AN AUGMENTED LAGRANGIAN METHOD FOR FAST GRADIENT VECTOR FLOW COMPUTATION

Jianfeng Li, Wangmeng Zuo, Xiaofei Zhao, David Zhang

Biocomputing Research Centre, School of Computer Science and Technology
Harbin Institute of Technology, Harbin, 150001, China

ABSTRACT

Gradient vector flow (GVF) and its generalization have been widely applied in many image processing applications. The high cost of GVF computation, however, has restricted their potential applications to images with large size. In this paper, motivated by progress in fast image restoration algorithms, we reformulate the GVF computation problem as a convex optimization model with an equality constraint, and solve it using a fast algorithm, inexact augmented Lagrangian method (ALM). With fast Fourier transform (FFT), we provide a novel simple and efficient algorithm for GVF computation. Experimental results show that the proposed method can improve the computational speed by an order of magnitude, and is even more efficient for images with large sizes.

Index Terms — Gradient vector flow, convex optimization, augmented Lagrange multiplier, fast Fourier transform

1. INTRODUCTION

Active contour models, or snakes, have been extensively applied to many image segmentation tasks. Different from classical segmentation methods such as edge detection and thresholding, an active contour model deforms on the image domain to get the desired result by minimizing an energy functional which usually involves two components, an internal energy or regularization term, and an external energy term [1]. An active contour model can provide smooth, closed or open contours, and the internal energy functional allows incorporating prior knowledge on smoothness, shape, and region, which makes active contour very popular in medical image segmentation and other vision applications.

In general, active contour models can be classified into two categories, parametric and non-parametric (or geometric) models. The classical snake model introduced by Kass *et al*. [1] is a parametric active contour model, where the active contour is explicitly represented as a parametric curve $\mathbf{v}(s) = [x(s), y(s)]^T$. This model, however, requires the initial contour be closed to the true segmentation boundary, and exhibits poor convergence to boundary concavities. To address these problems, Xu and Prince [2, 3] proposed new types of external forces, gradient vector flow (GVF) forces and generalized GVF forces. By far, GVF snakes and its variants have been widely applied to medical image segmentation [2, 4], and have also been extended to other image processing problems, e.g., object tracking [5], skeletonization [6], image denoising [7], and image enhancement [8].

Despite its success and popularity, the GVF method requires a high computational cost, which has restricted their potential applications to images with large sizes. Li and Action [9] suggested another external force, vector field convolution (VFC), with reduced computation cost. Several fast numerical schemes, e.g., multiresolution method [10] and multigrid method [11], have been proposed for fast GVF computation. Most recently, Boukerroui [12] compared several efficient numerical schemes for GVF computation, and showed that the alternating direction explicit scheme (ADES) may be a suitable alternative to the multigrid method.

Actually, GVF computation can be reformulated as a convex optimization model, where similar problems have been extensively investigated in image restoration [13, 14], compressed sensing [15], and robust principal component analysis [16, 17]. One common way to derive an efficient algorithm is to split the original problem into several easier subproblems. In this paper, we consider two strategies, variable splitting [18] and Lagrangian method [13, 16], to design a fast GVF computation scheme. For the relationship between variable splitting and Lagrangian methods and other fast algorithms, e.g., split-Bregman and Douglas-Rachford splitting (DRS), see [18]. Experimental results show that the proposed method can improve the computational speed by an order of magnitude, and is even more efficient for images with large size.

The remainder of the paper is organized as follows. Section 2 introduces some background knowledge, including gradient vector flow and augmented Lagrangian method. Section 3 presents the proposed fast GVF computation scheme and discusses its implementation details. Section 4 provides the experimental results and Section 5 ends this paper with a few concluding remarks.

2. PREREQUISITES AND RELATED WORK

2.1. Gradient Vector Flow

In [2], given the edge map $f(x, y)$, GVF field is defined as a vector field $\mathbf{w}(x, y) = [u(x, y), v(x, y)]$ that minimizes the following energy functional,

$$E(\mathbf{w}(x,y)) = \iint \mu\left|\nabla\mathbf{w}\right|^2 + \left|\nabla f\right|^2 \left|\mathbf{w} - \nabla f\right|^2 dxdy\,, \qquad (1)$$

where $|\cdot|$ denote the vector norm for tensor with $\left|\nabla\mathbf{w}\right|^2 = \left|u_x^2 + u_y^2 + v_x^2 + v_y^2\right|$. Based on the calculus of variations, the GVF field can be obtained by solving the partial differential equation (PDE) problems:

$$u_t(x,y,t) = \mu\nabla^2 u(x,y,t) + \left|\nabla f\right|^2 [u(x,y,t) - f_x(x,y)]\,, \qquad (2)$$

$$v_t(x,y,t) = \mu\nabla^2 v(x,y,t) + \left|\nabla f\right|^2 [v(x,y,t) - f_y(x,y)]\,, \qquad (3)$$

where $u(x, y)$ and $v(x, y)$ can be obtained in parallel. Xu and Prince [2] adopted an explicit difference scheme. Boukerroui [12] tested several other numerical schemes, including the alternating direction explicit scheme (ADES), the additive operating splitting (AOS), and the locally one dimensional (LOD) methods, and showed that ADES was more appropriate for fast GVF computation.

Using the calculus of variations, the solution to (1) can be directly computed by seeking the solution to the following Euler-Lagrange equations,

$$0 = \mu\nabla^2 u(x,y) + \left|\nabla f\right|^2 [u(x,y) - f_x(x,y)]\,, \qquad (4)$$

$$0 = \mu\nabla^2 v(x,y) + \left|\nabla f\right|^2 [v(x,y) - f_y(x,y)]\,. \qquad (5)$$

In [11], Han *et al*. proposed an efficient GVF computation scheme which applies the full multigrid algorithm (FMG) to solve the above equations.

2.2. Variable Splitting and Augmented Lagrangian

2.2.1. Variable Splitting

Consider the following type of unconstrained optimization problem,

$$\min_{\mathbf{u}\in\mathbb{R}^n} f(\mathbf{u}) + g(\mathbf{G}\mathbf{u})\,, \qquad (6)$$

where $\mathbf{G} \in \mathbb{R}^{d\times n}$. Rather than directly solving the above problem, variable splitting reformulates the problem (6) as an equivalent constrained optimization problem,

$$\min_{\mathbf{u}\in\mathbb{R}^n, \mathbf{v}\in\mathbb{R}^d} f(\mathbf{u}) + g(\mathbf{v}), \quad \text{subject to } \mathbf{v} = \mathbf{G}\mathbf{u}\,, \qquad (7)$$

by introducing an auxiliary variable $\mathbf{v}$. In several image processing applications [13,14,15], it is much easier to solve problem (7) than to solve the unconstrained problem (6).

2.2.2. Augmented Lagrangian Method

Consider a convex optimization problem with equality constraints,

$$\min_{\mathbf{z}\in\mathbb{R}^n} F(\mathbf{z}), \quad \text{subject to } \mathbf{A}\mathbf{z} - \mathbf{b} = 0\,, \qquad (8)$$

where $\mathbf{b} \in \mathbb{R}^p$ and $\mathbf{A} \in \mathbb{R}^{p\times n}$. The augmented Lagrangian (AL) function is then defined as,

$$\mathcal{L}(\mathbf{z}, \boldsymbol{\lambda}, \sigma) = F(\mathbf{z}) + \boldsymbol{\lambda}^T(\mathbf{b} - \mathbf{A}\mathbf{z}) + \frac{\sigma}{2}\left\|\mathbf{A}\mathbf{z} - \mathbf{b}\right\|_2^2\,, \qquad (9)$$

where $\boldsymbol{\lambda} \in \mathbb{R}^p$ is a vector of Lagrangian multiplier and $\sigma > 0$ is the AL penalty parameter. As described in Algorithm 1 [14], the augmented Lagrangian method (ALM) [19], also known as the method of multipliers (MM), solves problem (9) by iteratively updating $\mathbf{z}$, $\boldsymbol{\lambda}$, and σ ,.until some convergence criterion is satisfied.

