HIGH-PERFORMANCE 3D DECONVOLUTION OF FLUORESCENCE MICROGRAPHS

Sander Kromwijk, Stamatios Lefkimmiatis, and Michael Unser

Biomedical Imaging Group, EPFL, CH-1015 Lausanne, Switzerland

Email:[sander.kromwijk,stamatis.lefkimmiatis,michael.unser]@epfl.ch

ABSTRACT

In this work, we describe our approach of combining the most effective ideas and tools developed during the past years to build a variational 3D deconvolution system that can be successfully employed in fluorescence microscopy. In particular, the main components of our deconvolution system involve proper handling of image boundaries, choice of a regularizer that is best suited to biological images, and use of an optimization algorithm that can be efficiently implemented on graphics processing units (GPUs) and fully benefit from their massive parallel computational capabilities. We show that our system leads to very competitive results and reduces the computational time by at least one order of magnitude compared to a CPU implementation. This makes the use of advanced deconvolution techniques feasible in practice and attractive computationally.

Index Terms— Graphics Processing Unit, image regularization, variational reconstruction, convex optimization.

1. INTRODUCTION

Fluorescence microscopy is an imaging technique that has been proven to be a valuable tool for biologists. It has the capacity to provide three-dimensional (3D) structural information of biological specimens, which makes feasible the observation and study of their cellular and subcellular components. The 3D image acquisition is achieved by means of optical sectioning, in which a series of 2D images is recorded at different focal planes. In this setting, one of the greatest limitations is the presence in the measurements of out-of-focus information from adjacent planes [1]. This optical blur reduces the spatial resolution and, thus, the ability to resolve fine structures, and is an inherent limitation encountered in every diffraction-limited optical system. Another factor distorting the measurements is stochastic measurement noise. Both of these degradation factors can make the measured data very difficult to analyze.

Image deconvolution is a computational method that can be employed to mitigate the distortions introduced by the optical system and lead to high-resolution images. The first deconvolution method to be used in the context of 3D fluorescence microscopy was developed by Agard and Sedat in 1983 [2]. Since then, a plethora of deconvolution methods have been proposed. Most of them can be interpreted as variational techniques, where image restoration is cast as an optimization problem of an objective function consisting of two terms: (**a**) the *data fidelity* that measures the proximity of the reconstruction to the measurements and (**b**) the regularizer that encodes prior information about properties of the object to be restored.

Nowadays, with the emergence of sparsity and compressed sensing as new regularization paradigms, non-quadratic regularization methods that systematically produce state-of-the-art results have been developed. However, their applicability to 3D image deconvolution has been widely hindered so far. This is mainly because of the high computational complexity of these methods when faced with large volumes of image data, which translates to several minutes or even hours of processing time before obtaining the final deconvolved image. Fortunately, thanks to the development of multicore and General Purpose Graphics Processing Units (GP-GPU) architectures, which are now widely available and allow for massive parallel computations, this is an obstacle that we can overcome.

In this paper, we exploit the computational power offered by GPUs and we describe a variational 3D deconvolution system that we built to be used in fluorescence light microscopy. To do so, we combine the most effective ideas and tools that have been introduced during the past years. First, we adopt a more accurate observation model than the most common one which assumes periodic image boundary conditions. The adopted model allows us to avoid any assumptions about the boundaries and obtain reconstructions without the presence of border artifacts. Then, having in mind that biological images typically display piecewise-smooth intensity variations, we employ a family of non-quadratic second-order regularization functionals that involves the Hessian operator [3]. These regularizers are suitable for our problem since they promote piecewise-smooth image reconstructions and lead to results of improved quality. They are also well suited for preserving ridges & filament-like structures. Finally, to solve the corresponding minimization problem, we employ an optimization algorithm that takes full advantage of the GPU hardware and reduces significantly the execution time.

The rest of this paper is organized as follows: In Section 2, we discuss the image formation model and briefly describe the regularization functionals that we employ. In Section 3, we present the optimization strategy of choice for minimizing the corresponding objective function. Then, in Section 4 we describe our approach for efficiently implementing the deconvolution algorithm on the GPU and highlight the challenges we have faced. Finally, in Section 5 we provide 3D deconvolution results where we also report the speed-up that we achieve in terms of execution time, compared to a CPU implementation.

