Data Separation of ℓ_1-minimization for Real-time Motion Detection

Yu Liu[1]
jasonyuliu@nudt.edu.cn
Huaxin Xiao[1]
xiaohuaxin@nudt.edu.cn
Zheng Zhang[1]
zhangzheng@nudt.edu.cn
Wei Xu[1]
xuwei@nudt.edu.cn
Maojun Zhang[1]
mjzhang@nudt.edu.cn
Jianguo Zhang[2]
jgzhang@computing.dundee.ac.uk

[1] College of Information System and Mangement
National University of Defense Technology
Changsha, China

[2] School of Computing
University of Dundee
Dundee DD1 4HN, U.K.

Abstract

The ℓ_1-minimization used to seek the sparse solution restricts the applicability of compressed sensing. This paper proposes a data separation algorithm with computationally efficient strategies to achieve real-time performance of sparse model based motion detection. We use the traditional pursuit algorithms as a pre-process step that converts the iterative optimization into linear addition and multiplication operations. A novel motion detection method is implemented to compare the difference between the current frame and the background model in terms of sparse coefficients. The influence of dynamic texture or statistical noise diminishes after the process of sparse projection; thus, enhancing the robustness of the implementation. Results of the qualitative and quantitative evaluations demonstrate the higher efficiency and effectiveness of the proposed approach compared with those of other competing methods.

1 Introduction

Motion detection is the problem of segmenting moving objects from a given image sequence or surveillance video. This topic has drawn considerable attention in the field of computer vision and video processing over the past decades. Three types of methods have been developed in the existing literature, namely, optic flow, frame difference, and background subtraction (BGS). The most prevalent approach is BGS, which establishes a background model through a certain method, and then calculates the difference between the current frame and the background to segment the foreground area. One notable work of BGS is the statistical background model [9, 12, 21, 27], which has been adequately researched and developed in

Pages 75.1-75.12
DOI: https://dx.doi.org/10.5244/C.29.75

the past years. Then, the principal component analysis (PCA) [17, 18] was employed to capture the spatial configurations instead of pixel-wise processing. Previously, numerous BGS methods with various theories have been proposed, such as artificial neural networks [15], classification model [1], texture descriptor [19], and so on. Above mentioned approaches and algorithms can be categorised as classic BGS methods.

The successful development of sparsity resulted in the introduction of sparsity [3, 11, 20, 23, 25] as a watershed for motion detection. Cevher et al. [3] regarded background subtraction as a sparse approximation problem, and obtained a low-dimensional compressed representation of the background. Subsequently, the foreground was directly recovered via the compressed sensing (CS) theory. Huang et al. [11] added a prior of group sparsity clustering as a new constraint in the process of sparse recovery and extended the CS theory to manage the dynamic background scenes efficiently. Sivalingam et al. [20] and Zhao et al. [25] regarded the foreground mask as a problem of sparse error during their utilization of dictionary atoms to represent a new frame. Subsequently, Xiao et al. [23] extended the assumption of CS for BGS [3] by adding an assumption that the projection of the noise over the dictionary is irregular and random. Recently, low rank and sparse decomposition [5, 13, 26] becomes a novel direction for the sparse model based motion detection. This sort of methods employs the coherence of continuous frames and applies iterative optimization to obtain the foreground and background simultaneously.

The sparse model of BGS achieves more competitive results over classic approaches, especially performs more robust on adaptive and noisy images [14, 23]. In order to seek the solution of sparse model, a veriety of pursuit algorithms have been proposed. From the simplest matching pursuit (MP) [16] or orthogonal matching pursuit (OMP) [22] to the relatively efficient Bregman iterative algorithms [24].However, these optimization algorithms are not fast enough for real-time implementation. A more efficient sparse model based motion detection algorithm is imperative.

The key innovation of this work is the introduction of a novel decomposition scheme of sparse optimization for motion detection. This idea is inspired by the theory of data separation of sparse representations [8]. We simplify the process of ℓ_1-minimization and consider it as a pre-process step. In the proposed approach, the test/observed signal is separated into a number of basic atoms. For each atom, the sparse coefficient is calculated using the pre-learned dictionary with the existing ℓ_1-minimization algorithms. After iterating all these atoms, the same number of children sparse vectors can be obtained. We assume that any observed/test data can be linearly represented by those atoms. Consequently, the sparse coefficient can also be regarded as the linear combination of children sparse vectors. Given that the children sparse vectors calculation is a pre-processing step, the ℓ_1-minimization can be simplified into addition and multiplication operations.