Algorithm 1: ALM/MM

1. Initialize $\mathbf{z}_0$, $\boldsymbol{\lambda}_0$, $\sigma_0 > 0$, $\rho > 0$
2. **while** not converged
3. $\quad \mathbf{z}_{k+1} = \arg\min_{\mathbf{z}} \{\mathcal{L}(\mathbf{z}, \boldsymbol{\lambda}_k, \sigma_k)\}$
4. $\quad \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \sigma_k(\mathbf{A}\mathbf{z}_{k+1} - \mathbf{b})$
5. $\quad \sigma_{k+1} = \rho\sigma_k$
6. $\quad k \leftarrow k + 1$
7. **end while**

2.2.3. Inexact Augmented Lagrangian Method

One can use ALM to solve the problem (7) by defining $F(\mathbf{z}) = f(\mathbf{u}) + g(\mathbf{v})$ and choosing

$$\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T, \quad \mathbf{b} = 0, \quad \mathbf{A} = [\mathbf{G}, -\mathbf{I}]\,. \qquad (10)$$

Here Steps 3 and 4 of Algorithm 1 become,

$$(\mathbf{u}_{k+1}, \mathbf{v}_{k+1}) = \arg\min_{\mathbf{u},\mathbf{v}} \left\{ f(\mathbf{u}) + g(\mathbf{v}) + \frac{\sigma_{k+1}}{2}\left\|\mathbf{G}\mathbf{u} - \mathbf{v}\right\|_2^2 + \boldsymbol{\lambda}_k^T(\mathbf{v} - \mathbf{G}\mathbf{u}) \right\} \qquad (11)$$

$$\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \sigma_k(\mathbf{G}\mathbf{u}_{k+1} - \mathbf{v}_{k+1})\,. \qquad (12)$$

For most problems, the solution to (11) is not trivial. Fortunately, we can use the inexact ALM (IALM) algorithm, which is also called the alternating direction method, to iterate between updating $\mathbf{u}_{k+1}$ by keeping $\mathbf{v}$ fixed and updating $\mathbf{v}_{k+1}$ by keeping $\mathbf{u}$ fixed, and still guarantee the convergence and optimality [16,19]. The detail of IALM is described in Algorithm 2.

Algorithm 2: IALM

1. Initialize $\mathbf{z}_0$, $\boldsymbol{\lambda}_0$, $\sigma_0 > 0$, $\rho > 0$
2. **while** not converged
3. $\quad \mathbf{u}_{k+1} = \arg\min_{\mathbf{u}} \left\{ f(\mathbf{u}) + \frac{\sigma_{k+1}}{2}\left\|\mathbf{G}\mathbf{u} - \mathbf{v}_k\right\|_2^2 - \boldsymbol{\lambda}_k^T(\mathbf{G}\mathbf{u}) \right\}$
4. $\quad \mathbf{v}_{k+1} = \arg\min_{\mathbf{v}} \left\{ g(\mathbf{v}) + \frac{\sigma_{k+1}}{2}\left\|\mathbf{G}\mathbf{u}_{k+1} - \mathbf{v}\right\|_2^2 + \boldsymbol{\lambda}_k^T\mathbf{v} \right\}$
5. $\quad \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \sigma_k(\mathbf{G}\mathbf{u}_{k+1} - \mathbf{v}_{k+1})$
6. $\quad \sigma_{k+1} = \min(\rho\sigma_k, \sigma_{max})$
7. $\quad k \leftarrow k + 1$
8. **end while**

3. FAST GVF COMPUTATION

The discrete version of energy functional can be rewritten as,

$$E(\mathbf{u},\mathbf{v}) = \mu\left(\left|\mathbf{D}_x\mathbf{v}\right|^2 + \left|\mathbf{D}_y\mathbf{v}\right|^2 + \left|\mathbf{D}_x\mathbf{u}\right|^2 + \left|\mathbf{D}_y\mathbf{u}\right|^2\right) + (\mathbf{u}-\mathbf{f}_x)^T\mathbf{M}(\mathbf{u}-\mathbf{f}_x) + (\mathbf{v}-\mathbf{f}_y)^T\mathbf{M}(\mathbf{v}-\mathbf{f}_y), \quad (13)$$

where $\mathbf{D}_x$ and $\mathbf{D}_y$ are the matrix representations of the gradient operator in x and y directions, respectively, and $\mathbf{M}$ is the diagonal weight matrix with $M(i,i) = f_x(i)^2 + f_y(i)^2$. $\mathbf{u}$ and $\mathbf{v}$ can be solved individually by solving the following two optimization problems:

$$\mathbf{u} = \arg\min_{\mathbf{u}} \mu\left(\left\|\mathbf{D}_x\mathbf{u}\right\|^2 + \left\|\mathbf{D}_y\mathbf{u}\right\|^2\right) + (\mathbf{u}-\mathbf{f}_x)^T\mathbf{M}(\mathbf{u}-\mathbf{f}_x), \quad (14)$$

$$\mathbf{v} = \arg\min_{\mathbf{u}} \mu\left(\left\|\mathbf{D}_x\mathbf{v}\right\|^2 + \left\|\mathbf{D}_y\mathbf{v}\right\|^2\right) + (\mathbf{v}-\mathbf{f}_y)^T\mathbf{M}(\mathbf{v}-\mathbf{f}_y). \quad (15)$$

