Online and Batch Supervised Background Estimation via L1 Regression

Online and Batch Supervised Background Estimation via L1 Regression

Aritra Dutta
KAUST
aritra.dutta@kaust.edu.sa

Peter Richtárik
KAUST, Edinburgh, MIPT
peter.richtarik@kaust.edu.sa

Abstract

We propose a surprisingly simple model for supervised video background estimation. Our model is based on ℓ_1 regression. As existing methods for ℓ_1 regression do not scale to high-resolution videos, we propose several simple and scalable methods for solving the problem, including iteratively reweighted least squares, a homotopy method, and stochastic gradient descent. We show through extensive experiments that our model and methods match or outperform the state-of-the-art online and batch methods in virtually all quantitative and qualitative measures.

1. Introduction

Video background estimation and moving object detection is a classic problem in computer vision. Among several existing approaches, one of the most prevalent ones is to solve it in a matrix decomposition framework [6, 8]. Let $A \in \mathbb{R}^{m\times n'}$ be a matrix encoding n' video frames, each represented as a vector of size m. Our task is to decompose all frames of the video into background and foreground frames: $A = B + F$.

As described above, the problem is ill-posed, and more information about the structure of the decomposition is needed. In practice, background videos are often static or close to static, which typically means that B is of low rank [39]. On the other hand, foreground usually represents objects occasionally moving across the foreground, which typically means that F is sparse. These and similar observations leads to the development of models of the form [8, 6, 55, 31, 14]:

$$\min_B f_{\rm rank}(B) + f_{\rm spar}(A - B), \tag{1}$$

where $f_{\rm rank}$ is a suitable function that encourages the rank of B to be low, and $f_{\rm spar}$ is a suitable function that encourages the foreground F to be sparse.

Xin *et al*. [56] recently proposed a background estimation model—generalized fused lasso (GFL)—arising as a special case of [20] with the choice $f_{\rm rank}(B) = {\rm rank}(B)$ and $f_{\rm spar}(F) = \lambda\|F\|_{\rm GFL}$:

$$\min_B {\rm rank}(B) + \lambda\|A - B\|_{\rm GFL}. \tag{2}$$

In this model, $\|\cdot\|_{GFL}$ is the "generalized fused lasso" norm, which arises from the combination of the ℓ_1 norm (to encourage sparsity) and a local spatial total variation norm (to encourage connectivity of the foreground).

Supervised background estimation. In the modern world, *supervised* background estimation models play an important role in the analysis of the data captured from the surveillance cameras. As the name suggests, these models rely on prior availability of some "training" background frames, $B_1 \in \mathbb{R}^{m\times r}$. Without loss of generality, assume that the training background frames correspond to the first r frames of B, i.e., $B = [B_1\ B_2]$, where $B_1 \in \mathbb{R}^{m\times r}$ is known and $B_2 \in \mathbb{R}^{m\times n}$ is to be determined, with $n' = r + n$. Let $A = [A_1\ A_2]$ be partitioned accordingly, and let $F_2 = A_2 - B_2 \in \mathbb{R}^{m\times n}$. In this setting, [56] further specialized the model (2) by adding the extra assumption that ${\rm rank}(B) = {\rm rank}(B_1)$.As a result, the columns of the unknown matrix B_2 can be written as a linear combinations of the columns of B_1. Specifically, B_2 can be written as B_1S, where $S \in \mathbb{R}^{r\times n}$ is a coefficient matrix. Thus, problem (2) can be written in the form

$$\min_{S'} {\rm rank}(B_1[I\ S']) + \lambda\|A_2 - B_1S'\|_{\rm GFL}. \tag{3}$$

While (6.1) is the the problem Xin *et al*. [56] *wanted* to solve, they did not tackle it directly and instead further assumed that S is sparse, and solved the modified problem

$$\min_{S'} \|S'\|_1 + \lambda\|A_2 - B_1S'\|_{\rm GFL}, \tag{4}$$

where $\|\cdot\|_1$ denotes the ℓ_1 norm of matrices.

2. New Model

In this paper we propose a new supervised background estimation model, one that we argue is much better than (4) in several aspects. Moreover, our model and the methods we propose significantly outperform other state-of-the-art methods.

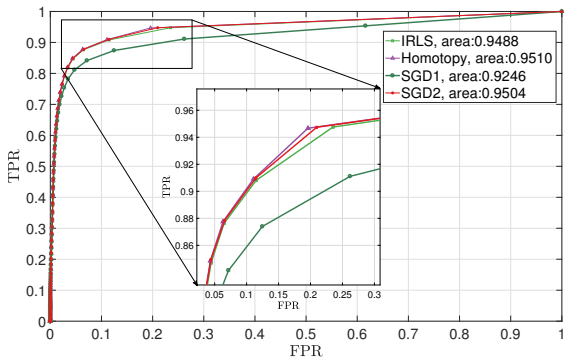


Figure 1: ROC curve to compare between our proposed ℓ_1 regression algorithms on `Basic` video, frame size 144×176.

L1 regression. As in (4), our model is also based on a modified version of (3). We do not need to assume any sparsity on S', and instead make the trivial observation that $\text{rank}(B_1[I\ S']) = \text{rank}(B_1)$. Since B_1 is known, the first term in the objective function (3) is constant, and hence does not contribute to the optimization problem. Hence we may drop it. Moreover, we suggest replacing the GFL norm by the ℓ_1 norm. This leads to a very simple *L1 (robust) regression problem*:

$$\min_{S' \in \mathbb{R}^{r \times n}} \|A_2 - B_1 S'\|_1. \tag{5}$$

Dimension reduction. The above model can be further simplified. It may be the case that the rank of $B_1 \in \mathbb{R}^{m \times r}$ is smaller[1] (or much smaller) than r. In such a situation, we can replace B_1 in (5) by a thinner matrix, which allows us to reduce the dimension of the optimization variable S'. In particular, let $B_1 = QR$ be the QR decomposition of B_1, where $Q \in \mathbb{R}^{m \times k}$, $R \in \mathbb{R}^{k \times r}$, $k = \text{rank}(B_1)$, and Q has orthonormal columns. Since the column space of B_1 is the same as the column space of Q, by using the substitution $B_1 S' = QS$, we can reformulate (5) as the *lower-dimensional* L1 regression problem:

$$\boxed{\min_{S \in \mathbb{R}^{k \times n}} f(S) := \|A_2 - QS\|_1} \tag{6}$$

Decomposition. Let $A_2 = [a_1, \ldots, a_n]$ and $S = [s_1, \ldots, s_n]$, where $a_i \in \mathbb{R}^m$, $s_i \in \mathbb{R}^t$ for all $i \in [n] := \{1, 2, \ldots, n\}$. Our model (6) can be decomposed into n parts, one for each frame:

$$f(S) = \sum_{i=1}^{n} f_i(s_i), \quad f_i(s_i) := \|a_i - Q s_i\|_1, \tag{7}$$

[1] If this is not the case, it still may be the case that the column space of B_1 can be very well approximated by a space with less or much less than r dimensions.

where $\|\cdot\|_1$ is the vector ℓ_1 norm. Therefore, (6) reduces to n small (k-dimensional) and independent ℓ_1 *regression problems:*

$$\boxed{\min_{s_i \in \mathbb{R}^t} f_i(s_i), \qquad i \in [n]} \tag{8}$$

Advantages of our model. We now list some advantages of our model (6) as compared to (4). We show that 1) our model does not involve the unnecessary sparsity inducing term $\|S'\|_1$, that 2) our model does not include the trade-off parameter λ and hence issues with tuning this parameter disappear, that 3) our model involves a simple ℓ_1 norm as opposed to the more complicated GFL norm, that 4) the dimension of S is smaller (and possibly much smaller) than that of S', that 5) our objective is separable across the n columns of S corresponding to frames, which means that we can solve for each column of S in *parallel* (for instance on a GPU), and that 6) for the same reason, we can solve for each frame as it arrives, in an *online* fashion.