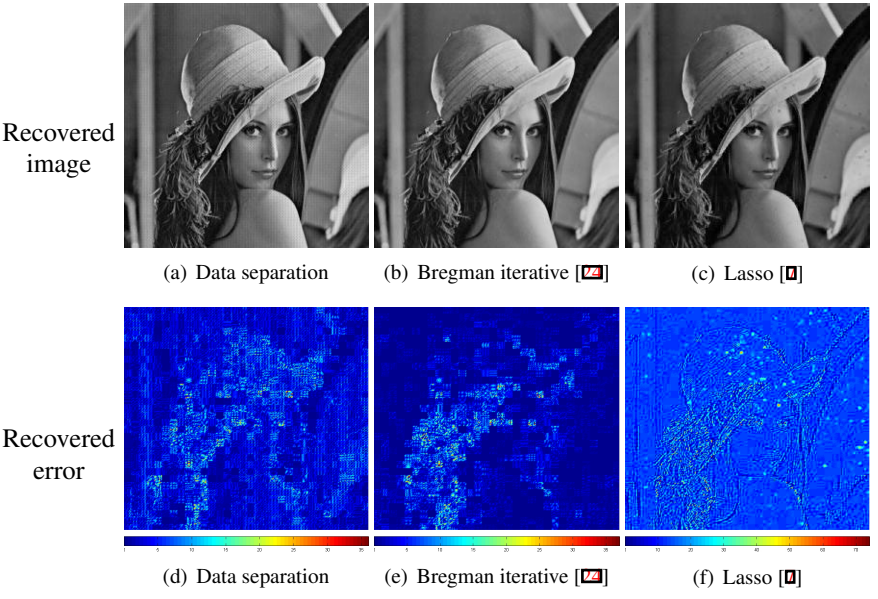
2 Data Separation of ℓ_1-Minimization

For a given signal $y \in \mathbb{R}^m$, the sparse model is a process of pursuing the sparsest solution $\alpha \in \mathbb{R}^n$ of y over a given dictionary $D \in \mathbb{R}^{m \times n}$ as follows:

$$P_1: \quad \hat{\alpha} = \arg\min \|\alpha\|_1 \qquad s.j. \;\; D\alpha = y. \tag{1}$$

where, in the problem of motion detection, foreground is defined as the sparse error $\varepsilon = y - D\hat{\alpha}$ [20], where $D\hat{\alpha}$ can be regarded as the background model.

Commonly, the problem P_1 is a transformation of the original sparse model with replacing the ℓ_0 norm with the ℓ_1 norm. This is because of the non-convex characteristic of the ℓ_0 norm. Ideally, it is not capable of finding the sparsest solution except by exhausting the entire subspaces of α. P_1 can be solved in polynomial by standard convex programming methods [4] or other optimization algorithms [7, 16, 22, 24]. Nevertheless, these existing ℓ_1 minimization approaches are computationally expensive because the convex value is optimized in an iterative manner. This work proposed a data separation approach to tackle this problem.



(a) Data separation (b) Bregman iterative [24] (c) Lasso [7]

(d) Data separation (e) Bregman iterative [24] (f) Lasso [7]

Figure 1: Comparison of recovered results with different ℓ_1-minimization algorithm. (a)-(c): The recovered Lena image(256×256) by proposed Data separation, Bregman iterative [24], and Lasso [7]. Their execution time of recovery are 0.12s, 100.16s, and 4.18s, respectively. (d)-(e): The recovered error by proposed Data separation, Bregman iterative [24], and Lasso [7]. The percentage of recovered error in pixel level is 0.0331, 0.0176 and 0.1317 respectively.

For a given signal $y \in \mathbb{R}^m$, the proposed approach separates y as the linear combination of the basis vectors e_i, which is defined as atoms in this paper:

$$y = \gamma_1 e_1 + \cdots + \gamma_i e_i + \cdots \gamma_m e_m, \tag{2}$$

where γ_i is the coefficient of y over the basis vectors e_i, and it could be computed easily by simply projecting y on a basic vector e_i, i.e., the i-th coordinate of y. For example, the image vector in this work is separated to the smallest atoms e_i which contains one non-zero standard element as:

$$e_i = \left[0, 0, \cdots, \underbrace{1}_{i}, \cdots, 0, 0\right]^T. \tag{3}$$

The signal is not limited to the atom used in this work, but can be separated into a variety of patterns of atoms. Each e_i can be considered as the observed signal in (1) and convert P_1

as follows:

$$P_1^{e_i}: \quad \hat{\beta}_i = \arg\min \|\beta_i\|_1 \qquad s.j. \;\; D\beta_i = e_i, \tag{4}$$

where β_i is defined as the children sparse vector in this paper.