Using variable splitting, the problem (14) can be formulated as an equivalent constrained problem,

$$\mathbf{u} = \arg\min_{\mathbf{u},\mathbf{u}'} \mu\left(\left\|\mathbf{D}_x\mathbf{u}\right\|^2 + \left\|\mathbf{D}_y\mathbf{u}\right\|^2\right) + (\mathbf{u}'-\mathbf{f}_x)^T\mathbf{M}(\mathbf{u}'-\mathbf{f}_x) \quad \text{subject to } \mathbf{u}' = \mathbf{u}, \quad (16)$$

The augmented Lagrangian function of (16) is given by,

$$\mathcal{L}(\mathbf{u},\mathbf{u}',\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\sigma) = \mu\left(\left|\mathbf{D}_x\mathbf{u}\right|^2 + \left|\mathbf{D}_y\mathbf{u}\right|^2\right) + (\mathbf{u}'-\mathbf{f}_x)^T\mathbf{M}(\mathbf{u}'-\mathbf{f}_x) + \boldsymbol{\lambda}^T(\mathbf{u}'-\mathbf{u}) + \frac{\sigma}{2}\left\|\mathbf{u}-\mathbf{u}'\right\|_2^2. \quad (17)$$

Here Step 3 of Algorithm 2 becomes

$$\mathbf{u}_{k+1} = \arg\min_{\mathbf{u}} \mu\mathbf{u}^T(\mathbf{D}_x^T\mathbf{D}_x + \mathbf{D}_y^T\mathbf{D}_y)\mathbf{u} - \boldsymbol{\lambda}_k^T\mathbf{u} + \frac{\sigma_k}{2}\left\|\mathbf{u}-\mathbf{u}'_k\right\|_2^2. \quad (18)$$

Similar to [13], with the help of fast Fourier transform (FFT), we can derive the closed-form solution of problem (18),

$$\mathbf{u}_{k+1} = FFT^{-1}\left\{\frac{FFT(\boldsymbol{\lambda}_k + \sigma_k\mathbf{u}'_k)}{2\mu FFT(\mathbf{D}_x^T\mathbf{D}_x + \mathbf{D}_y^T\mathbf{D}_y) + \sigma_k\mathbf{I}}\right\}. \quad (19)$$

Step 4 of Algorithm 2 becomes

$$\mathbf{u}'_{k+1} = \arg\min_{\mathbf{u}'}(\mathbf{u}'-\mathbf{f}_x)^T\mathbf{M}(\mathbf{u}'-\mathbf{f}_x) + \boldsymbol{\lambda}_k^T\mathbf{u}' + \frac{\sigma_k}{2}\left\|\mathbf{u}_{k+1}-\mathbf{u}'\right\|_2^2, \quad (20)$$

and the closed-form solution of $\mathbf{u}'_{k+1}$ is,

$$\mathbf{u}'_{k+1} = \left(2\mathbf{M} + \sigma_k\mathbf{I}\right)^{-1}(2\mathbf{M}\mathbf{f}_x - \boldsymbol{\lambda}_k + \sigma_k\mathbf{u}_{k+1}). \quad (21)$$

Finally, we describe the IALM-based algorithm for computing $\mathbf{u}$ in Algorithm 3. We can also apply this algorithm to compute $\mathbf{v}$ by solving the problem (15).

Algorithm 3: IALM-GVF

1. Initialize $\mathbf{u}_0$, $\mathbf{u}'_0$, $\boldsymbol{\lambda}_0$, $\sigma_0 > 0$, $\rho > 0$
2. **while** not converged
3. $\mathbf{u}_{k+1} = FFT^{-1}\left\{\frac{FFT(\boldsymbol{\lambda}_k + \sigma_k\mathbf{u}'_k)}{2\mu FFT(\mathbf{D}_x^T\mathbf{D}_x + \mathbf{D}_y^T\mathbf{D}_y) + \sigma_k\mathbf{I}}\right\}$
4. $\mathbf{u}'_{k+1} = \left(2\mathbf{M} + \sigma_k\mathbf{I}\right)^{-1}(2\mathbf{M}\mathbf{f}_x - \boldsymbol{\lambda}_k + \sigma_k\mathbf{u}_{k+1})$
5. $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \mu_k(\mathbf{u}'_{k+1} - \mathbf{u}_{k+1})$
6. $\mu_{k+1} = \min(\rho\mu_k, \mu_{max})$
7. $k \leftarrow k + 1$

