O(1) Algorithms for Overlapping Group Sparsity

Chen Chen, Zhongxing Peng and Junzhou Huang
Department of Computer Science and Engineering
University of Texas at Arlington
Arlington, TX 76019

***Abstract*—Sparsity based techniques have become very popular in machine learning, medical imaging and computer vision. Recently, with the emerging and development of structured sparsity, signals can be recovered more accurately. However, solving structured sparsity problems often involves much higher computational complexity. Few of existing works can reduce the computational complexity of such problems. Especially for overlapping group sparsity, the computational complexity for each entry is linear to the degree of overlapping, making it infeasible for large-scale problems. In this paper, we propose novel algorithms to efficiently address this issue, where the computational complexity for each entry is always $\mathcal{O}(1)$ and independent to the degree of overlapping. Experiments on 1D signal and 2D image demonstrate the effectiveness and efficiency of our methods. This work may inspire more scalable algorithms for structured sparsity.**

I. Introduction

Sparsity regularization has become one of the most successful techniques in the past decade, which benefits from both efficient algorithms (e.g. [1][2][3]) and compressive sensing theory [4][5]. It has been proved that sparse signals can be exactly recovered from fewer measurements than those indicated by Nyquist-Shannon sampling theorem [4][5]. These sparsity inducing algorithms have been widely used in practical applications, such as signal reconstruction [5], image denoising/deblurring [1], magnetic resonance imaging (MRI) [6], face recognition [7], background substraction [8] etc.

As an extension to conventional sparsity methods, structured sparsity has emerged recently [9][10]. Structured sparsity theories show that improvements can be gained if more prior information is utilized, e.g. exploiting structures of the non-zero entries of the signal. If non-zero entries cluster in groups, $\ell_{2,1}$ norm regularization can be used to exploit the group sparsity [11]. When the groups overlap, it comes to overlapping group sparsity [12], which could be used to solve clustered sparsity [13][8][14] and tree sparsity [15][16]. Overlapping group sparsity model is more generalized: if each entry is only in one group, it comes to the non-overlapping group sparsity; when each group contains only one entry, it is identical to standard sparsity with ℓ_1 norm regularization.

Although overlapping group sparsity can bring significant benefits, the computational complexity for each entry is often linear dependent on overlapping degree, which is much higher than that of conventional sparsity or non-overlapping group sparsity. In existing works, few of them study the computational complexity reduction. Generally, if the sparse signal has N entries and each entry overlaps in M groups, the computational complexity will be $\mathcal{O}(NM)$. For tree-based overlapping group sparsity, it costs $\mathcal{O}(N \log N)$ instead. Due to the computation issues, overlapping group sparsity may not feasible to be applied on severely overlapped signals or the applications that require real time processing.

To bridge this gap, efficient algorithms are proposed in this paper for different overlapping group sparsity problems. For clustered sparsity, the integral image [17] introduced in face detection is used for fast optimization. For tree sparsity, we build a bottom-up tree and a top-down tree to reduce computation. These algorithms can be extended to multi-channel data, e.g. RGB images and multi-spectral images. The computational costs for all these algorithms are all $\mathcal{O}(N)$, i.e., $\mathcal{O}(1)$ for each entry. The proposed algorithms can be widely applied on applications such as denoising, background substraction [8], face recognition [7], medical imaging [18] etc. Experiments on signal/image denoising and background substraction demonstrate the efficiency and effectiveness of our method. Future algorithms for different structured sparsity patterns may be inspired by our work.

II. Related Work

Consider the signal denoising problem with overlapping group sparsity regularization [12]:

$$\hat{x} = \arg\min_x \{\frac{1}{2}||x - v||_2^2 + \lambda \Sigma_{s=1}^S ||x_{g_s}||_2\} \qquad (1)$$

where $v \in \mathbb{R}^N$ is the noisy observation and λ is a positive parameter. g_s denotes the indices of the s-th group and S is the total number of groups. For example, $g_1 = [1, 2]$ and $g_2 = [2, 3]$ can be an overlapping pattern for a signal $x \in \mathbb{R}^3$. If this denoising problem can be solved, more complex problems such as reconstruction, deblurring then can be solved via proximal methods (e.g. [1]).