2. VARIATIONAL IMAGE RECONSTRUCTION

2.1. Image Formation Model

For any deconvolution method to produce high-quality results, it is necessary that it takes into account the physical phenomena that provide the basis of the image acquisition. In this work, we focus on image acquisition using a widefield microscope, which can be modeled in intensity as a linear space-invariant system [1]. In this case, the image formation can be mathematically formulated as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{M \times N}$ is a linear operator that corresponds to the point-spread function (PSF) of the microscope, $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{y} \in \mathbb{R}^M$ are the vectorized versions of the 3D image to be restored and the observation data, respectively, while $\mathbf{n}$ is a term that accounts for all pos-

sible experimental errors, including model mismatch and stochastic measurement noise. Hereafter, we will consider $\mathbf{n}$ to be i.i.d Gaussian noise with variance σ_n^2.

2.2. Handling Image Boundaries

In the deconvolution literature, the most common approach is to consider that the convolution operator $\mathbf{A}$ is circulant. This is due to the computational simplicity that it offers, since it allows the use of the fast Fourier transform (FFT). However, the price to pay for this simplification is that the restored image is periodic which often results in spurious artifacts at the boundaries. To avoid this problem, we consider that $\mathbf{A}$ in (1) is a composition of two operators, *i.e.*, $\mathbf{A} = \mathbf{MH}$, where $\mathbf{H} \in \mathbb{R}^{N\times N}$ is a circulant convolution operator, while $\mathbf{M} \in \mathbb{R}^{M\times N}$ is a masking operator that keeps only the valid part of the convolution and truncates the circular wraparound at the boundaries. This approach, which is similar to [4–6], ensures that the formation model is more accurate, since no specific assumptions about the boundaries of the 3D image are made. Instead, we hold the deconvolution algorithm responsible for reconstructing the image in a way that best explains the data.

2.3. Regularization Strategy

Similarly to many inverse imaging problems, image deconvolution is an ill-posed problem [7]. This means that the equations relating the object of interest to the measurements are not enough by themselves to uniquely characterize the solution. In order for the recovered solution to be physically or statistically meaningful, one needs to take into account prior information about known properties of the object. The use of a regularization term serves exactly this purpose in the variational framework that we follow in this work.

Having in mind that biomedical images are typically smooth, we employ the family of non-quadratic regularizers that was recently proposed in [3]. These regularizers penalize the eigenvalues of the Hessian matrix computed at every voxel of the 3D image and, thus, favor piecewise-smooth reconstructions. Moreover, they are convex and invariant w.r.t to transformations of the coordinate system, which qualifies them as ideal candidates for imaging applications.

The Hessian Schatten-norm regularizers are defined as [3]

$$\mathcal{HS}_p\left[\mathbf{x}\right] = \left\|\boldsymbol{\mathcal{H}}\mathbf{x}\right\|_{1,p} = \sum_{n=1}^{N} \left\|\left[\boldsymbol{\mathcal{H}}\mathbf{x}\right]_n\right\|_{\mathcal{S}_p}, \forall p \geq 1, \qquad (2)$$

where

$$\left[\boldsymbol{\mathcal{H}}\mathbf{x}\right]_n = \begin{bmatrix} \left[\Delta_{r_1r_1}\mathbf{x}\right]_n & \left[\Delta_{r_2r_1}\mathbf{x}\right]_n & \left[\Delta_{r_3r_1}\mathbf{x}\right]_n \\ \left[\Delta_{r_1r_2}\mathbf{x}\right]_n & \left[\Delta_{r_2r_2}\mathbf{x}\right]_n & \left[\Delta_{r_3r_2}\mathbf{x}\right]_n \\ \left[\Delta_{r_1r_3}\mathbf{x}\right]_n & \left[\Delta_{r_2r_3}\mathbf{x}\right]_n & \left[\Delta_{r_3r_3}\mathbf{x}\right]_n \end{bmatrix} \qquad (3)$$