Further contributions. Our model works well with just a few training background frames (e.g., $r = 10$). This should be compared with the 200 training frames in GFL model. We propose 5 methods for solving the model, out of which 4 can work online and all 5 can work in a batch mode. Our model solves all the following challenges: static and semi-static foreground, newly added static foreground, shadows that are already present in the background and newly created by moving foreground, occlusion and disocclusion of the static and dynamic foreground, the ghosting effect of the foreground in the background. To the best of our knowledge, no other algorithm can solve all the above challenges in a single framework.

3. Scalable Algorithms for L1 Regression

The separable (across frames) structure of our model allows us to devise both batch and online background estimation algorithms. To the best of our knowledge, this is the first formulation which can operate in both batch and online mode. Since our problem decomposes across frames $i \in [n]$, it suffices to describe algorithms for solving the ℓ_1 regression problem (8) for a single i. This problem has the form

$$\min_{x \in \mathbb{R}^t} \phi(x) := \|Qx - b\|_1 = \sum_{j=1}^{m} |q_j^\top x - b_j|, \tag{9}$$

where $x \in \mathbb{R}^t$ corresponds to one of the reconstruction vectors s_i, and $b \in \mathbb{R}^m$ corresponds to the related frame a_i. We write $b = (b_1, \ldots, b_m) \in \mathbb{R}^m$, and let $q_j \in \mathbb{R}^t$ be the jth row of Q for $j \in [m]$.

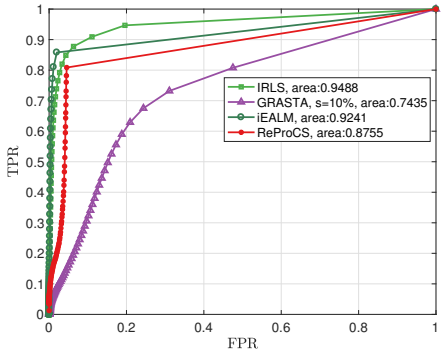


Figure 2: ROC curve to compare between IRLS, iEALM, GRASTA, and ReProCS on `Basic` video, frame size 144×176.

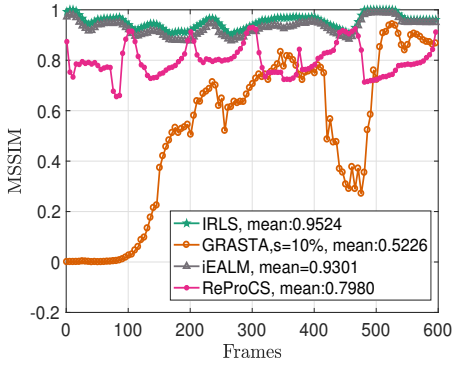


Figure 3: Comparison of Mean SSIM (MSSIM) of IRLS, iEALM, GRASTA, and ReProCS on `Basic` video. IRLS has the best MSSIM. To process 600 frames each of size 144×176, iEALM takes 164.03 seconds, GRASTA takes 20.25 seconds, ReProCS takes 14.20 seconds, and our IRLS takes 7.51 seconds.

Five methods. In this work we propose to solve (9) via four algorithms: (a) iteratively reweighted least squares (IRLS), (b) homotopy method, (c) stochastic subgradient descent (variant 1), (d) stochastic subgradient descent (variant 2), and (e) Augmented Lagrangian Method of Multipliers (ALM) (see Appendix 2).

The first four algorithms can be used in both batch and online setting and can deal with grayscale and color images. If we assume the camera is static, and assume constant illumination throughout the video sequence, then our online methods can provide a good estimate of the background. Moreover, all algorithms are robust to the intermittent object motion artifacts, that is, static foreground (whenever a foreground object stops moving for a few frames), which poses a big challenge to the state-of-the-art methods. Additionally, our online methods are fast as we perform neither conventional nor incremental principal component analysis (PCA). In contrast, conventional PCA [29] is an essential subproblem to numerically solve both RPCA and GFL problems. In these problems, each iteration involves computing PCA, which operates at a cost $\mathcal{O}(mn^2)$ and is due to SVD on a $m \times n$ matrix. We also recall that the state-of-the-art online, semi-online, or batch incremental algorithms, such as the Grassmannian robust adaptive subspace estimation (GRASTA) [27], recursive projected compressive sensing algorithm (ReProCS) [24, 25, 41], or incremental principal component pursuit (incPCP) [46, 44, 45], use either thin or partial PCA as well.

The need for simpler solvers for ℓ_1 regression. It is natural to ask: why do we need a new set of algorithms to solve the classical ℓ_1 regression problem when there are several well known solvers, for example, CVX [22, 21], ℓ_1 magic [47], and SparseLab 2.1-core [1]? It turns out that a high resolution video sequence (characterized by very large m) is computationally extremely expensive for the above mentioned classic solvers. Moreover, we do not need highly accurate solutions. Hence, simple and scalable methods are preferable to more involved and computationally demanding methods. The ℓ_1 magic software, for example, in our experiments took 126 minutes (on a computer with Intel i7 Processor and 16 GB memory) to estimate the background on the `Waving Tree` dataset with $A_2 \in \mathbb{R}^{19,200 \times 66}$. In contrast, our IRLS method took 0.59 seconds only for 66 frames.

3.1. Iteratively Reweighted Least Squares (IRLS)

In the past decade, IRLS has been used in various domains, ranging from reconstruction of sparse signals from underdetermined systems, to the low-rank and sparse matrix minimization problems in face clustering, motion segmentation, filter design, automatic target detection, to mention just a few applications [43, 11, 12, 13, 40, 32, 36]. We find that the IRLS algorithm is a good fit to solve (9). Also, each iteration of IRLS reduces to a single weighted ℓ_2 regression problem for an over determined system. To the best of our knowledge, we are the first to use IRLS to propose a background estimation model.

We now briefly describe IRLS for solving (9). First note that the cost function f in (9) can be written in the form

$$\phi(x) = \sum_{j=1}^{m} |q_j^\top x - b_j| = \sum_{j=1}^{m} \frac{(q_j^\top x - b_j)^2}{|q_j^\top x - b_j|}. \qquad (10)$$

For $x \in \mathbb{R}^m$ and $\delta > 0$ define a diagonal weight matrix via $W_\delta(x) := \mathrm{Diag}(1/\max\{|q_j^\top x - b_j|, \delta\})$. Given a current iterate x_k, we may fix the denominator in (10) by substituting x_k for x, which makes ϕ dependent on x via x appearing in the numerator only. The problem of minimizing the resulting function in x is a *weighted least squares problem.*

The normal equations for this problem have the form

$$Q^\top W_0(x_k) Q x = Q^\top W_0(x_k) b. \tag{11}$$

IRLS is obtained by setting x_{k+1} to be equal to the solution of (11). For stability purposes, however, we shall use weight matrices $W_\delta(x_k)$ for some threshold parameter $\delta > 0$ instead. This leads to the IRLS method:

$$\boxed{x_{k+1} = (Q^\top W_\delta(x_k) Q)^{-1} Q^\top W_\delta(x_k) b} \tag{12}$$

Osborne [40] and more recently [49] performed a comprehensive analysis of the performance of IRLS for ℓ_p minimization with $1 < p < 3$.