To solve the overlapping group sparsity regularization problem, a simple way is to convert the problem to a non-overlapping one by duplicating the overlapped entries [12]. Suppose each entry is assigned in M groups and each group contains M entries, the computation complexity of these algorithms is $\mathcal{O}(NM)$ in each iteration. YALL1 [20] uses Augmented Lagrange to relax the original problem and solve it with alternating direction method (ADM). Prime-dual methods SLEP [21] and GLO-pridu [22] solve this problem by iteratively identifying active groups (non-zero groups). These are the state-of-the-art methods for overlapping group sparsity regularization, but their speeds strongly depend on M. Very

(a) (b) (c)

Fig. 1. (a) A clustered sparse signal. The white and red nodes denote the zero and non-zero entries, respectively. (b) A frame in a benchmark surveillance video [19]. (c) The foreground object that segmented by groundtruth mask, which has the clustered sparsity property.

recently, a method called overlapping group shrinkage (OGS) [23] is proposed, which uses convolution operations to reduce the complexity to $\mathcal{O}(N \log M)$. However, OGS can only be applied on clustered sparse signals but is unknown how to be extended to tree sparsity. So far, the computational complexity of all existing methods depends on overlapping degree M.

III. Algorithm

Note problem (1) is convex but not smooth. We use majorization-minimization (MM) method [24], [25] to reformulate it:

$$\hat{x} = \arg\min_x \{\frac{1}{2}||x - v||_2^2 + \frac{\lambda}{2}\Sigma_{s=1}^S w_{g_s} ||x_{g_s}||_2^2\} \quad (2)$$

where $w_{g_s} \in \mathbb{R}$ is the weight for the s-th group. The convergence of the alternating algorithm is guaranteed by MM method [24] (also known as expectation-maximization method). (2) is quadratic. By taking the derivatives, x and w are updated iteratively until convergence:

$$w_{g_s}^k = 1/(||x_{g_s}^{k-1}||_2 + \epsilon) \text{ for } s = 1, 2, ..., S \quad (3)$$

$$x_i^k = v/(1 + \lambda \Sigma_{i \in g_s} w_{g_s}^k) \text{ for } i = 1, 2, ..., N \quad (4)$$

where a small constant ϵ is added to avoid the weight becoming infinity; k is the iteration index. The i-th entry of x corresponds all the weights that contain i. Using naive methods, (3) takes $\mathcal{O}(NM)$ additions and multiplications. (4) requires $\mathcal{O}(NM)$ additions but $\mathcal{O}(N)$ multiplications. Therefore, the total complexity is $\mathcal{O}(NM)$ in terms of multiplications. We'll demonstrate our accelerated $\mathcal{O}(N)$ algorithms on two concrete overlapping group sparsity patterns (also the most widely existing ones): clustered sparsity [13] and tree sparsity [15][16].

A. $\mathcal{O}(1)$ *Algorithm for Clustered Sparsity*

We first consider the clustered sparsity [13] for 2D images, where the non-zeros entries are highly correlated with its surrounding entries (e.g. Fig. 1 (a)). If an entry is non-zero/zero, its surrounding entries tend be to non-zeros/zeros with high probability. This pattern widely appears in computer vision tasks, such as the foreground image in background substraction [8] and the continuous occlusion on faces [7]. An example foreground image in video surveillance is shown in Fig 1 (c). It is suggested to estimate such signal by overlapping group sparsity [12][8], where each pixel and its neighboring pixels (e.g. 8 neighbors and $M = 9$) are assigned into one group. With this setting, the recovered non-zero entries will tend to be clustered [12][8].

Suppose the image X is n-by-n (i.e. $n^2 = N$) and the group size is m-by-m (i.e. $m^2 = M$). Both (3) and (4) are updated by rectangle areas of $m \times m$ pixels/weights. This sparsity pattern is like a sliding window or template operation. Two adjacent groups only slightly differ on boundaries, which motivates us to reduce the computation. OGS [23] is proposed to solve this problem using convolution. For example, the weight can be updated by $w^k = 1/(\sqrt{\mathrm{conv}(X^{k-1}. \times X^{k-1}, h)} + \epsilon)$, where h is a $m \times m$ template with all ones and ".” denotes the element-wise operation. Similar, X^k can be updated with the same template. OGS can reduce the complexity to $\mathcal{O}(N \log M)$.