3.1. Implementation Details

In the initialization, we choose $\mathbf{u}_0 = \mathbf{f}_x$, $\mathbf{u}'_0 = \mathbf{f}_x$. Although the IALM-GVF algorithm converges for any $\sigma_0 > 0$, the values of σ_0 and ρ do affect the convergence speed of the algorithm. In our work, for an $m \times m$ image, we empirically set $\sigma_0 = 0.5/m$, $\sigma_{\max} = 100\sigma_0$ and $\rho = 2$.

Fast Fourier transform usually involves some assumptions on boundary conditions, e.g., periodic or reflective. To alleviate the adverse influence of boundary condition, we first make a larger zero image $\mathbf{g}$ with the size of $1.14m \times 1.14m$, and then put the edge map $\mathbf{f}$ at the centre of the image $\mathbf{g}$. Finally, we use IALM-GVF on the larger image $\mathbf{g}$ to compute the GVF field $\mathbf{w}(x, y) = [u(x, y), v(x, y)]$.

There are several possible choices of the stopping criteria of IALM-GVF. One may stop the iteration when the following condition [14] is met,

$$\frac{\left\|\mathbf{z}_k - \mathbf{z}_{k-1}\right\|_2}{\left\|\mathbf{z}_{k-1}\right\|_2} \le \delta, \quad (22)$$

by choosing δ to be a sufficient small positive value.

4. EXPERIMENTAL RESULTS AND DISCUSSION

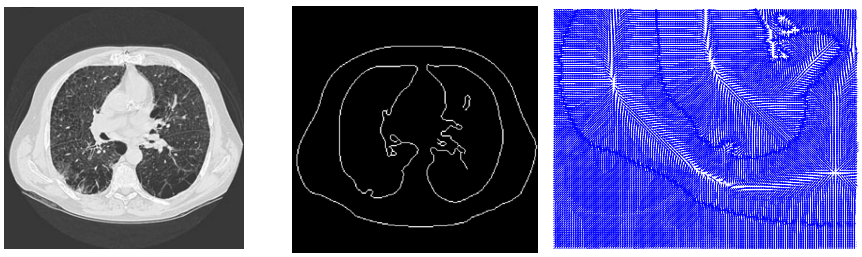
In this section, we used a set of human lung CT images to evaluate the computational efficiency of IALM-GVF, and compared the speed of original GVF, multigrid GVF (MGVF), and IALM-GVF. It should be noted that, MGVF is implemented in C, while IALM-GVF and original GVF in MATLAB. Thus, the speed difference between IALM-GVF and MGVF can only be used for reference.

We compare the computational cost and the number of iterations (*iter*) of the three GVF computation methods under the stopping criterion that the relative difference between the current solution and the converged solution is less than 0.001. We used a set of 2D human lung CT images with sizes from 256 × 256 to 1024 × 1024. The GVF regularization coefficient of the three methods is chosen to be 0.2. One image from the set is shown in Fig. 1.

Table1: Comparison of computational speed

	Original GVF		MGVF		IALM-GVF	
Size	Time, *s*	*Iter*	Time, *s*	*Iter*	Time, *s*	*Iter*
256×256	15.26	350	0.188	3	0.256	5
512×512	205.1	1024	0.670	3	0.962	5
1024×1024			3.153	3	3.926	5

The results are summarized in Table 1. Because the original GVF is hard to converge when the size of the image is higher, here we do not report the results of the original GVF on the 1024 × 1024 images. Compared with the original GVF, IALM-GVF can improve the computational speed by one or several orders of magnitude, and IALM-GVF would be more efficient when the size of image increases. One can see that IALM-GVF is a little slower



(a) (b) (c)

Fig. 1 GVF computation on a human lung CT image: (a) 256 × 256 image, (b) edge map, and (c) part of the GVF field

than MGVF. As we have mentioned, since MGVF is implemented in C while IALM-GVF in MATLAB, one cannot draw the conclusion that MGVF would be faster than IALM-GVF. Besides, IALM-GVF is simple to be understood and is easy to be implemented, which also makes IALM-GVF a suitable alternative of MGVF.