is the discrete Hessian and $\left[\Delta_{r_ir_j}\mathbf{x}\right]_n$ denotes the discrete approximations of the second-order partial derivatives along two dimensions of the image (volume) at voxel n. In (2), $\|\mathbf{X}\|_{\mathcal{S}_p}$ denotes the Schatten norm of order p of a matrix $\mathbf{X} \in \mathbb{R}^{N_1\times N_2}$. This is defined as the ℓ_p norm of the singular values of $\mathbf{X}$. Note that since the Hessian in (3) is symmetric, its singular values are equal to the absolute eigenvalues. Definition (2) also highlights the relation of these regularizers with the sparsity-promoting group norms commonly met in compressive sensing (see [8], for instance). However, a noteworthy difference is that in (2) the mixed norm is a vector-matrix norm rather than a vector-vector norm. Therefore, the sparsity is enforced on the eigenvalues of the Hessian rather than directly on its elements.

Algorithm 1 : Proximal map evaluation of Hessian Schatten-norm regularizers.

Input: $\mathbf{z}$, $\tau > 0$, $p \geq 1$, $\mathcal{P}_{\mathcal{C}}$.
Initialization: $\boldsymbol{\Psi}_1 = \boldsymbol{\Omega}_0 = \mathbf{0} \in \mathbb{R}^{N\times 3\times 3}$, $t_1 = 1$.
Output: $\hat{\mathbf{x}}$.
while *stopping criterion is not satisfied* **do**
  $\boldsymbol{\Omega}_n \leftarrow \mathcal{P}_{\mathcal{B}_{\mathcal{S}_q}}\left(\boldsymbol{\Psi}_n + \frac{1}{144\tau}\boldsymbol{\mathcal{H}}\mathcal{P}_{\mathcal{C}}\left(\mathbf{z} - \tau\boldsymbol{\mathcal{H}}^*\boldsymbol{\Psi}_n\right)\right)$;
  $t_{n+1} \leftarrow \frac{1+\sqrt{1+4t_n^2}}{2}$;
  $\boldsymbol{\Psi}_{n+1} \leftarrow \boldsymbol{\Omega}_n + \left(\frac{t_n-1}{t_{n+1}}\right)\left(\boldsymbol{\Omega}_n - \boldsymbol{\Omega}_{n-1}\right)$;
  $n \leftarrow n+1$;
end
return $\mathcal{P}_{\mathcal{C}}\left(\mathbf{z} - \tau\boldsymbol{\mathcal{H}}^*\boldsymbol{\Omega}_{n-1}\right)$;

3. OBJECTIVE FUNCTION MINIMIZATION

Having defined the observation model and chosen the regularizer that we will use, we can now obtain the solution to our 3D deconvolution problem as the minimizer of the following objective:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}\in\mathbb{R}^N}{\arg\min}\ \frac{1}{2}\left\|\mathbf{y} - \mathbf{MHx}\right\|_2^2 + \tau\left\|\boldsymbol{\mathcal{H}}\mathbf{x}\right\|_{1,p} + \iota_{\mathcal{C}}\left(\mathbf{x}\right), \qquad (4)$$

where ι_C is the indicator function of a convex set $\mathcal{C}$, defining additional constraints, while $\tau \geq 0$ is a regularization parameter that balances the contributions of the data fidelity and the regularizer in the final result. To solve (4) we choose to employ the FISTA algorithm [9, 10], which exhibits state-of-the-art convergence rates while it does not require any parameter tuning. In this framework, the solution is obtained via the successive minimization of a sequence of surrogate functions that upper-bound the original objective. The FISTA algorithm has two main components : (**a**) the computation of the forward operator $\mathbf{MH}$ and its adjoint and (**b**) the evaluation of the proximal map of the employed regularizers, $\varphi\left(\mathbf{x}\right) = \tau\left\|\boldsymbol{\mathcal{H}}\mathbf{x}\right\|_{1,p} + \iota_{\mathcal{C}}\left(\mathbf{x}\right)$, defined as

$$\mathrm{prox}_{\varphi}\left(\mathbf{z}\right) = \underset{\mathbf{x}\in\mathbb{R}^N}{\arg\min}\ \frac{1}{2}\left\|\mathbf{x} - \mathbf{z}\right\|_2^2 + \varphi\left(\mathbf{x}\right). \qquad (5)$$