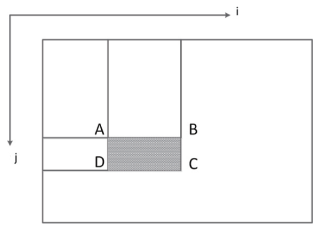


Fig. 2. The Integral image. The summation of an rectangle area can be obtained in constant time.

The proposed algorithm for clustered sparsity is inspired by the integral image that widely used in face detection [17]. The value at $(i, j) \in ([1, n], [1, n])$ on the integral image I is the summation of all pixels left and above (i, j) in the original image X:

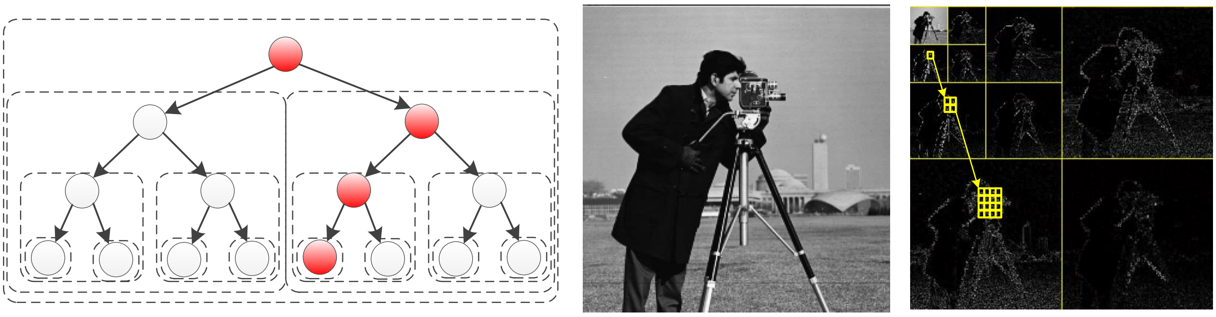
$$I(i, j) = \sum_{i' \leq i} \sum_{j' \leq j} X(i', j') \quad (5)$$

or a more efficient way:

$$I(i, j) = X(i, j) + I(i-1, j) + I(i, j-1) - I(i-1, j-1) \quad (6)$$

It cost $\mathcal{O}(N)$ memory and time to calculate this integral image. When we need the summation of pixels in a window $ABCD$ (Fig. 2), it can be obtained directly by:

$$\sum\sum_{((i',j') \in ABCD)} X(i', j') = I(i_A - 1, j_A - 1) + I(i_C, j_C) - I(i_B, j_B - 1) - I(i_D - 1, j_D) \quad (7)$$



(a) (b) (c)

Fig. 3. (a) A tree sparse signal. The white and red nodes denote the zero and non-zero entries, respectively. (b) The cameraman image. (c) The set of wavelet coefficients of the cameraman image, which has the tree sparsity property.

which only cost $\mathcal{O}(1)$ instead of summing all pixels in the window. We denote the operations (5) and (7) for all pixels (i.e. building and applying integral image) with constant window size (group size) $p \times p$, as $X = IntegralImage(I, p, p)$. For (3) and (4), we could build an integral image for the X^2 and an integral image for the weights. It cost $\mathcal{O}(N)$ time to build each integral image and then each entry/weight can be updated with $\mathcal{O}(1)$ time. Let V denotes the observed noisy image. The whole algorithm is summarized in Algorithm 1.

Algorithm 1 $\mathcal{O}(1)$ Clustered Sparsity

Input: λ, group size (window size) $m \times m$, V, $X^0 = V$.
for $k = 1$ **to** K **do**
1) $W^k = IntegralImage(X^{k-1}. \times X^{k-1}, m, m)$
2) $W^k = 1./(\sqrt{W^k} + \epsilon)$
3) $Y = IntegralImage(W^k, m, m)$
4) $X^k = V./(1 + \lambda Y)$
end for