3.2. Homotopy Method

In this section we generalize the IRLS method (12) by introducing a *homotopy* [11] parameter $1 \leq p \leq 2$. We set $p_0 = 2$ and choose $x_0 \in \mathbb{R}^t$ (in our experiments, random initialization will do). Consider the function

$$\phi_p(x, y) := \sum_{j=1}^{m} \frac{(q_j^\top x - b_j)^2}{|q_j^\top y - b_j|^{2-p}}.$$

Note that $\phi_1(x, x)$ is identical to the ℓ_1 regression function ϕ appearing in (10). Given current iterate x_k, consider function $\phi_{p_k}(x, x_k)$. This is a weighted least squares function of x. Our homotopy method is defined by setting

$$x_{k+1} = \arg\min_x \phi_{p_k}(x, x_k),$$

and subsequently decreasing the homotopy parameter as $p_{k+1} = \max\{p_k \eta, 1\}$, where $0 < \eta < 1$ is a constant reduction factor.

As in the case of IRLS, the normal equations for the above problem have the form

$$Q^\top W_{0,p_k}(x_k) Q x = Q^\top W_{0,p_k}(x_k) b, \tag{13}$$

where $W_{\delta,p}(x) := \mathrm{Diag}(1/\max\{|q_j^\top x - b_j|^{2-p}, \delta\})$. The (stabilized) solution of (13) is given by

$$\boxed{x_{k+1} = (Q^\top W_{\delta,p_k}(x_k) Q)^{-1} Q^\top W_{0,p_k}(x_k) b} \tag{14}$$

As mentioned above, one step of the homotopy scheme (14) is identical to one step of IRLS (11) when $p_k = 1$. In practice, however, the homotopy method sometimes performs better (see Figures 1, 7, and Table 4).

3.3. Stochastic Subgradient Descent

In this section we propose the use of *two variants* of stochastic subgradient descent (SGD) to solve (9):

$$\min_{x \in \mathbb{R}^t} \phi(x) := \frac{1}{m} \sum_{j=1}^{m} \phi_j(x), \tag{15}$$

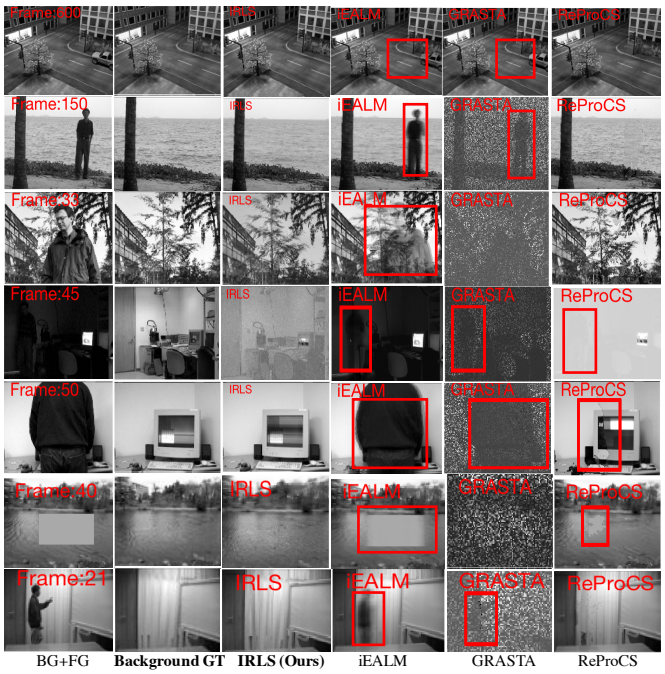


Figure 4: Background recovered on Stuttgart, Wallflower, and I2R dataset. Comparing with the ground truth (second column), IRLS recovers the best quality background.

where $\phi_j(x) := m|q_j^\top x - b_j|$. Functions ϕ_j are convex, but not differentiable. However, they are subdifferentiable. A classical result from convex analysis says that the subdifferential of a sum of convex functions is the sum of the subdifferentials. Therefore, the subddiferential $\partial\phi$ of ϕ is given by the formula $\partial\phi(x) = \frac{1}{m}\sum_{j=1}^{m} \partial\phi_j(x)$. In particular, if we choose $j \in [m]$ uniformly at random, and pick $g_j(x) \in \partial\phi_j(x)$, then $\mathrm{E}[g_j(x)] \in \partial\phi(x)$. That is, $g_j(x)$ is an unbiased estimator of a subgradient of ϕ at x.

A generic SGD method applied to (15) (or, equivalently, to (9)) has the form

$$\boxed{x_{k+1} = x_k - \eta_k g_i(x_k)} \tag{16}$$

An easy calculation using the chain rule for subdifferentials of convex functions gives the following formula for $\partial\phi_j(x) = m q_j \partial|q_j^\top x - b_j|$ (see, for instance, [38]):

$$\partial\phi_j(x) = \begin{cases} mq_j, & \text{if } q_j^\top x - b_j > 0 \\ -mq_j, & \text{if } q_j^\top x - b_j < 0. \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

When $q_j^\top x_k - b_j$ is nonzero, each iterate of SGD moves in the direction of either vector q_j or $-q_j$, with an appropriate stepsize. The initialization of the method (i.e., choice of $x_0 \in \mathbb{R}^t$ and the learning rate parameters η_k) plays an important role in the convergence of the method.

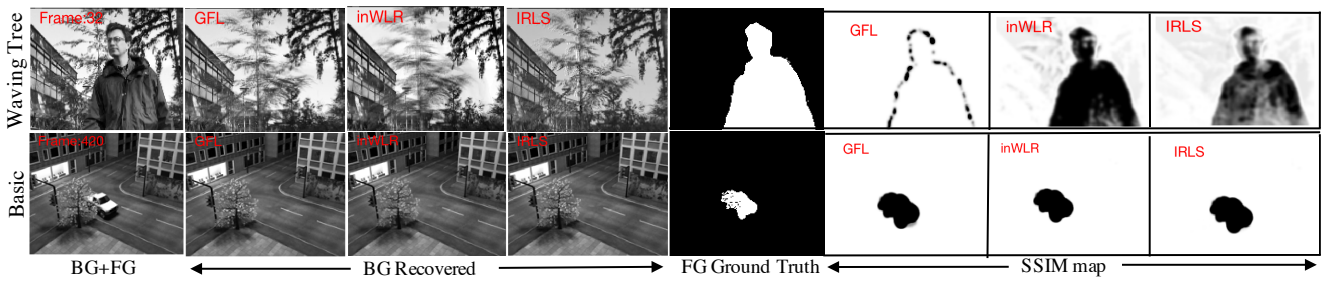


Figure 5: Qualitative and Quantitative comparison with supervised GFL and inWLR. GFL and IRLS construct better backgrounds on the `Waving Tree` video. On `Basic` video all the methods have similar performance. However, supervised GFL takes 117.11 seconds and 6.25 seconds on `Waving Tree` and `Basic` video, respectively, to process 1 frame; whereas inWLR takes 3.39 seconds and 17.83 seconds, respectively on those two sequences. In contrast, IRLS takes 0.59 seconds and 7.02 seconds, respectively and recovers the similar SSIM map.