In Algorithm 1 we describe the necessary steps for obtaining a numerical solution of the proximal map in (5) according to the method proposed in [3], while in the upcoming sections we will discuss its efficient implementation on GPU. Note that in the algorithmic description provided above, $\boldsymbol{\mathcal{H}}^*$ stands for the adjoint Hessian, $\mathcal{P}_{\mathcal{C}}$ is the projection on $\mathcal{C}$, and $\mathcal{P}_{\mathcal{B}_{\mathcal{S}_q}}$ is the projection on the set $\mathcal{B}_{\mathcal{S}_q} = \{\mathbf{X} \in \mathbb{R}^{N\times 3\times 3} : \|\mathbf{X}_n\|_{\mathcal{S}_q} \leq 1\ \forall n = 1, \ldots, N\}$.

4. GPU IMPLEMENTATION WITH CUDA

Our GPU implementation is based on the CUDA programming environment. CUDA is a parallel computing model created by NVIDIA. It extends a subset of the C++ programming language, enabling code to be run on the GPU while delegating work such as task scheduling and synchronization to the driver. It also provides a programming environment with debugging and profiling support, vital for more complex applications. CUDA is very popular for scientific computing applications, thanks both to its performance and to the collection of standardized libraries provided by NVIDIA.

As mentioned above, there are two main components in the FISTA algorithm. Regarding the forward operator and its adjoint ($\mathbf{A}$ and $\mathbf{A}^*$, respectively), their special structure allows an efficient

GPU implementation that can be accomplished in a straightforward way. It suffices to note that the convolution operator $\mathbf{A}$ is expressed in our observation model as the composition of a masking operator $\mathbf{M}$ and a circulant convolution operator $\mathbf{H}$. Based on this, we can efficiently compute $\mathbf{H}$ in the Fourier domain using the FFT transform of the `cuFFT` library [11] provided by NVIDIA. Then, the masking operation is implemented as a point-wise multiplication of the result with a 3D matrix that consists of zeros and ones. The adjoint operator is implemented using the same type of operations, but they take place in a reverse order.

The next and most critical component of the minimization approach is the computation of the proximal map of the Hessian-based regularizers. From Algorithm 1, we see that a numerical solution of (5) can be obtained by following a series of steps that involve the projection onto the set $\mathcal{B}_{\mathcal{S}_q}$, the computation of the Hessian and its adjoint, point-wise additions and multiplications, and finally the projection of the result onto the convex set $\mathcal{C}$. In our experiments we consider $\mathcal{C}$ to be the positive orthant so as to avoid image reconstructions with negative intensities. In this case, the corresponding projection is a point-wise operation and amounts to setting the negative voxel values to zero. According to the definition of the set $\mathcal{B}_{\mathcal{S}_q}$ provided in Section 3, it is clear that the projection $\mathcal{P}_{\mathcal{B}_{\mathcal{S}_q}}$ can be performed independently for each one of the N matrix components of a multidimensional array of size $\mathbb{R}^{N\times 3\times 3}$. Finally the Hessian operator and its adjoint can also be computed independently for each voxel of the 3D image. In short, all the involved operations for the evaluation of the proximal map are separable; the values depend only on results of previous operations and each voxel can be treated independently. This means that the overall algorithm is an ideal candidate for a fast and efficient implementation on the GPU, since it can fully benefit from the massive parallel computational capabilities of modern GPU hardware. Next, we discuss in more detail the parts of the Algorithm 1 that required special care in our implementation.