Most data can be classified as multimodal data composed of irrelevant subcomponents, such as imaging data from neurobiology, which are typically composed of the soma of a neuron, dendrites, and spines [8]. Donoho and Huo [6] suggested that choosing distinct bases that are adapted to different subcomponents will allow separation. Inspired by this idea, we assume that the sparse solution α of y can be separated into the linear combination of its children sparse vector β_i as follows:

$$\alpha \approx \gamma_1\beta_1 + \cdots + \gamma_i\beta_i + \cdots \gamma_m\beta_m. \tag{5}$$

For the motion detection problem, we can pre-solve the children sparse vector β_i with the existing ℓ_1-minimization algorithms. Then, the sparse solution of a new signal or image y can be rapidly obtained by Eq. (5). Then, the iterative process in the existing ℓ_1 algorithms is replaced by addition and multiplication operations.

Another important question is about the numerical distance of the sparse solution between the use of the classic ℓ_1-minimization and data separation algorithm. The distance is acceptable for many applications, (e.g., motion detection), but not for others, (e.g., image deblurring). If tolerable in a specific work, distance can be used as acceleration engine, which can dramatically improve applicability. The numerical distance can be visually represented as the recovered error in Fig. 1. Although the recovered results are not the best, the proposed data separation approach can significantly accelerate the ℓ_1-minimization.

3 Motion detection

3.1 Sparse model for motion detection

The background subtraction problem is usually formulated as a linear combination of a background model I_B and a foreground candidate I_F. The key idea of the sparse model of motion detection is the application of dictionary D. Generally, dictionary D has various forms, such as a pre-learned [23] or updating online [25]. The atoms d in dictionary D represent the bases of image signal or the special configurations. Subsequently, the background model in motion detection can be represented as the sparse and linear combination of the atoms in the dictionary D :

$$I_B = D \times \alpha, \tag{6}$$

where α is the sparse coefficients that can be computed by linear programming P_1 in Eq. (1).

According to background subtraction, the foreground I_F is the difference between the current frame and the background model I_B:

$$I_F = I - I_B = I - D \times \alpha, \tag{7}$$

where foreground I_F is formulated as a sparse error.

In the classic sparse model of motion detection, the background update process serves to update the sparse coefficients α or both of the coefficients α and dictionary D in each frame or couple of frames according to the implementation requirements. The computation of sparse coefficients α or updating dictionary D accounts for most of the computing resources and restricts the real-time implementation of above model.

3.2 Data separation for motion detection

The proposed method improve the classic model in two aspects, namely, (1) data separation acceleration and (2) the comparison of the sparse coefficients.

We construct the image atoms e_i in Eq. (3) as the same size as the background I_B. The Bregman iterative algorithm [24] is employed to calculate the children sparse vector β_i of e_i. Consequently, the background model in Eq. (6) can be rewritten as follows:

$$I_B = D \times \alpha \approx D \times \sum_i \gamma_i \beta_i, \tag{8}$$

where γ_i is linear coefficients of the background model I_B over the atom e_i.

Detection requires a high level of consideration to enhance the robustness because the dynamic textures or complicated environment can affect the corrupted frame I in Eq. (7). In this work, we project the current frame I over the pre-learned dictionary D and compute the sparse codes α' with the data separation algorithm. Then, the formula is converted as follows:

$$I_F = D \times \alpha' - D \times \alpha \approx D \times \sum_i \gamma'_i \beta_i - D \times \sum_i \gamma_i \beta_i = D \times \sum_i (\gamma'_i - \gamma_i) \beta_i, \tag{9}$$

where γ'_i is linear coefficients of current frame I over the atom e_i.

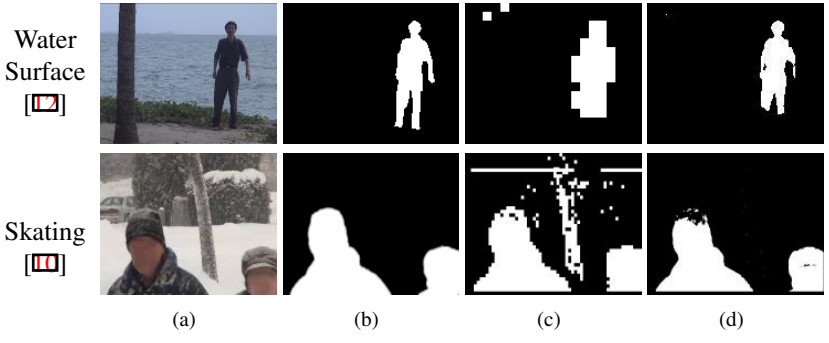


Figure 2: Patch refinement. (a) Two frames are extracted from water surface [12] and skating [10] and which reflect dynamic scenes. (b): Ground truth. (c)-(d): First and second stage detection results with proposed method.