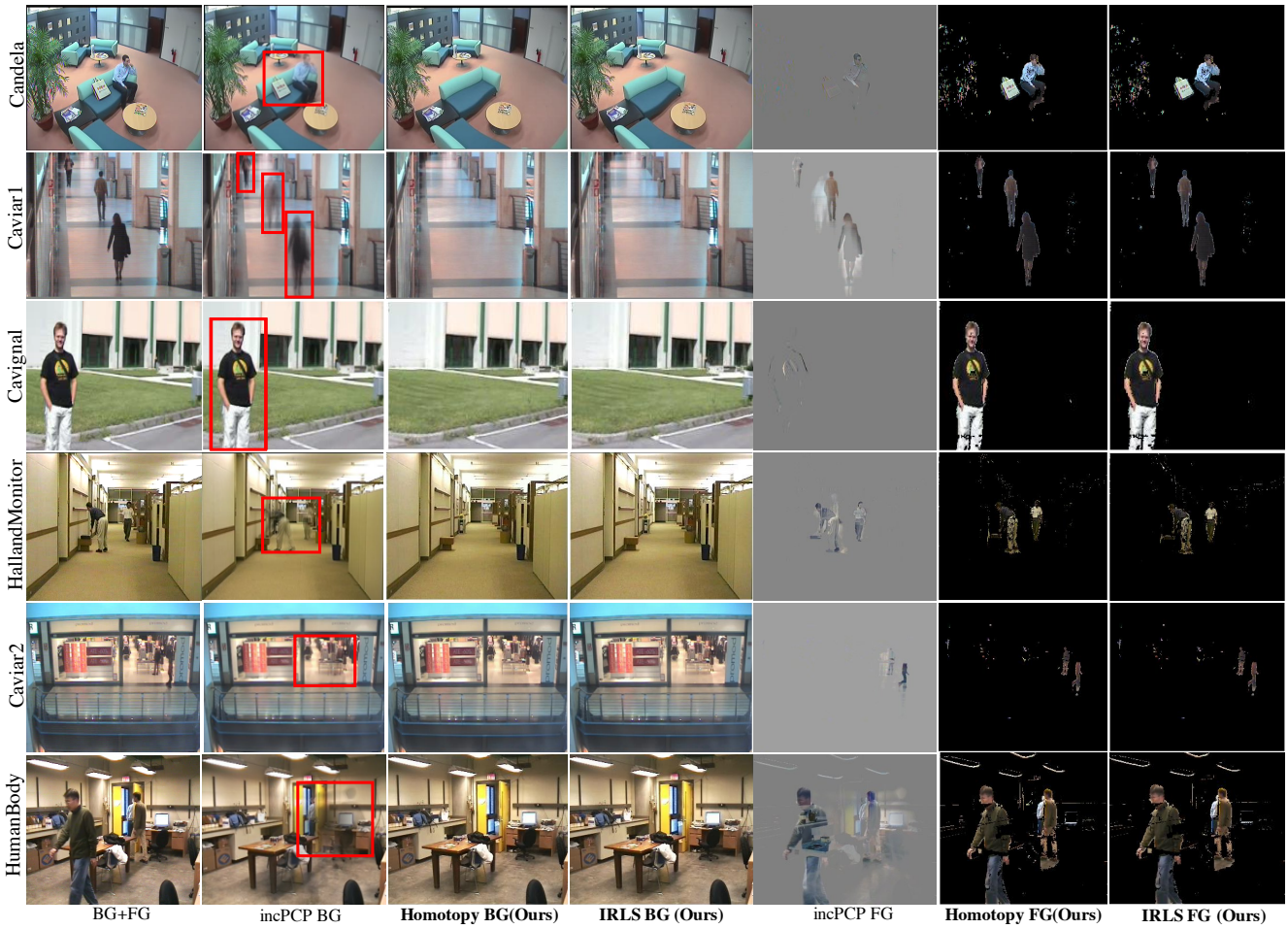


Figure 6: Background and foreground recovered by online methods on SBI dataset. The videos have static, semi-static foreground, newly added static foreground, shadows that already present in the background and newly created by moving foreground, and occlusion and disocclusion of static and dynamic foreground. For a comprehensive review of the dataset we refer the readers to [34].

We consider two variants of SGD depending on the choice of η_k and on the vector that we output. In SGD 1 we always normalize each stochastic subgradient, and multiply the resulting vector by $R/\sqrt{k}$, where k is the iteration counter, for some constant $R > 0$ which needs to be tuned. This method is a direct extension of the subgradient descent

Figure 7: Qualitative and Quantitative comparison on `Toscana-HD` video. Besides IRLS and Homotopy, the two best methods on `Toscana`, that is, Photomontage [3] and SOBS1 [33] have MSSIM 0.9616 and 0.9892 and CQM 50.2416 and 43.3002, respectively [7].

method	η_k	output
SGD 1	$\frac{R}{\sqrt{k}\|g_j(x_k)\|}$	x_k
SGD 2	$\frac{B}{\rho\sqrt{K}}$	$\hat{x}_K = \frac{1}{K}\sum_{k=0}^{K-1} x_k$

Table 1: Two variants of SGD.

method in [38]. The output is the last iterate. While we provide no theoretical guarantees for this method, it performs well in our experiments. On the other hand, SGD 2 is a more principled method. This arises as a special case of the SGD mehod described and analyzed in [48]. In this method, one needs to decide on the number of iterations K to be performed in advance. The method ultimately outputs the average of the iterates. The stepsize η_k is set to $B/\rho\sqrt{K}$, where $B > 0$ and $\rho > 0$ are parameters the value of which can be derived from the following iteration complexity result:

Theorem 1 ([48]). *Let x_* be a solution of (15) and let $B > 0$ be such that $\|x_*\| \leq B$. Further, assume that $\|g_j(x)\| \leq \rho$ for all $x \in \mathbb{R}^t$ and $j \in [m]$. If SGD 2 runs for K iterations with $\eta = \frac{B}{\rho\sqrt{T}}$, then $\mathrm{E}[\phi(\hat{x}_K)] - \phi(x_*) \leq \frac{B\rho}{\sqrt{K}}$, where $\hat{x}_K$ is given as in Table 1. Moreover, for any $\epsilon > 0$ to achieve $\mathrm{E}[\phi(\hat{x}_K)] - \phi(x_*) \leq \epsilon$ it suffices to run SGD 2 for K iterations where $K \geq \frac{B^2\rho^2}{\epsilon^2}$.*

4. Numerical Experiments

To validate the robustness of our proposed algorithms, we tested them on some challenging real world and synthetic video sequences containing occlusion, dynamic background, static, and semi-static foreground. For this purpose, we extensively use 19 gray scale and RGB videos from the Stuttgart, I2R, Wallflower, and the SBI dataset [10, 30, 34, 2, 50]. . We refer the readers to Table 2 to get an overall idea of the number of frames of each video sequence used, video type, and resolution.