4.1. Projection onto Schatten-norm balls

According to [3], the projection of a matrix onto an $\mathcal{S}_q$-norm balls is performed by projecting the singular values of the matrix onto the corresponding ℓ_q vector-norm ball and then reconstructing the projected matrix using the original singular vectors. In our case, the symmetric nature of the Hessian matrix allows us to replace the singular value decomposition operation by an eigenvalue decomposition. Then, one of the main costs of the projection is the computation of the eigenvalue and the corresponding eigenvectors. While there are existing GPU libraries available for this task, these are mostly efficient for handling very large matrices. In our case, the problem we need to deal with is different, since we have to handle a large number (equal to the number of voxels) of 3×3 symmetric matrices. We have implemented three alternative algorithms to solve this problem efficiently, namely a direct method [12], the Householder algorithm, and the Jacobi algorithm. As we can see in Fig. 1, the direct method performs best on CPU but loses its advantage on GPU due to the high amount of branching. On the other hand, the Jacobi method shows exactly the opposite behaviour and is the best performing on GPU. For this reason, we have incorporated the Jacobi method in our final software. Further, we note that in this work we consider only three members of the Hessian-based regularization family that are the most interesting and best performing ones. In particular, we have implemented the Hessian regularizers that involve the Nuclear ($\mathcal{S}_1$), Frobenius ($\mathcal{S}_2$), and Spectral ($\mathcal{S}_\infty$) matrix norms. In these cases, the corresponding projections of the absolute eigenvalues (singular values) of the matrices are given in closed-form and, thus, we have efficiently implemented them on GPU without any problems.

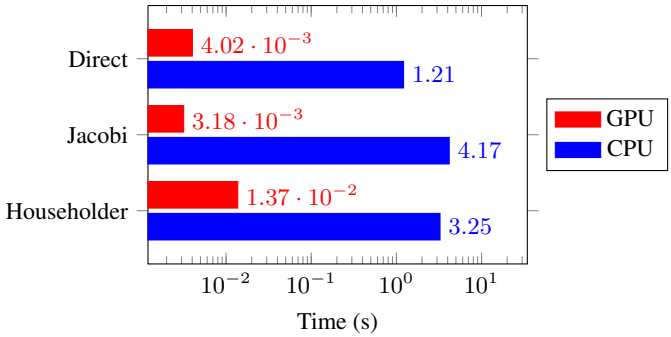


Fig. 1: Execution times for different methods of eigenvalue decomposition in a logarithmic scale. The input data consists of a $100 \times 100 \times 100$ cube of 3×3 symmetric matrices. We then run the test for each algorithm in turn. Each algorithm is executed multiple times to obtain an accurate and consistent measurement.

4.2. Hessian operator and its Adjoint

The computation of the Hessian and its adjoint is also one of the main bottlenecks of our GPU implementation. As the discrete Hessian corresponds to a second-order finite difference operator, to compute it for a single voxel we need to consider 10 voxels from the input. For the adjoint operator, this is exacerbated as its computation requires 60 distinct voxel values. Reads and writes from global memory are the most expensive operation on the GPU. To address this issue our approach is to exploit the data locality. Specifically, the Hessian of neighboring voxels will share most of the input values necessary for its computation. Therefore, we use shared memory (fast on-chip memory) as a custom cache on which different threads collaboratively preload the data necessary for the computation. Compared to the naive implementation, this reduces the amount of read operations by more than 80%[1]. This improvement is most notable for the adjoint operator which, as noted before, needs 6 times more read operations than the Hessian operator. Further, we have exploited the symmetry of the Hessian matrices to save memory usage. In particular, instead of storing all the elements of each matrix we only store the distinct elements, i.e., the upper triangle.

4.3. Other implementation issues

The FISTA algorithm requires an accurate solution of the proximal to lead to a satisfying reconstruction. This means that we need to run the proximal algorithm for enough iterations in order for this condition to be fulfilled. To circumvent this issue, we follow a warm-up strategy where in every FISTA iteration we provide as initial solution to the proximal algorithm the solution obtained in the previous iteration. This leads to an increase of the performance of the overall algorithm at no extra computational cost.