Accordingly, we compare each patch to decide whether the frame belongs to the foreground through the distribution of the sparse coefficients.

$$\begin{cases} \Delta_1 = \left\| \sum_i \gamma'_i \beta_i - \sum_i \gamma_i \beta_i \right\|_1, \\ \Delta_2 = \left| \left\| \sum_i \gamma'_i \beta_i \right\|_0 - \left\| \sum_i \gamma_i \beta_i \right\|_0 \right|, \end{cases} \tag{10}$$

where Δ_1 and Δ_2 are the differences of sparse coefficients distributions and values between the background model and the current frame. Given that the distributions and values reflect which subspace is expanded by the test frame, we can use these parameters to decide whether the content of the monitoring scene has moving activity. Specifically, if the image content remains the same, it tends to have identical distributions and corresponding values. By contrast, if the foreground object enters the scene and changes the content, it generates distinct distributions.

To obtain a more precision result, we post-process the differences of the sparse coefficients as shown in follow:

$$\Delta = \lambda_1 \Delta_1 + \lambda_2 \Delta_2, \tag{11}$$

where λ_1 and λ_2 are the unitary parameters which determine the weight of Δ_1 and Δ_2 respectively. Since ℓ_1 norm can better represent the distribution of the sparse coefficient and make the difference more distinguishable, λ_1 is then set to be a relatively larger value (0.70) as the dominant weight, while λ_2 is 0.30.

The dictionary based on patches leads to unsatisfactory exhibition in precision as shown in Fig. 2(c). To obtain a pixel level result, the proposed method includes two-stage foreground detection. The first stage roughly detects the foreground in the patch, and the second stage refines the detection results on top of the first stage. For each foreground block, we use a smaller sliding window to determine whether the central pixel belongs to the foreground. Pixel-wise refinement as shown in Fig. 2(d) achieves more precise detection results. This two-stage approach can speed up the process of detection compared with single sliding window operation. The whole data separation of motion detection method is described in Algorithm 1.

Algorithm 1: Data separation of motion detection

Input : Image sequence $I_1, \cdots, I_N$;
Output: The binary foreground $I_1^F, \cdots, I_N^F$;

Data separation (pre-process):

Separate each image I_i into M independent patches: $P_i^1, \cdots, P_i^M \in \mathbb{R}^{m \times 1}$;
Construct atom e_j likes Eq. (3) as the same dimension as the image patch P, $j \in [1, \cdots, m]$;
Compute the children sparse vectors β_j of e_j by Bregman iterative [24];

Motion detection:

Set the patch of background model $P_B^k = P_1^k \approx D \times \sum_j \gamma_j^k \beta_j$, $k \in [1, \cdots, M]$;
for $i \leftarrow 2$ **to** N **do**
    for $k \leftarrow 1$ **to** M **do**
        Compute the foreground difference $P_F^k = D \times \sum_i \gamma_i^k \beta_i - P_B^k$;
        Obtain the binary foreground by threshold selection with Eq. (10) and (11);
        if P_i^k *is detected as foreground* **then** Refine the Pathc P_i^k;
        Update the background model P_B^k;

4 Experiments

To evaluate the performance of the proposed method, the study experiments the proposed method in two parts: one is tested on public datasets [10, 17] and the other one is tested on corrupted video signal. The size of the tested videos is 160×128. The sizes of the dictionary in the two-stage detection are 8×8 pixels with 256 atoms in the first stage and 3×3 pixels with 256 atoms in the second stage. Under above spatial resolution and patch sizes, the proposed method deals with a frame in 0.03-0.05 seconds on MATLAB implementation.

The execution time is recorded by a laptop with a 2.50 GHz Intel Core i7-4710MQ processor and 16GB of RAM.