For quantitative measure, we use the receiver operating characteristic (ROC) curve, recall and precision (RP) curve, the structural similarity index (SSIM), SSIM map [52], multi-scale structural similarity index (MSSSIM) [53], and color image quality measure (CQM) [7, 57]. Due to the availability of ground truth (GT) frames, we use the Stuttgart artificial dataset (has foreground GT) and the SBI dataset (have background GTs) to analyze the results quantitatively and qualitatively. To calculate the average computational time we ran each algorithm five times on the same dataset and compute the average. Throughout this section, the best and the 2^{nd} best results are colored with red and blue, respectively.

4.1. Comparison between our proposed algorithms

First we compare the performance of our proposed algorithms in batch mode on the `Basic` scenario. Figure 1 shows that all four algorithms are are very competitive and we note that IRLS has the least computational time. We ran each of IRLS and Homotopy method for five iterations, and SGD 1 and SGD 2 for 5000 iterations. IRLS takes 7.02 seconds, Homotopy takes 8.47 seconds, SGD 1 takes 17.81 seconds, and SGD 2 takes 17.67 seconds. We mention that the choice of R in SGD 1 and B and ρ in SGD 2 are problem specific. Due to computational efficiency, we compare IRLS ℓ_1 with other batch methods in the next section.

4.2. Comparison with RPCA, GFL, and other state-of-the-art methods

In this section we compare IRLS with other state-of-the-art batch background estimation methods, such as, iEALM [31] of RPCA, GRASTA, and ReProCS on the `Basic` scenario. Figure 12a shows that IRLS sweeps the maximum area under the ROC curve. Additionally, in Figure 3 IRLS has the best mean SSIM (MSSIM) among all other methods. Moreover, in batch mode, IRLS takes the least computational time.

Next in Figure 4 we present the background recovered by each method on Stuttgart, Wallflower, and I2R dataset. The video sequences have occlusion, dynamic background, and static foreground. IRLS can detect the static foreground and also robust to sudden illumination changes.

Finally, we compare our IRLS with the supervised GFL model of Xin *et al*. [56] and inWLR of Dutta *et al*. [18] (see Figure 5). For `Waving Tree` scenario, supervised GFL uses 200 training frames and it takes 117.11 seconds to compute the background and foreground from one training

Dataset	Video	No. of frames	Resolution
Stuttgart [10]	`Basic` (Grayscale)	600	144×176
	`Basic` (RGB-HD)	600	600×800
	`Lightswitch` (RGB-HD)	600	600×800
SBI [34]	`IBMTest2` (RGB)	91	320×240
	`Candela` (RGB)	351	352×288
	`Caviar1` (RGB)	610	384×288
	`Caviar2` (RGB)	461	384×288
	`Cavignal` (RGB)	258	200×136
	`HumanBody` (RGB)	741	320×240
	`HallandMonitor` (RGB)	296	352×240
	`Highway1` (RGB)	440	320×240
	`Highway2` (RGB)	500	320×240
	`Toscana`(RGB-HD)	6	600×800
Wallflower [50]	`Waving Tree`(Grayscale)	66	120×160
	`Camouflage` (Grayscale)	52	120×160
I2R/Li dataset [30]	`Meeting Room`(Grayscale)	1209	64×80
	`Watersurface` (Grayscale)	162	128×160
	`Lightswitch`(Grayscale)	1430	120×160
	`Lake`(Grayscale)	80	72×90

Table 2: Data used in this paper.

Algorithm	Abbreviation	Appearing in Experiment	Reference
Iterative Reweighted Laast Squares	IRLS	Figure 1–11,and Table4,5	This paper
Homotopy	Homotopy	Figure 1,6–11, and Table 4, 5	This paper
Stochastic Subgradient Descent 1	SGD 1	Figure 1 and Table 1	This paper
Stochastic Subgradient Descent 2	SGD 2	Figure 1 and Table 1	This paper
Inexact Augmented Lagrange Method of Multipliers	iEALM	Figure 12a-4	[31]
Supervised Generalized Fused Lasso	GFL	Figure 5, 8	[56]
Grassmannian Robust Adaptive Subspace Tracking Algorithm	GRASTA	Figure 12a–4	[27]
Recurssive Projected Compressive Sensing	ReProCS	Figure 12a–4	[24, 25, 41]
Incremental Weighted Low-Rank	inWLR	Figure 5	[18]
Incremental Principal Component Pursuit	incPCP	Figure 6–11, and Table 4, 5	[46, 44, 45]
Background estimated by weightless neural networks	BEWIS	Table 4	[23]
Independent Multimodal Background Subtraction Multi-Thread	IMBS-MT	Table 4	[5]
RSL2011	-	Table 4	[42]
Color Median	-	Table 4	[28]
Photomontage	-	Table 4	[3]
Self-Organizing Background Subtraction1	SOBS1	Table 4	[33]

Table 3: Algorithms compared in this paper.

frame and the ssim of the FG is 0.9996. inWLR does not use any training frames and takes 3.39 seconds to compute the background and foreground from the video sequence that consists of 66 frames and the MSSIM is 0.9592. In contrast, IRLS uses 15 training frames and takes 0.59 seconds to process the entire video with an MSSIM 0.9398. For `Basic` scenario, supervised GFL again uses 200 training frames and takes 6.25 seconds to process one training frame and the ssim of the FG is 0.9462. inWLR does not require any training frame and takes 17.83 seconds to process 600 frames in a batch-incremental mode and the MSSIM is 0.9463. In contrast, IRLS uses only 15 training frames and takes 7.02 seconds to process the entire video and the MSSIM is 0.9524.

4.3. Online implementation on RGB videos

In this section we show the robustness of two of our algorithms on RGB videos in online mode. Due to the space limitation we only provide results on IRLS and homotopy algorithm (these two methods were also the fastest in the batch mode). Primarily, we compare our results with incPCP and GFL [46, 44, 45, 56]. We should mention that besides incPCP, probabilistic robust matrix factorization (PRMF) [51] and RPCA bilinear projection (RPCA-BL) [35] has online extensions. However, PRMF uses the entire available data in its batch normalization step and there is no available implementation of online RPCA-BL. To the best of our knowledge incPCP is the only state-of-the-art online method which deals with HD RGB videos in full online mode. The incPCP code is downloaded from the author's

Video	SOBS1	RSL2011	IMBS-MT	BEWIS	Color Median	IRLS	Homotopy	incPCP
IBMTest2	**0.9954**	0.9303	0.9721	0.9602	0.9939	0.9950	**0.9953**	0.9670
Candela	0.9775	0.9916	0.9893	0.9852	0.9382	**0.9995**	**0.9992**	0.9412
Caviar1	0.9781	0.9947	0.9967	0.9813	0.9918	**0.9994**	**0.9993**	0.8649
Caviar2	0.9994	0.9962	0.9986	0.9994	0.9994	**0.9999**	**0.9998**	0.9935
Cavignal	0.9947	0.9973	0.9982	**0.9984**	0.7984	**0.9989**	0.9975	0.8312
HumanBody	0.9980	0.9959	0.9958	0.9866	0.9970	**0.9996**	**0.9990**	0.9360
HallandMonitor	0.9832	0.9377	0.9954	0.9626	0.9640	**0.9991**	**0.9992**	0.9355
Highway1	0.9968	0.9899	0.9939	0.9886	0.9924	**0.9980**	**0.9985**	0.8847
Highway2	0.9991	0.9907	0.9960	0.9942	0.9961	**0.9994**	**0.9997**	0.9819
Toscana	0.9616	0.0662	0.8903	0.8878	0.8707	**0.9853**	**0.9996**	0.8416
Average	0.9814	0.9491	0.9929	0.9745	0.9542	**0.9975**	**0.9987**	0.9177

Table 4: Comparison of average MSSSIM of the different methods on SBI dataset. Source: [2, 7, 34].