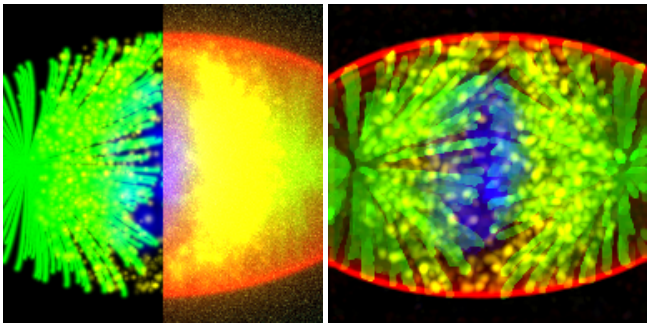
5. EXPERIMENTS AND RESULTS

To demonstrate the overall performance of our 3D deconvolution software, we report reconstruction results on four 3D image stacks

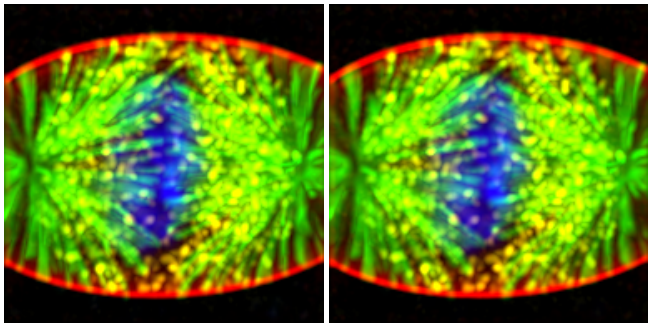
[1]With a collaborative loading block size of 8x8x8 — a good balance between resource usage and speed — there is an effective reduction of 80.47%.

Table 1: ISNR comparisons for the image deconvolution task. Bold values indicate the best reconstruction for the given channel/BSNR pair.

Regularizer		TV				Hessian (Nuclear)				Hessian (Frobenius)			
Channel		1	2	3	4	1	2	3	4	1	2	3	4
BSNR	5	2.90	2.52	0.51	6.00	2.87	**2.64**	0.51	9.61	**2.90**	2.64	**0.57**	**9.92**
	10	**2.91**	2.62	**0.60**	8.33	2.85	**3.31**	0.54	9.67	2.89	2.95	0.57	**9.93**
	15	**2.92**	2.74	0.53	8.49	2.83	**3.42**	0.52	9.72	2.88	2.95	**0.53**	**9.89**
	20	**2.84**	2.75	0.52	9.24	2.84	**3.21**	0.52	9.69	2.84	2.95	**0.52**	**9.83**



(a) Ground-truth/Degraded (b) Total Variation



(c) $\mathcal{HS}_1$ regularizer (d) $\mathcal{HS}_2$ regularizer

Fig. 2: Maximum intensity projections of (a) the ground truth and the degraded image (note that the ground truth does not include the red channel, which would display as a uniform ellipsoid covering the cell), and (b)-(d) the restored images using different regularizers.

from the ISBI 2013 deconvolution challenge that simulate optically-sectioned digital micrographs of biological samples. These image stacks are of size $320 \times 320 \times 64$ and they reproduce four distinct sub-cellular structures that are typically observed in mitotic cells, i.e., point sources, filaments, membranes and dense volumetric features, corresponding to channels one to four, respectively. In Fig. 2a the maximum intensity projection of the entire synthetic cell model is shown, where each sub-cellular structure is represented with a different colour corresponding to the use of a different fluorescence marker. In this figure we juxtapose the cell image and the degraded version after applying optical blur and Gaussian noise. To simulate an optical blur that is typically encountered in widefield microscopy, we used dedicated software, which is freely available at `http://bigwww.epfl.ch/algorithms/psfgenerator/`, and we generated a PSF of size $129 \times 129 \times 127$ according to the Gibson-Lani PSF model. To provide comparisons with alternative regularization approaches, apart from the Hessian-based deconvolution software, we have also implemented on GPU a deconvolution method that relies on the total variation (TV) regularizer [13]. The results for a blurred SNR of 10 dBs are shown in Fig. 2b-2d. From Fig. 2 and

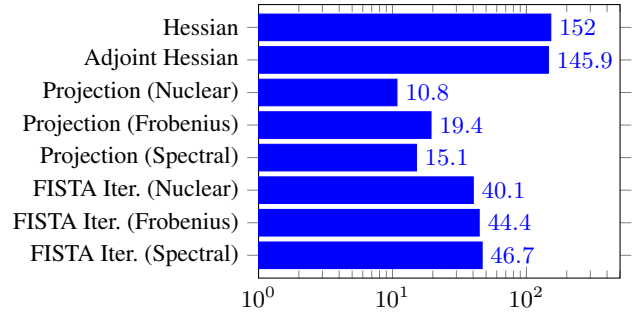


Fig. 3: Relative performance between different parts of the GPU and MATLAB implementation, calculated as t_{MATLAB}/t_{GPU}.