In this section, we qualitatively and quantitatively compare the proposed method with classic motion detection algorithms improved mixtures of Gaussian model (MoG) [27] and kernel density estimate (KDE) model [9], advanced algorithms SOBS [15] and ViBe [1], sparse model Xiao et al. [23] and low rank model DECOLOR [26]. For all algorithms, we experiment with adjustments of parameters until results seem optimal on the tested dataset.

4.1 Dynamic scenes

The movement in the captured scenes can be divided into two parts. One part is the foreground, which is an independent object and has no relationship with the scene. The other part is periodical or irregular, such as rain, snow, waves, and shaking trees and should be classified as the background based on its relevance to the scene. Therefore, the ability to distinguish the two kinds of movement becomes an important criterion for motion detection.

We compare various motion detection approaches with our method under diverse dynamic scenes as shown in Fig. 3. The testing frames are extracted from datasets, namely, curtain [12], fountain [12], water surface [12] in which different types of dynamic turbulences, such as the curtain blown by the wind (Fig. 3(a)), flowing water (Figs. 3(b), 3(c), 3(f)), or falling snowflakes (Fig. 3(d), 3(e)). We find that the detection results of traditional statistical model (rows 2 to 3 of Fig. 3) are not satisfactory because the statistical model or kernel function failed to manage the irregular turbulence well. The imprecise model leads to unsatisfactory results. SOBS [15] and the proposed method effectively eliminate the influence of the dynamic textures and detect the foreground precisely. The low rank and sparse model DECOLOR [26] and Xiao et al. [23] achieves more stable performance but lacks efficiency. While the data separation algorithm is able to help those sparse approaches more competitive on processing time without sacrificing the effectiveness.

4.2 Corrupted scenes

Brutzer et al. [2] reviewed numerous methods used in noisy night videos and concluded that, the results of this experiment show quite low performance for all evaluated approaches. The classic motion detection methods are focused more on handling complex and dynamic scenes, such as rain, snow, waves, and shaking trees, and do not consider the quality of the images. In real applications, image signals are polluted in such cases as low light surveillance (Figs. 4(d)-4(f)), heat in the sensor, or turbulence in the signal transmission, in which high level of noise or corrupted signal causes existing algorithms to perform inappropriately (rows 3 to 7 of Fig. 4).

We explore the robustness of different motion detection methods for corrupted signal as shown in Fig. 4. The datasets of Figs. 4(a)-4(c) are extracted from [10], whereas the Figs. 4(d)-4(f) are captured by SONY IMX 104CMOS. In Figs. 4(a)-4(b), we add synthetic noise to the original frames to simulate the low signal to noise ratio (SNR). The compared classic approaches are obviously affected by noise in different degrees, whereas our method performs robustly and manages noise efficiently. The sparse method of Xiao et al. [23] is designed to cope with noisy sences and obtain a better results than classic methods. However, the proposed data separation method not only accelerates the efficiency of Xiao et al. [23] but also refines the cursory results. DECOLOR [26] appears to be robust to noise, but its accuracy is much lower than that of the previously proposed method (row 8 of Fig. 4).