B. $\mathcal{O}(1)$ Algorithm for Tree Sparsity

The wavelet coefficients of most nature 1D signal/2D image are approximately sparse, and naturally yield binary tree/quadtree structures. If a coefficient is non-zero, all of its ancestor nodes tend to be non-zeros. Such data has the tree sparsity property. An example of tree sparse signal is shown in Fig. 3. Improvements can be gained by exploiting the tree structure [9][10]. This problem can be solved by overlapping group sparsity [15][16], where each node and all its descendants are assigned into one group. The group setting in the example Fig. 3(a) is indicated by dashed boxes. Suppose there are $l = \log_2(N+1)$ levels, each leave node is overlapped in $l - 1$ groups and the root node is only assigned in one group. By duplicating the overlapped nodes, existing methods [15][16] have computational complexity $\mathcal{O}(N \log N)$.

It is not hard to find the different group are not independent. Similar the integral image, we could build the "integral tree" to reduce computation. To calculate the weights (3), we first build a bottom-up tree. Let $x \in \mathbb{R}^N$ denotes the original tree sparse signal and T_{bu} denotes the corresponding bottom-up tree. Therefore:

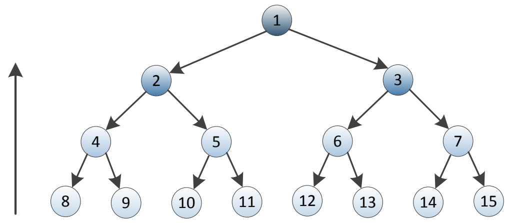
$$T_{bu}(i) = x(i) + T_{bu}(\text{children}(i)), \text{ for } i = N, N-1, ..., 1 \quad (8)$$

where the children of leaf nodes are defined as empty. An example of this bottom-up tree is shown in Fig. 4 (a). With such bottom-up tree, the weights in (3) can be updated in $\mathcal{O}(N)$ time instead of $\mathcal{O}(N \log N)$.

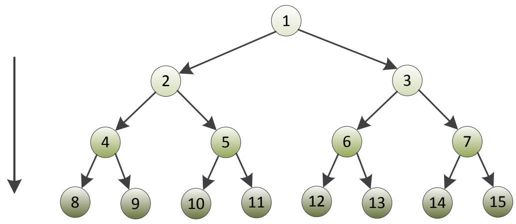
To update x in (4), each entry corresponds to all the weights that contain it. A top-down tree could reduce the computation:

$$T_{td}(i) = x(i) + T_{td}(\text{parent}(i)), \text{ for } i = 1, 2, 3, ..., N \quad (9)$$

where the parent of root node is defined as empty. An example is given in Fig. 4 (b).



(a)



(b)

Fig. 4. (a) An example of a bottom-up tree. For example, $T_{bu}(2) = x(2) + T_{bu}(4) + T_{bu}(5)$. (b) An example of a top-down tree. For example, $T_{td}(5) = x(5) + T_{td}(2)$.

We denote the operations (8) and (9) for all nodes as: $T = BottomUpTree(x)$ and $T = TopdownTree(x)$, respectively. We summarize the algorithm for tree sparsity in Algorithm 2. With the similar logic, the algorithm for quadtree structure can be derived. The total complexity for this algorithm is $\mathcal{O}(N)$, which is significantly better than $\mathcal{O}(N \log N)$ for existing methods such as [15][16]. Note that the OGS [23] can not solve tree sparsity regularization.

Algorithm 2 $\mathcal{O}(1)$ Tree Sparsity

Input: λ, v, $x^0 = v$.
for $k = 1$ **to** K **do**
1) $w^k = BottomUpTree(x^{k-1}. \times x^{k-1})$.
2) $w^k = 1./(\sqrt{w^k} + \epsilon)$
3) $y = TopdownTree(w^k)$
4) update $x^k = v./(1 + \lambda y)$.
end for

C. Extension to Multi-channel Signals

Our algorithms can be extended to multi-channel signals, e.g. RGB images. On the first and second coordinates, they have the clustered sparsity/tree sparsity. On the third coordinate (across different channels), they have joint sparsity property [26]. Different from that of 3D signals, this sparsity pattern is not symmetric on all coordinates. Therefore, the convolution in OGS can only be applied channel-by-channel. However, we could built multi-channel integral images and bottom-up/top-down trees for these cases to further reduce the computation. Suppose there are C channels in total and $I(i, j, c)$ denotes the integral image for the c-th channel, the multi-channel integral image MI at (i, j, c) can be build by:

$$MI(i, j, c) = MI(i, j, c - 1) + I(i, j, c) \text{ for } c = 2, ..., C \quad (10)$$