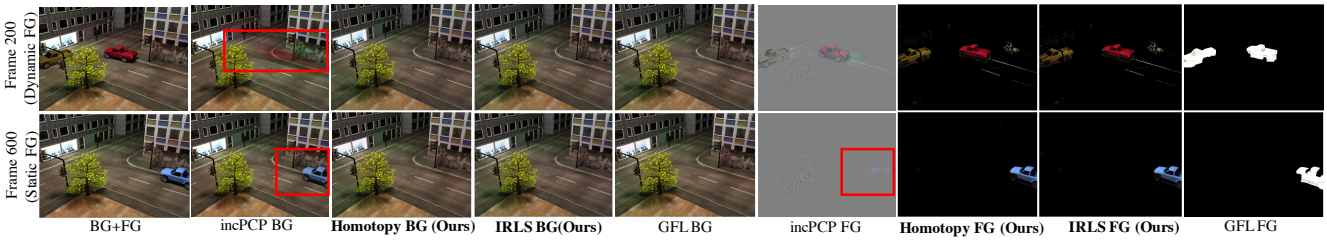


Figure 8: Qualitative comparison on Basic scenario HD scene. The **SSIM**s are (the 1st number indicates frame 200 and the 2nd number indicates frame 600): incPCP 0.021 and 0.0173, IRLS 0.9089 and 0.9731, homotopy 0.9327 and 0.9705, GFL 0.9705 and 0.9310. The **MSSSIM**s are: incPCP 0.6315 and 0.4208, IRLS 0.8777 and 0.9746, homotopy 0.9166 and 0.9725, GFL 0.9175 and 0.9645.

website[2]. As mentioned in the software package we use the standard PCP (fixed camera) mode for incPCP [46, 44] implementation.

Discussions. We use Basic-HD and the SBI dataset to provide extensive qualitative and quantitative comparison. The online mode of our algorithm only uses the available pure background frames to learn the basis Q for each color channel and then operate on each test frame in a complete online mode. Note that we only use 10 training frames and we strongly believe that one can use even less number of training frames to obtain almost the similar performance. Homotopy uses less iterations than IRLS to produce a comparable background and hence it is faster than IRLS in online mode. In Figure 6 and 9 we compare IRLS and homotopy against incPCP on the SBI dataset. Compare to the ghosting appearances in the incPCP backgrounds, our online methods construct a clean background for each video sequence. We also removed the static foreground, occluded foreground, and the foreground shadows. In Figure 7 and 8 we show our performance on HD video sequences. In addition to incPCP, we compared with supervised GFL on the Basic-HD (see Figure 8). Supervised GFL uses 200 training frames (the average processing time of the training frames is 7.31 seconds) and takes 431.78 seconds to process each test frame and produce a comparable quantitative result as online IRLS and homotopy. For computational time comparison with incPCP we refer to Table 5. Finally we provide the results of online IRLS and homotopy on one of the most challenging HD video sequences, that is, Lighswitch of the Stuttgart dataset. This scenario is a nighttime scenario and has varying illumination effects throughout the video sequence. Starting from frame 125 the illumination suddenly changes. Additionally, it has reflections, traffic light change, and movements of the tree leaves. We used 10 daytime pure background frames for training purpose and by using them we estimated the nighttime scene. As expected in Figure 10 both IRLS and homotopy perform pretty well with the changing illumination which can be verified from the pure Lightswitch BG frame (Figure 10 third column). Additionally, we compare our quantitative results against other state-of-the-art algorithms, such as, the adaptive neural background algorithm aka Self-Organizing Background Subtraction1 (SOBS1) [33], Photomontage [3], Color Median, RSL2011 [42], Independent Multimodal Background Subtraction Multi-Thread (IMBS-MT) [5], background estimated by weightless neural networks (BEWIS) [23] on SBI dataset. We refer Table 4 (Source: [7]) and Figure 7. Finally in Figure 11 we provide the mean CQM of the online methods on SBI dataset and Basic-HD video. In online mode, IRLS and Homotopy outperform incPCP in mean CQM and mean MSSSIM in each video.

[2]https://sites.google.com/a/istec.net/prodrig/Home/en/pubs/incpcp

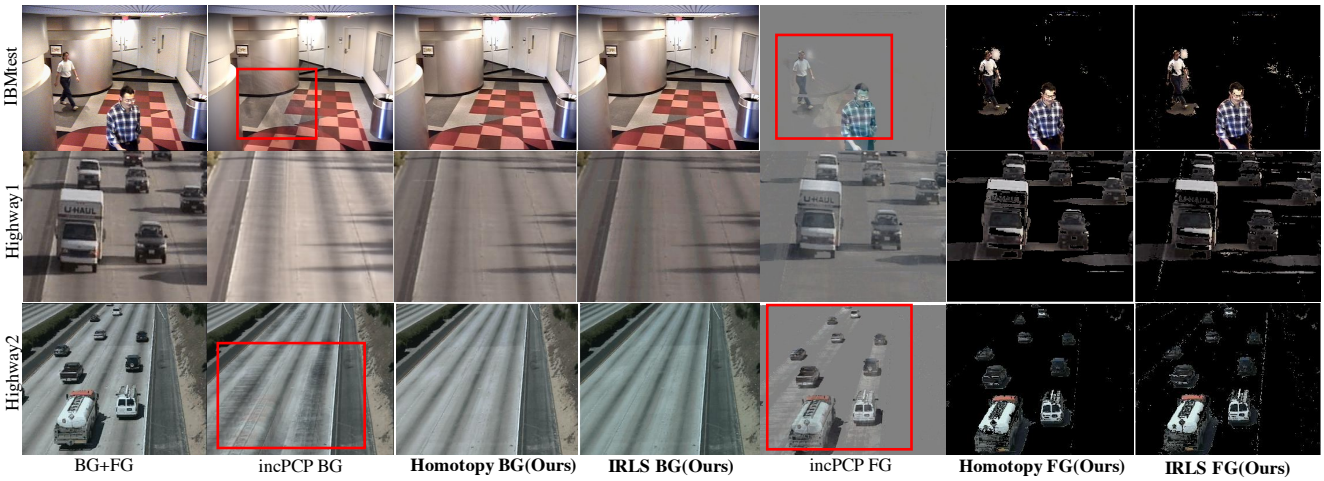


Figure 9: Background and foreground recovered by online methods on SBI dataset. The videos have shadows that already present in the background and newly created by moving foreground, occlusion and disocclusion of dynamic foreground.