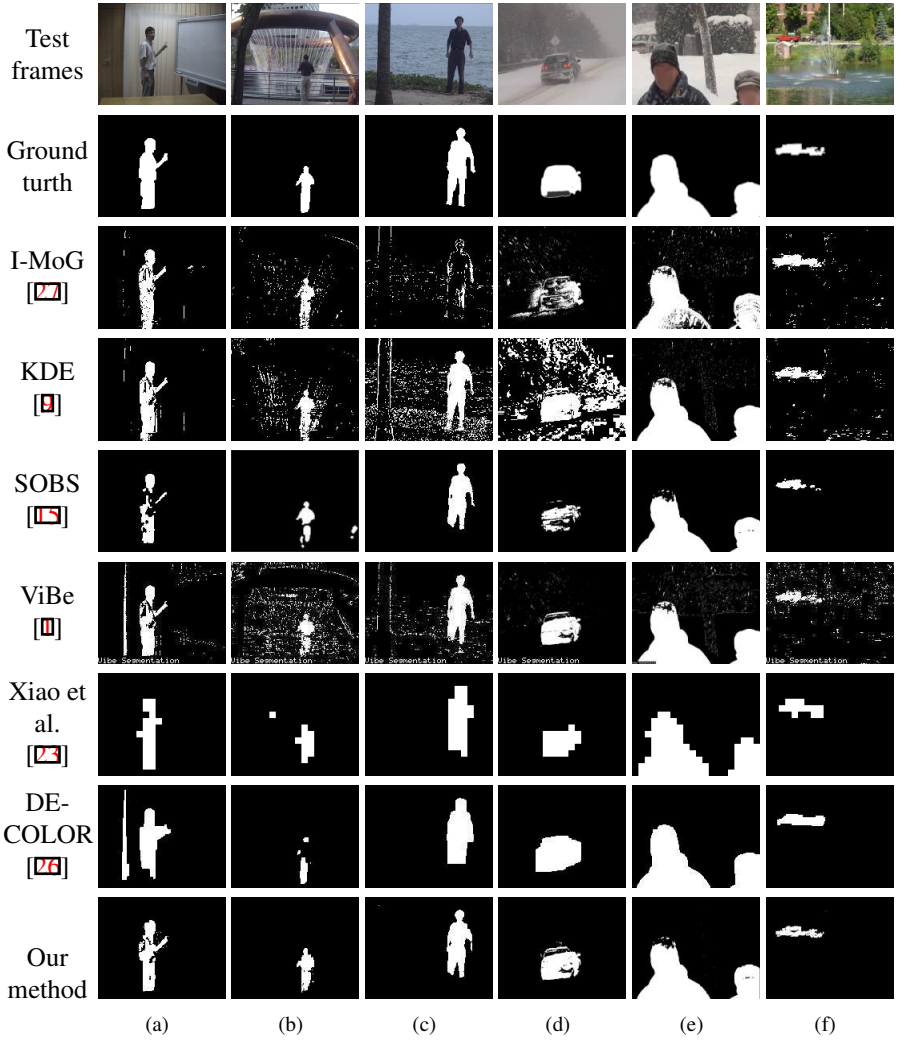


Figure 3: Comparison of detection results on dynamic videos. The testing frames are extracted from dataset Curtain [12], Fountain [12], Water surface [12], Snow fall [10], Skating [10] and Fountain02 [10] that exists dynamic turbulence such as the curtain blown by the wind, flowing water or the falling snowflakes. 1st row: testing frames; 2nd row: ground truth; 3rd to 6th rows are the detection results of classic methods including improved MoG [27], KDE model [9], SOBS [15] and ViBe [1]; 7th to 9th are the detection results of sparse methods including Xiao et al. [23], DECOLOR [26] and our method.

4.3 Quantitative results

The quantitative performance of these algorithms on pixel-level is evaluated in this section. This study adopts three different quantitative metrics: Recall, Precision and F-measure from [15]. And, the execution time of each presented methods is also considered. We regard the processed frames per second (fps) as the quantitative criterion for the computational