For a signal with C channels $X = [X_1, X_2, ..., X_C] \in \mathbb{R}^{n \times n \times C}$, we can find that all channels share the same weight matrix (i.e. $W \in \mathbb{R}^{n \times n}$). Building the multi-channel integral image (corresponds to (6)) takes $\mathcal{O}(CN)$ time while applying it to obtain the weights (corresponds to (7)) only cost $\mathcal{O}(N)$. After the weight matrix is obtained, the second step (4) can be solved channel-by-channel with this weight matrix. $\mathcal{O}(1)$ algorithm for multi-channel clustered sparse signal is presented in Algorithm 3. To avoid repetition, the algorithm for multi-channel tree sparsity is not shown.

Algorithm 3 $\mathcal{O}(1)$ Multi-channel Clustered Sparsity

Input: λ, group size (window size) $m \times m$, V, $X^0 = V$
for $k = 1$ **to** K **do**
1) $W^k = MIntegralImage(X^{k-1}. \times X^{k-1}, m, m)$
2) $W^k = 1./(\sqrt{W^k} + \epsilon)$
3) $Y = IntegralImage(W^k, m, m)$
4) $X^k_c = V_c./(1 + \lambda Y)$, $c = 1, 2, ..., C$
end for

We compare different algorithms and their computational complexity in Table I, which includes naive methods (e.g. [12][21]) without computation reduction, OGS [23], and the proposed method.

TABLE I
COMPUTATIONAL COMPLEXITY COMPARISON.

Sparsity Pattern	Naive	OGS	Proposed
Clustered	$\mathcal{O}(NM)$	$\mathcal{O}(N \log M)$	$\mathcal{O}(N)$
Multi-channel Clustered	$\mathcal{O}(CNM)$	$\mathcal{O}(CN \log M)$	$\mathcal{O}(CN)$
Tree	$\mathcal{O}(N \log N)$	N/A	$\mathcal{O}(N)$
Multi-channel Tree	$\mathcal{O}(CN \log N)$	N/A	$\mathcal{O}(CN)$

IV. EXPERIMENT

We validate the practical performance of our method on signal/image denoising. In addition, comparison with a $\mathcal{O}(1)$ algorithm on background substraction further validates the benefit of our method. All experiments are conducted using Matlab on a desktop with 3.4GHz Intel core i7 3770 CPU.

A. Image Denoising with Clustered Sparsity

First, an experiment is conducted on the clustered sparse image in Fig.1 (c). We add white Gaussian noise with 0.05 standard derivation on the image and compare the results with the original image for evaluation. SLEP [21] (one of the fastest algorithms for overlapping group sparsity) and OGS [23] are compared with the proposed Algorithm 1, which have $\mathcal{O}(M)$, $\mathcal{O}(\log M)$ and $\mathcal{O}(1)$ for each entry, respectively. The window size for group configuration is from 1×1 to 11×11. Peak signal-to-noise ratio (PSNR) are used as metric for result evaluation. For fairness, the shown results are the average over 100 runs.

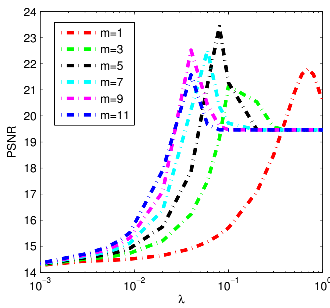
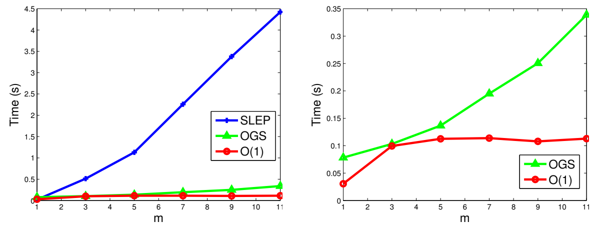


Fig. 5. Denoising performance for different group settings.