Table 1 we can verify that the Hessian-based regularizers seem more suitable for the reconstruction of biological data. Only for channel 1 (point sources), TV shows a slight advantage but the overall reconstruction is significantly better both visually and quantitatively using the Hessian regularizers.

To demonstrate the speed-up that we achieved through our GPU implementation, we compare it with a reference MATLAB implementation. The latter is parallelized and the critical paths are implemented in C through MATLAB's MEX file API. We run all experiments on a Mac Pro with a six core, 3.33GHz, Xeon W3680 CPU, 16GiB of ECC Memory, and an NVIDIA Quadro 5000K for Mac GPU. The display is driven by a separate graphics card (NVIDIA Quadro 4000) to ensure there is no interference between graphical display and CUDA computation. In Fig. 3 we report the relative performance of the Matlab and GPU implementations, calculated as t_{MATLAB}/t_{GPU}. In absolute terms, for a single iteration of the FISTA algorithm involving ten internal iterations for the evaluation of the proximal, the GPU versions of the $\mathcal{HS}_1$ and $\mathcal{HS}_2$ regularizers took on average 1.089 and 0.872 secs, while their MATLAB counterparts took 43.647 and 38.424 secs, respectively. In practice, this translates to 3D deconvolution in about one minute per channel.

6. CONCLUSIONS

To summarize, in this paper we described our approach of combining the most effective variational methods and optimization tools to build a 3D deconvolution system that can be applied in fluorescence microscopy. The central parts of our system involve proper handling of image boundaries, a suitable regularizer for biological data, and an optimization strategy that can be efficiently implemented on GPUs. Our GPU implementation achieves full 3D deconvolution using state-of-the-art signal processing in about one minute, which is very competitive with solutions in current use.

Acknowledgments: The authors would like to thank Cédric Vonesch for kindly providing the synthetic micrographs used in the 3D deconvolution experiments.

7. REFERENCES

[1] C. Vonesch, F. Aguet, J.-L. Vonesch, and M. Unser, "The colored revolution of bioimaging," *IEEE Sig. Process. Magazine*, vol. 23, pp. 20–31, 2006.

[2] D. A. Agard and J. W. Sedat, "Three-dimensional architecture of a polytene nucleus," *Nature*, vol. 302, no. 5910, pp. 676–681, April 1983.

[3] S. Lefkimmiatis, J. Ward, and M. Unser, "Hessian Schatten-norm regularization for linear inverse problems," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1873–1888, 2013.

[4] S. J. Reeves, "Fast image restoration without boundary artifacts," *IEEE Trans. Image Process.*, vol. 14, pp. 1448–1453, 2005.

[5] A. Matakos, S. Ramani, and J. A. Fessler, "Accelerated edge-preserving image restoration without boundary artifacts," *IEEE Trans. Image Process.*, vol. 22, pp. 2019–2029, 2013.

[6] M. S. C. Almeida and M. A. T. Figueiredo, "Deconvolving images with unknown boundaries using the alternating direction method of multipliers," *IEEE Trans. Image Process.*, vol. 22, pp. 3074–3086, 2013.

[7] P. C. Hansen, J. G. Nagy, and D. P. O'Leary, *Deblurring Images: Matrices, Spectra, and Filtering*, SIAM, 2006.

[8] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Foundations and Trends in Theoretical Computer Science*, vol. 4, no. 1, pp. 1–106, 2011.

[9] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imaging Sci.*, vol. 2, pp. 183–202, 2009.

[10] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Trans. Image Process.*, vol. 18, pp. 2419–2434, 2009.

[11] Nvidia, "Cufft library user manual," 2013.

[12] David Eberly, "Eigensystems for 3 x 3 symmetric matrices (revisited)," *Geometric Tools, LLC*, 2006.

[13] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259–268, 1992.