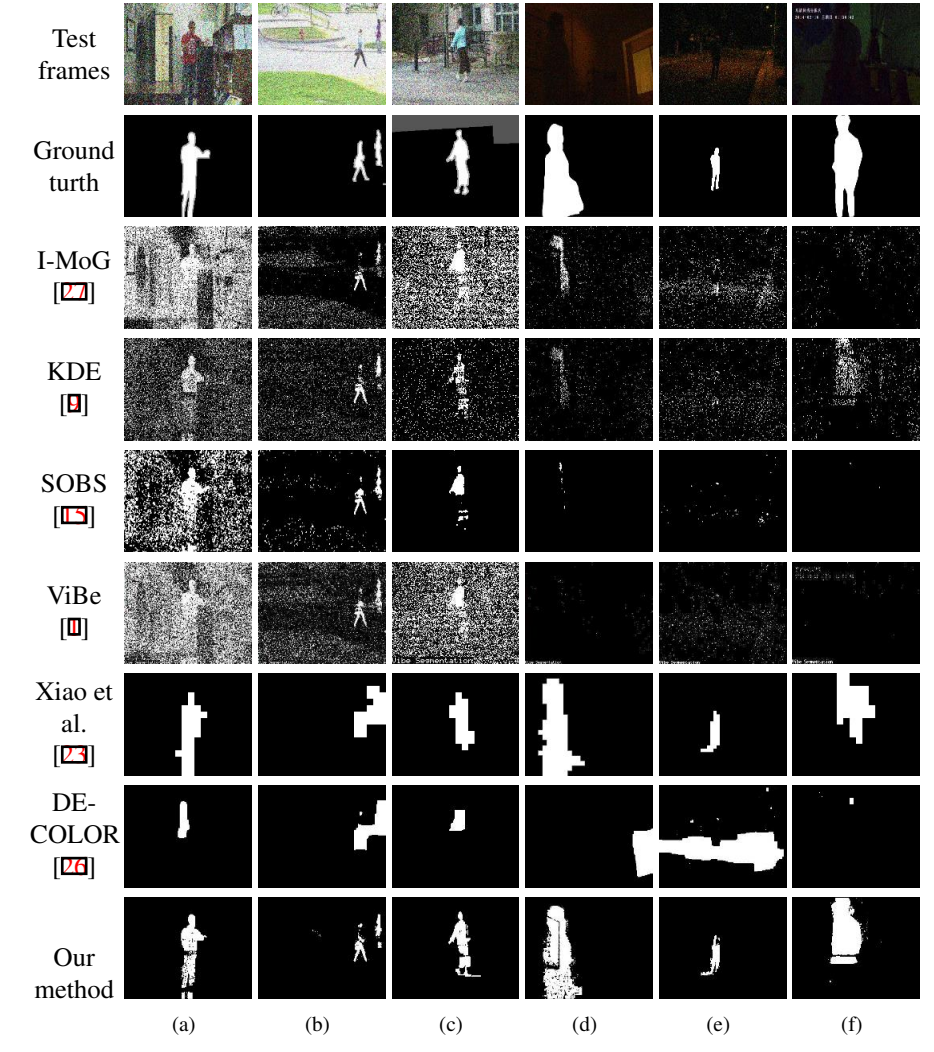


Figure 4: Comparison of detection results on corrupted videos. The testing frames are extracted from dataset Office [10], Pedestrian [10], Back door [10] or Low light video taken by ourselves. We added synthetic noise that contains a mixture of Gaussian white noise ($\sigma = 30$) and Poisson noise (scale factor $\rho = 12$) to the first three datasets. The illumination of last two videos is about 1.5-2.0 **lx**, 0.7-1.0 **lx** and 0.3-0.5 **lx**, respectively. 1st row: testing frames; 2nd row: ground truth; 3rd to 6th rows are the detection results of classic methods including improved MoG [27], KDE model [9], SOBS [15] and ViBe [1]; 7th to 9th are the detection results of sparse methods including Xiao et al. [23], DECOLOR [26] and our method.

efficiency.

Table 1 shows the results of the quantitative metrics over four datasets. All of the testing datasets in Table 1 provide available ground truth as a binary mask. The last two are the basic motion detection datasets, to which we add synthetic noise. The compared methods

have poor performances because false detection was affected by noise, whereas the proposed method operates well in terms of quantity regardless of the noise signal. Since the classic ℓ_1 minimization is optimized in an iterative manner, the sparse model based approach Xiao et al. [23] and DECOLOR [26] can only achieve 4 fps and 1 fps respectively which is relatively slow for realistic applications. While the proposed method significantly improves the efficiency (around 30 fps) by converting the iterative optimization into linear addition and multiplication operations. Note that these operations also facilitate implementations on hardware, which help the CS more practical.

Table 1: The quantitative metrics of compared methods. Recall, Precision and F-measure are selected as the quantitative performances of the compared methods while the computational efficiency is evaluated by the criterion of fps. (Best: Bold; Second best: Underline)

Dataset	Metrics	Classic methods				Sparse methods		
		I-MoG [24]	KDE [9]	SOBS [15]	ViBe [8]	Xiao et al. [23]	DECOLOR [26]	Ours
Curtain [12]	Recall	0.6721	0.8198	0.8821	0.7120	0.8758	**0.9433**	0.9382
	Precision	0.7833	**0.9352**	0.8892	0.8672	0.8155	0.9202	0.9223
	F-measure	0.7235	0.8737	0.8856	0.7820	0.8446	**0.9316**	0.9302
	Fps	15.6	12.0	30.1	31.9	3.8	0.87	29.8
Fountain [13]	Recall	0.7912	0.7550	**0.9731**	0.8201	0.8025	0.7980	0.9233
	Precision	0.6300	0.6039	0.9324	0.5241	0.7741	0.7512	**0.9690**
	F-measure	0.7015	0.6710	**0.9523**	0.6396	0.7880	0.7739	0.9456
	Fps	18.2	15.8	32.5	33.3	4.3	1.1	30.2
Office with noise [10]	Recall	0.5230	0.4109	0.5663	0.5273	0.7322	0.4582	**0.7922**
	Precision	0.2052	0.3060	0.5627	0.1828	0.8634	0.5133	**0.9525**
	F-measure	0.2948	0.3508	0.5645	0.2715	0.7924	0.4842	**0.8650**
	Fps	10.2	8.5	24.0	29.3	3.8	2.1	25.0
Pedestrian with noise [10]	Recall	0.4052	0.3631	0.5628	0.5963	**0.6594**	0.5922	0.6168
	Precision	0.1060	0.1252	0.4608	0.0844	0.6984	0.4752	**0.8523**
	F-measure	0.1680	0.1862	0.5067	0.1479	0.6783	0.5273	**0.7157**
	Fps	11.5	9.8	28.3	30.1	5.3	1.8	26.3

5 Conclusion

Sparse model based motion detection methods are more robust and adaptive [14]. However, the iterative optimization process to seek the sparse solution is computationally expensive. This paper proposed a data separation algorithm to fill the gap of efficiency where previous sparse model based approaches could not reach.