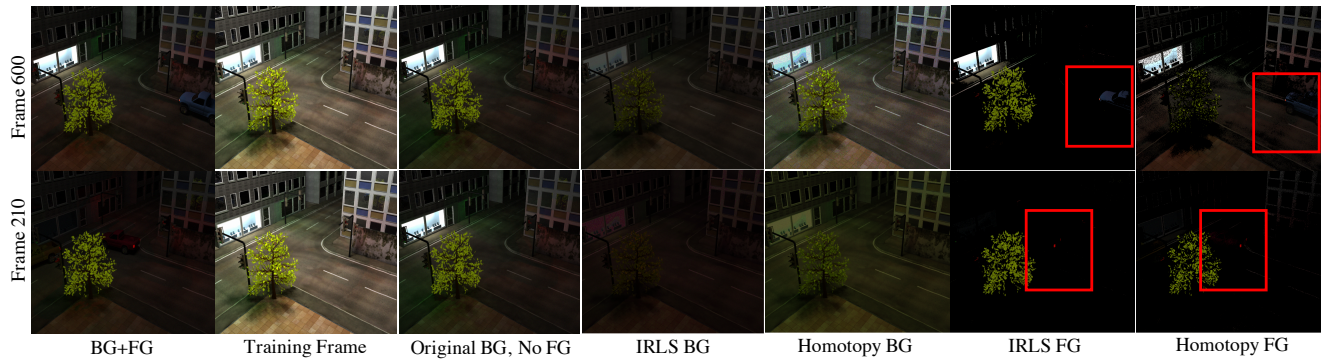


Figure 10: Background and foreground recovered by our proposed online methods on `Lightswitch` video. Both IRLS and homotopy captures the effect of change in illumination, irregular movements of the tree leaves, and reflections. Comparing with the No FG image both of our proposed method do pretty well.

5. Conclusion

We proposed a novel and fast model for supervised video background estimation. Moreover, it is robust to several background estimation challenges. We used the simple and well-known ℓ_1 regression technique and provided several online and batch background estimation methods that can process high resolution videos accurately. Our extensive qualitative and quantitative comparison on real and synthetic video sequences demonstrated that our supervised model outperforms the state-of-the-art online and batch methods in almost all cases.

6. Appendix 1: Historical Comments

We start by making a connection between the supervised GFL model proposed by Xin *et al*. [56] and the constrained low-rank approximation problem of Golub *et al*. [20].

6.1. Golub's constrained low-rank approximation problem

In 1987, Golub et al. [20] formulated the following constrained low-rank approximation problem: Given $A = [A_1 \ A_2] \in \mathbb{R}^{m\times n'}$ with $A_1 \in \mathbb{R}^{m\times r}$ and $A_2 \in \mathbb{R}^{m\times n}$, find $A_G = [\tilde{B}_1 \ \tilde{B}_2]$ such that, $\tilde{B}_1 \in \mathbb{R}^{m\times r}, \tilde{B}_2 \in \mathbb{R}^{m\times n}$, solve:

$$[\tilde{B}_1 \ \tilde{B}_2] = \arg \min_{\substack{B=[B_1 \ B_2] \\ B_1=A_1 \\ \text{rank}(B)\leq r}} \|A - B\|_F^2, \tag{18}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of matrices. Motivated by [20], Dutta *et al*. recently proposed more general weighted low-rank (WLR) approximation problems and showed their application in the background estimation problem [15, 16, 17].

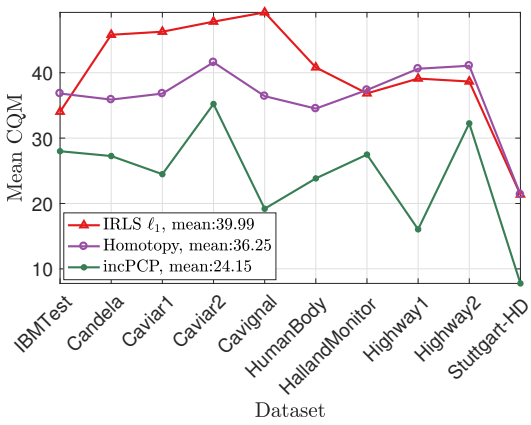


Figure 11: Mean CQM of online methods on SBI dataset and `Basic-HD` video. The higher the CQM value, the better is the recovered image.

Video (No. of frames)	IRLS	Homotopy	incPCP
`IBMTest2` (91)	37.28	**21.84**	22.45
`Candela` (351)	163.80	**133.6**	**72.15**
`Caviar1` (610)	279.99	**213.99**	**120.58**
`Caviar2` (461)	199.16	**158.1**	**85.68**
`Cavignal` (258)	71.26	**70.77**	**39**
`HumanBody` (741)	261.94	**227.25**	**134.83**
`HallandMonitor` (296)	116.86	**88.99**	**59.63**
`Highway1` (440)	155.84	**134.03**	**81.44**
`Highway2` (500)	181.85	**156.92**	**87**
`Basic-HD` (600)	599.06	**457.2464**	**382.41**
`Toscana-HD` (6)	7.73	**5.13**	**3.1**

Table 5: Computational time (in seconds) comparison for online methods.

Connection with (18)**.** Recall that the background estimation model the generalized fused lasso (GFL) proposed by Xin *et al*. [56] with the choice $f_{\text{rank}}(B) = \text{rank}(B)$ and $f_{\text{spar}}(F) = \lambda\|F\|_{\text{GFL}}$ can be written as:

$$\min_B \text{rank}(B) + \lambda\|A - B\|_{\text{GFL}}.$$

In this model, $\|\cdot\|_{GFL}$ is the "generalized fused lasso" norm. With the extra assumption that $\text{rank}(B) = \text{rank}(B_1)$ and by using the $\|\cdot\|_{\text{GFL}}$ norm, problem (2) is a constrained low rank approximation problem as in (18) and can be written as follows:

$$\min_{B=[B_1\ B_2]} \{\|A - B\|_{\text{GFL}} \text{ subject to } \text{rank}(B) \leq r, B_1 = A_1\}.$$

7. Appendix 2 : Augmented Lagrangian Method of Multipliers (ALM)

In this section, we demonstrate an additional background estimation method by using the decomposition model used in the main paper. This method was not described in the main paper. As mentioned in the **Further contributions** Section, we device a batch background estimation model (fifth method) by using the augmented Lagrangian method of multipliers (ALM).