The denoising results with different parameters and different group size are shown in Fig. 5. As the problem is convex, SLEP, OGS, and the proposed method converge to the same results. We could observe that overlapping group sparsity is better than conventional sparsity (i.e. group size is 1) for this data. In this experiment, the optimal group size is 5×5. For different data and different levels of noise, the optimal group size may change.

Although the accuracy of different overlapping group sparsity methods are the same, their computational complexities differ significantly. The speed comparisons are presented at Fig. 6 for different group sizes. The computational cost of SLEP is much higher than those of OGS and the proposed, especially when the group size increases. A close look at Fig. 6 (b) demonstrates the efficiency of the proposed over OGS.



(a) (b)

Fig. 6. Denoising performance on the sparse image in Fig.1(b). (a) CPU time comparison among SLEP, OGS and the proposed. (d) CPU time comparison between OGS and the proposed.

It is obvious that our method always cost constant time and is independent to the degree of overlapping. These results are expected and validate the advantage of our method.

B. Signal Denoising with Tree Sparsity

We then conduct an experiment for tree sparsity. Recall that $SLEP$ cost $\mathcal{O}(N \log N)$ and the proposed algorithm cost $\mathcal{O}(N)$ in this case. A random signal with binary tree sparsity is simulated, which is corrupted by additive white Gaussian noise with 0.05 standard derivation. The size N of the signal varies from $2^6 - 1$ to $2^{18} - 1$.

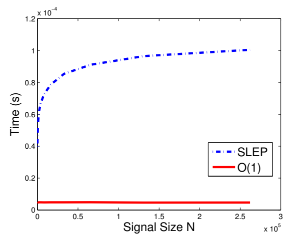


Fig. 7. Computational cost for each entry in tree sparsity denoising.

The computational cost comparison is shown in Fig. 7. For each entry, the cost of SLEP increases in logarithmic speed, while the speed our method is constant. These results are consistent with the theories. The benefit of tree sparsity over conventional sparsity is not shown specially, as it has been validated in many previous works. Due to the $\mathcal{O}(1)$ feature, the proposed algorithms can be potentially applied to many applications that require real time processing, which is not feasible for existing non-$\mathcal{O}(1)$ algorithms. Next, we will show the performance of our method on one of these applications.

C. Real Time Background Substraction

Finally, the proposed method is applied on a practical application: background substraction for video surveillance. In a survey [27], it is said that "Since interesting changes are often associated with localized groups of pixels, it is common for the change decision at a given pixel to be based on a small block of pixels in the neighborhood". This idea coincides with that of clustered sparsity. "However, this option is computationally more expensive". Due to this reason, algorithms that utilize the block/clustered structures are rarely used in real time background substraction. With the proposed $\mathcal{O}(1)$ algorithms, the computational cost is not an issue any more. We compare the proposed Algorithm 3 with Vibe [28], which is a $\mathcal{O}(1)$ algorithm for background substraction. In this application, v in (1) is the difference of the current frame and the background image. The group size is 5×5 and the parameter is fixed to be 0.006 for all image sequences. We use the default parameters for Vibe. $Precision = TruePositive/(TruePositive + FalsePositive)$ is used for evaluation.

The results on 4 frames of the benchmark datasets [19] are shown in Fig. 8. Due to advantage of overlapping group sparsity, it is obviously that the proposed algorithm is more robust to noise, shadows and the perturbation of background. The overall precisions in these experiments are 0.9447 and 0.9730 for Vibe and the proposed, respectively. This is also because Vibe [28] does not consider the dependence of pixels as ours does. Currently, the proposed algorithm is slightly slower than Vibe, as ours is implemented in Matlab but Vibe is in C++. Our method can be accelerated by optimizing the code or implemented in parallel. The current Matlab version can run approximately 30 fps for 360×240 images.

V. Conclusion

We have proposed several $\mathcal{O}(1)$ algorithms for overlapping group sparsity in this paper. It is motivated by that the groups often do not randomly overlapped, but have special structures. The integral image and bottom-up/top-down trees are used to reduce the computational complexity. Experiments on 1D signals with tree sparsity, 2D images with clustered sparsity have demonstrate the efficiency and effectiveness of our method. This work makes overlapping group sparsity more feasible for applications that require computational efficiency.