This work is at very preliminary stages. How the signal separated into basic atoms has remained an open question. A satisfactory result can be obtained in separating the signal even with the use of the simplest method as demonstrated by this work. Simultaneously, many other approaches can potentially define and describe the basic features or atoms of an image signal better. Another future work is to measure the numerical difference of the sparse solution with or without using data separation. The difference is acceptable for motion detection, but it does not mean it can be used for other applications. Thus, mathematically defining this difference is able to decide the potential of the data separation algorithm.

References

[1] O. Barnich and M. V. Droogenbroeck. Vibe: A universal background subtraction algorithm for video sequences. *IEEE T-IP*, 20(6):1709–1724, 2011.

[2] S. Brutzer, B. Hoferlin, and G. Heidemann. Evaluation of background subtraction techniques for video surveillance. In *Proc. CVPR*, 2011.

[3] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa. Compressive sensing for background subtraction. In *Proc. ECCV*, 2008.

[4] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43(1):129–159, 2001.

[5] X. Cui, J. Huang, S. Zhang, and D. N. Metaxas. Background subtraction using low rank and group sparsity constraints. In *Proc. ECCV*, 2012.

[6] D. Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *IEEE T-IT*, 47(7):2845–2862, 2001.

[7] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[8] Y. C. Eldar and G. Kutyniok (Eds.). *Compressed sensing: theory and applications. Chapter 11*. Cambridge University Press, ISBN: 9781107005587, 2012.

[9] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *Proc. ECCV*, 2000.

[10] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection.net: A new change detection benchmark dataset. In *Proc. CVPRW*, 2012.

[11] J. Z. Huang, X. L. Huang, and D. Metaxas. Learning with dynamic group sparsity. In *Proc. ICCV*, 2009.

[12] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE T-IP*, 13(11):1459–1472, 2004.

[13] X. Liu, G. Zhao, J. Yao, and C. Qi. Background subtraction based on low-rank and structured sparse decomposition. *IEEE T-IP*, 24(8):2502–2514, 2015.

[14] Y. Liu, H. Xiao, W. Wang, and M. Zhang. A robust motion detection algorithm on noisy videos. In *Proc. ICASSP*, 2015.

[15] L. Maddalena and A. Petrosino. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE T-IP*, 17(7):1168–1177, 2008.

[16] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE T-SP*, 41(12):3397–3415, 1993.

[17] A. Monnet, A. Mittal, N. Paragios, and V. Ramesh. Background modeling and subtraction of dynamic scenes. In *Proc. ICCV*, 2003.

[18] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE T-PAMI*, 22(8):831–843, 2000.

[19] V. Reddy, C. Sanderson, and B. C. Lovell. Improved foreground detection via block-based classifier cascade with probabilistic decision integration. *IEEE T-CSVT*, 23(1): 83–93, 2013.

[20] R. Sivalingam, A. D'Souza, M. Bazakos, V. Morellas, and N. Papanikolopoulos. Dictionary learning for robust background modeling. In *Proc. ICRA*, 2011.

[21] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. CVPR*, 1999.

[22] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE T-IT*, 50(10):2231–2242, 2004.

[23] H. Xiao, Y. Liu, S. Tan, J. Duan, and M. Zhang. A noisy videos background subtraction algorithm based on dictionary learning. *KSII T-TTS*, 8(6):1946–1963, 2014.

[24] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for compressed sensingand related problems. *SIAM J. Imaging Sciences*, 1(1):143–168, 2008.

[25] C. Zhao, X. Wang, and W. Cham. Background subtraction via robust dictionary learning. *EURASIP J. Image and Video Processing*, 2011.

[26] X. Zhou, C. Yang, and W. Yu. Moving object detection by detecting contiguous outliers in the low-rank representation. *IEEE T-PAMI*, 35(3):597–610, 2013.

[27] Z. Zivkovic and F. van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7): 773–780, 2006.