5. CONCLUSION

Motivated by recent progress in fast image restoration algorithms, based on variable splitting and augmented Lagrangian method, we proposed a novel inexact ALM algorithm for fast GVF computation (IALM-GVF). Experimental results show that IALM-GVF can improve the computational speed of the classical scheme [2] by an order of magnitude, and is even more efficient for images with large sizes. Moreover, IALM-GVF is guaranteed to converge to the global optimal solution, and is simple and easy to be implemented, which makes IALM-GVF a suitable alternative of multigrid GVF. In the future, we will implement IALM-GVF in C, and further modify and extend it to solve the generalized GVF computation problem.

Acknowledgements: The work is partially supported by the GRF fund from the HKSAR Government, the central fund from the Hong Kong Polytechnic University, the National S&T Major Project of China under Contract No. 2008ZXJ09004-035, and the NSFC funds of China under Contract No.s 60902099, 61071179, and 61001037.

11. REFERENCES

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snake: Active contour model," *IJCV*, vol. 1, no. 4, pp. 321-331, 1987.
[2] C. Xu, and J.L. Prince, "Snakes, shapes, and gradient vector flow," *IEEE Trans. IP*, vol. 7, no. 3, pp. 359–369, 1998.
[3] C. Xu, and J.L. Prince, "Generalized gradient vector flow external forces for active contours," *Signal Processing*, vol. 71, no. 2, pp. 131–139, 1998.
[4] N. Paragios, O. Mellina-Gottardo, and V. Ramesh, "Gradient vector flow fast geometric active contours," *IEEE Trans. PAMI*, vol. 26, no. 3, pp. 402-407, Mar. 2004.
[5] N. Ray, and S.T. Acton, "Motion gradient vector flow: An external force for tracking rolling leukocytes with shape and size constrained active contours, " *IEEE Trans. MI*, vol. 23, no. 12, pp. 1466-1478, 2004.
[6] M.S. Hassouna, and A.A Farag, "Variational curve skeletons using gradient vector flow," *IEEE Trans. PAMI*, vol. 31, no. 12, pp. 2257-2274, Dec. 2009.
[7] L. He, C. Li, and C. Xu, "Intensity statistics-based HSI diffusion for color photo denoising," *CVPR 2008*, Anchorage, AK, June 2008.
[8] O. Ghita, and P.F. Whelan, "A new GVF-based image enhancement formulation for use in the presence of mixed noise," *Pattern Recognition*, vol. 43, no. 8, pp. 2646-2658, August 2010.
[9] L. Bing, and S.T. Acton, "Active contour external force using vector field convolution for image segmentation," *IEEE Trans. IP*, vol. 16, no. 8, pp. 2096-2106, 2007.
[10] K. Ntalianis, N. Doulamis, A. Doulamis, and S. Kollias, "Multiresolution gradient vector flow field: a fast implementation towards video object plane segmentation," *ICME 2001*.
[11] X. Han, C. Xu, and J.L. Prince, "Fast numerical scheme for gradient vector flow computation using a multigrid method," *IET Image Processing*, vol. 1, pp. 1, pp. 48–55, 2007.
[12] D. Boukerroui, "Efficient numerical schemes for gradient vector flow," *ICIP 2009*, Nov. 2009, pp. 4057-4060.
[13] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, pp. 248–272, 2008.
[14] M.V. Afonso, J.M. Bioucas-Dias, and M.A.T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. IP*, vol. 19, no. 9, pp. 2345–2356, 2010.
[15] M.V. Afonso, J.M. Bioucas-Dias, and M.A.T. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Trans. IP*, DOI: 10.1109/TIP.2010.2076294, 2011.
[16] A. Ganesh, Z. Lin, J. Wright, L. Wu, M. Chen, and Y. Ma, "Fast algorithms for recovering a corrupted low-rank matrix," *International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2009.
[17] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," *ICML 2010*, Haifa, Israel, 2010.
[18] S. Setzer, "Operator splittings, Bregman methods and frame shrinkage in image processing," *IJCV*, DOI: 10.1007/s11263-010-0357-3, 2011.
[19] D.P. Bertsekas, *Nonlinear Programming*, 2nd edition, Arthena Scientific, 1999.