References

[1] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[2] S. Wright, R. Nowak, and M. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Trans. Signal Process.*, vol. 57, pp. 2479–2493, 2009.

[3] J. Huang, S. Zhang, H. Li, and D. Metaxas, "Composite splitting algorithms for convex optimization," *Computer Vision and Image Understanding*, vol. 115, no. 12, pp. 1610–1622, 2011.

[4] D. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[5] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[6] M. Lustig, D. Donoho, and J. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magn. Reson. Med.*, vol. 58, pp. 1182–1195, 2007.

[7] J. Wright, A. Yang, A. Ganesh, S. Shankar, and M. Yi, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, 2009.

[8] J. Huang, X. Huang, and D. Metaxas, "Learning with dynamic group sparsity," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2009, pp. 64–71.

[9] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, "Model-based compressive sensing," *IEEE Trans. Inf. Theory*, vol. 56, pp. 1982–2001, 2010.

Fig. 8. Performance comparison for background substraction. (a) The original frames. (b) The results obtained by the proposed Algorithm 3. (c) The masks segmented from (b) with a simple threshold. (d) The results obtained by Vibe. (e) The groundtruth.

[10] J. Huang, T. Zhang, and D. Metaxas, "Learning with structured sparsity," in *Proc. IEEE Int. Conf. Mach. Learn. (ICML)*, 2009.

[11] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[12] L. Jacob, G. Obozinski, and J. Vert, "Group lasso with overlap and graph lasso," in *Proc. IEEE Int. Conf. Mach. Learn. (ICML)*, 2009.

[13] V. Cevher, P. Indyk, C. Hegde, and R. Baraniuk, "Recovery of clustered sparse signals from compressive measurements," in *Int. Conf. on Sampling Theory and Applications (SAMPTA)*, 2009.

[14] C. Chen, F. Tian, J. Yao, H. Liu, and J. Huang, "2D diffuse optical imaging using clustered sparsity," in *Proc IEEE Int. Symp. Biomed. Imaging. (ISBI)*, 2014.

[15] J. Liu and J. Ye, "Moreau-yosida regularization for grouped tree structure learning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2010, pp. 1459–1467.

[16] S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," in *Proc. IEEE Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 543–550.

[17] P. Viola and M. Jones, "Face detection," *Int. J. Comput. Vis.*, vol. 57, p. 2, 2004.

[18] C. Chen and J. Huang, "Compressive sensing MRI with wavelet tree sparsity," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1124–1132.

[19] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection. net: A new change detection benchmark dataset," in *Proc. IEEE Workshop on Change Detection (CDW-12) at CVPR*, 2012, pp. 1–8. [Online]. Available: www.changedetection.net

[20] W. Deng, W. Yin, and Y. Zhang, "Group sparse optimization by alternating direction method," *TR11-06, Department of Computational and Applied Mathematics, Rice University*, 2011.

[21] L. Yuan, J. Liu, and J. Ye, "Efficient methods for overlapping group lasso," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, pp. 2104–2116, 2013.

[22] S. Mosci, S. Villa, A. Verri, and L. Rosasco, "A primal-dual algorithm for group sparse regularization with overlapping groups," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2010, pp. 2604–2612.

[23] P. Chen and I. Selesnick, "Translation-invariant shrinkage/thresholding of group sparse signals," *Signal Process.*, vol. 94, pp. 476–489, 2014.

[24] D. Hunter and K. Lange, "A tutorial on MM algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.

[25] C. Chen, J. Huang, L. He, and H. Li, "Preconditioning for accelerated iteratively reweighted least squares in structured sparsity reconstruction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogn. (CVPR)*, 2014.

[26] C. Chen, Y. Li, and J. Huang, "Forest sparsity for multi-channel compressive sensing," *IEEE Transactions on Signal Processing*, vol. 62, no. 11, pp. 2803 – 2813, 2014.

[27] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, "Image change detection algorithms: a systematic survey," *IEEE Trans. Image Process.*, vol. 14, no. 3, pp. 294–307, 2005.

[28] O. Barnich and M. Van Droogenbroeck, "Vibe: A universal background subtraction algorithm for video sequences," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, 2011.