (video). Part of the blurring effect present in an image has a temporal component in addition to a spatial component, where the information from adjacent frames are used in the blurring process. For this experiment, a square filter with spatial component of 9x9 and a temporal component of 3 (9x9x3) was used.

For the frame-based method, figure 3 illustrates the estimation process of each frame, as for the volume-based method, figure 3 illustrated the estimation process of a space-time volume (video). This has significant importance because it differentiates both methods regarding the regularization parameter: for the frame-based method, each frame has its own value of \( \lambda \), on the other hand, for the volume-based method all frames have the same value of \( \lambda \).

Table 1 shows the results of all experiments performed in BID. After analyzing the results, we can safely conclude that the BID method was successfully implemented and the results are comparable with the ones in [15] [16]. Both image estimates (for example, figure 5) and filter estimates (for example, figure 6) have good quality and even the presence of noise did not change the results significantly, although in the filter estimate in figure 6 we notice some differences between b) and c). The presence of noise had a small influence on the elapsed time as well.

The stopping criteria used for both methods is the same as introduced in IV.A.

B.1) Experimental Results for Progressive Blur

Figures 7 to 10 show the result of the image estimates in this experiment. By analysis, we concluded that the volume-based method had inferior results in comparison with the frame-based method. There is a clear reduction in quality for the image estimates (as the blur intensifies). The frame-based method maintains a good level of quality throughout the video sequence. For the filter estimates, the volume-based method was unable to perform a decent estimation, as shown in figure 12. The frame-based method was able to estimate the progressive blur correctly and with good quality, as shown in
Table 2 summarizes the results for both methods in this experiment.

<table>
<thead>
<tr>
<th>BVD Method</th>
<th>ISNR [dB]</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame-based</td>
<td>2.5</td>
<td>1 h 30 min</td>
</tr>
<tr>
<td>Volume-based</td>
<td>-5</td>
<td>3 h 21 min</td>
</tr>
</tbody>
</table>

Table 2 – Experimental results for blind video deblurring, using progressive blur.

B.2) Experimental Results for 3D Blur

Figures 13 to 16 show the results of the image estimate in this experiment. Both methods of blind video deblurring presented difficulties in estimating the images and the 3D blur. For the volume-based method, the image estimates were better for the frames in the middle of the video, and worse for the frames near the beginning and near the end of the video. This shows that this method was unable to maintain the quality of the estimation throughout the video sequence. The frame-based method had worse results when motion was present. For the filter estimates, the volume-based method identified that there was a temporal component present in the blur (the 3 squares in figure 18) but the spatial component had poor quality. The frame-based method only estimated the spatial component of the blur present in the frames but this estimation was worse during motion. It was expected that the volume-based method would perform better than the frame-based method in this context (especially during motion), but from Table 3 we can conclude that this is not the case. The ISNR value for both methods are practically the same.

<table>
<thead>
<tr>
<th>BVD Method</th>
<th>ISNR [dB]</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame-based</td>
<td>0.3</td>
<td>1 h 31 min</td>
</tr>
<tr>
<td>Volume-based</td>
<td>0.4</td>
<td>3 h 19 min</td>
</tr>
</tbody>
</table>

Table 3 – Experimental results for blind video deblurring, using 3D blur.

Figure 7 – Image estimates for frame 3, using progressive blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.
Figure 8 – Image estimates for frame 8, using progressive blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 9 – Image estimates for frame 13, using progressive blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 10 – Image estimates for frame 17, using progressive blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 11 – Filter estimates using the frame-based method, with progressive blur: a) frame 3, b) frame 8, c) frame 13 and d) frame 17.
Figure 12 – Filter estimate using the volume-based method, with progressive blur.

Figure 13 – Image estimates for frame 3, using 3D blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 14 – Image estimates for frame 8, using 3D blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 15 – Image estimates for frame 13, using 3D blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.
Figure 16 – Image estimates for frame 17, using 3D blur: a) original frame, b) blurred frame, c) image estimate using frame-based method and d) image estimate using volume-based method.

Figure 17 – Filter estimates using the frame-based method, with 3D blur: a) frame 3, b) frame 8, c) frame 13 and d) frame 17.

Figure 18 – Filter estimate using the volume-based method, with 3D blur.
V. CONCLUSION AND FUTURE WORK

The BID method in [15] [17] was successfully implemented, as shown by the results. Image and filter estimations were performed with good quality even with the presence of noise.

For blind video deblurring, the frame-based method showed better results in comparison with the volume-based method. However, it had difficulties handling 3D blur, as expected. For the volume-based method, better results were expected, especially for the 3D blur experiment. The poor quality of the estimation for this experiment might be related with the amount of temporal component used by the blurring filter, as being too strong. Since 3D blurring uses adjacent frame information to perform part of the blur, the video sample used might have been too small. For a video with 20 frames, 15% of the video sample is used to perform the temporal blur in one frame, which is considerable. One solution might be increasing the video sample, thus reducing the total amount of information used to perform the temporal blur in one frame.

For future work, an extension of the BID method to include colored pictures would be the next step. This would permit extending the frame-based method to include colored video sequences as well. The volume-based method needs more work, not only regarding the estimation quality, but also in the total elapsed time needed to run the method, which is considerably higher than the frame-based method.

VI. ACKNOWLEDGEMENT

The author of this paper would like to thank Diogo Guerreiro for his collaboration in implementing the BID method presented in II.

VII. REFERENCES