7.1. The algorithm

The Augmented Lagrangian method of multipliers are one of the most popular class of algorithms in convex programming. In our setup, the proposed method does not provide an incremental algorithm. Instead it relies on fast batch processing of the video sequence. We can write (6) as an equality constrained problem by introducing the variable F_2 as follows:

$$\begin{aligned} &\min_{F_2,S} \|F_2\|_{\ell_1} \\ &\text{subject to } A_2 = QS + F_2. \end{aligned} \tag{19}$$

We now form the augmented Lagrangian of (19):

$$\begin{aligned} L(S, F_2, Y, \mu) = \|F_2\|_{\ell_1} + \langle Y, A_2 - QS - F_2\rangle \\ + \frac{\mu}{2}\|A_2 - QS - F_2\|_F^2, \end{aligned} \tag{20}$$

where $Y \in \mathbb{R}^{m\times n}$ is the Lagrange multiplier, $\langle Y, X\rangle = \text{Trace}(Y^\top X)$ is the trace inner product, and $\mu > 0$ is a penalty parameter. Completing the square and keeping only the relevant terms in (20), for the given iterates $\{S^{(k)}, F_2^{(k)}, Y^{(k)}, \mu_k\}$ we have

$$\begin{aligned} S^{(k+1)} &= \arg\min_S L(S, F_2^{(k)}, Y^{(k)}, \mu_k) \\ &= \arg\min_S \frac{\mu_k}{2}\left\|A_2 - QS - F_2^{(k)} + \frac{1}{\mu_k}Y^{(k)}\right\|_F^2, \\ F_2^{(k+1)} &= \arg\min_{F_2} L(S^{(k+1)}, F_2, Y^{(k)}, \mu_k) \\ &= \arg\min_{F_2} \|F_2\|_{\ell_1} + \frac{\mu_k}{2}\left\|A_2 - QS^{(k+1)} - F_2 + \frac{1}{\mu_k}Y^{(k)}\right\|_F^2. \end{aligned}$$

The solution to the first subproblem is obtained by setting the gradient of $L(S, F_2^{(k)}, Y^{(k)}, \mu_k)$ with respect to S to 0, and using the fact that $Q^\top Q = I$:

$$S^{(k+1)} = Q^\top\left(A_2 - F_2^{(k)} + \frac{1}{\mu_k}Y^{(k)}\right). \tag{21}$$

The second subproblem is the classic sparse recovery problem and its solution is given by

$$F_2^{(k+1)} = \mathcal{S}_{\frac{1}{\mu_k}}\left(A_2 - QS^{(k+1)} + \frac{1}{\mu_k}Y^{(k)}\right), \tag{22}$$

where $\mathcal{S}_{\frac{1}{\mu_k}}(\cdot)$ is the elementwise shrinkage function [26, 4]. We update Y_k and μ_k via:

$$\begin{cases} Y^{(k+1)} = Y^{(k)} + \mu_k(A_2 - QS^{(k+1)} - F_2^{(k+1)}) \\ \mu_{k+1} = \rho\mu_k \end{cases}, \tag{23}$$

for a fixed $\rho > 1$.

Algorithm 1: ALM

Input : $A = [A_1 \;\; A_2] \in \mathbb{R}^{m \times n'}$ (data matrix), threshold $\epsilon > 0, \rho > 1, \mu_0 > 0$;

Initialize: $A_1 = QR, Y^{(0)} = A_2/\|A_2\|_\infty, S^{(0)}, F_2^{(0)}$;

while *not converged* **do**

$S^{(k+1)} = Q^\top(A_2 - F_2^{(k)} + \frac{1}{\mu_k} Y^{(k)})$;

$F_2^{(k+1)} = \mathcal{S}_{\frac{1}{\mu_k}}(A_2 - QS^{(k+1)} + \frac{1}{\mu_k} Y^{(k)})$;

$Y^{(k+1)} = Y^{(k)} + \mu_k(A_2 - QS^{(k+1)} - F_2^{(k+1)})$;

$\mu_{k+1} = \rho\mu_k$;

$k = k + 1$;

end

Output : $S^{(k)}, F_2^{(k)}$

Dual problem. Next we formulate the Lagrangian dual of (6) to get an insight into the choice of the Lagrange multiplier Y. Using standard arguments, we obtain

$$\begin{aligned} \min_{F_2, S \,:\, A_2 = QS + F_2} \|F_2\|_{\ell_1} &= \min_{F_2, S} \sup_{Y} \|F_2\|_{\ell_1} \\ &\quad + \langle Y, A_2 - QS - F_2 \rangle \\ &\geq \sup_{Y} \min_{F_2, S} \|F_2\|_{\ell_1} \\ &\quad + \langle Y, A_2 - QS - F_2 \rangle \\ &= \sup_{Y \,:\, \|Y\|_\infty \leq 1,\, Q^\top Y = 0} \langle Y, A_2 \rangle. \end{aligned} \tag{24}$$

The last problem above is the dual of (6) . Clearly, the dual is a linear program. Note that the constraint $Q^\top Y$ dictates that the columns of Y be orthogonal to all columns of Q . That is, the columns of Y must be from the nullspace of Q. If we relax this constraint, the resulting problem has a simple closed form solution, namely

$$Y^{(0)} = A_2/\|A_2\|_\infty.$$

This is a good choice for the initial value of Y in Algorithm 1 .

7.2. Grassmannian robust adaptive subspace estimation (GRASTA)

Due to close connection with our ALM, we explain the Grassmannian robust adaptive subspace estimation (GRASTA) in this section. In 2012, He et al. [27] proposed GRASTA, a robust subspace tracking algorithm, and showed its application in background estimation problem. Unlike Robust PCA [31, 55], GRASTA is not a batch-video background estimation algorithm. GRASTA solves the background estimation problem in an incremental manner, considering one frame at a time. At each time step i, it observes a subsampled video frame $a_{i_{\Omega_s}}$. That is, each video frame $a_i \in \mathbb{R}^m$ is subsampled over the index set $\Omega_s \subset \{1, 2, \cdots, m\}$ to produce $a_{i_{\Omega_s}}$, where s is the subsample percentage. Similarly, denote the foreground as $F_2 = (f_1, \ldots, f_n)$. Therefore, $f_{i_{\Omega_s}} \in \mathbb{R}^{|\Omega_s|}$ is a vector whose entries are indexed by Ω_s. Considering each video frame $a_{i_{\Omega_s}}$ has a low rank (say, r) and sparse structure, GRASTA models the video frame as:

$$a_{i_{\Omega_s}} = U_{\Omega_s} x + f_{i_{\Omega_s}} + \epsilon_{\Omega_s},$$

where $U \in \mathbb{R}^{m \times r}$ be an orthonormal basis of the low-dimensional subspace, $x \in \mathbb{R}^r$ is a weight vector, and $\epsilon_{\Omega_s} \in \mathbb{R}^{|\Omega_s|}$ is a Gaussian noise vector. The matrix $U_{\Omega_s} \in \mathbb{R}^{|\Omega_s| \times r}$ results from choosing the rows of U corresponding to the index set Ω_s. With the notations above, at each time step i, GRASTA solves the following optimization problem: For a given orthonormal basis $U_{\Omega_s} \in \mathbb{R}^{|\Omega_s| \times r}$ solve

$$\min_x \|U_{\Omega_s} x - a_{i_{\Omega_s}}\|_{\ell_1}. \tag{25}$$

Problem (25) is the classic least absolute deviations problem similar to (7) and can be rewritten as:

$$\begin{aligned} &\min_{f_{i_{\Omega_s}}} \|f_{i_{\Omega_s}}\|_{\ell_1} \\ &\text{subject to } U_{\Omega_s} x + f_{i_{\Omega_s}} - a_{i_{\Omega_s}} = 0. \end{aligned} \tag{26}$$

Problem (26) can be solved by the use of the augmented Lagrangian multiplier method (ALM) [9]. In GRASTA, after updating x and $f_{i_{\Omega_s}}$, one has to update the orthonormal basis U_{Ω_s} as well. The rank one U_{Ω_s} update step is done first by finding a gradient of the augmented Lagrange dual of (26), and then by using the classic gradient descent algorithm. In summary, at each time step i, given a $U^{(i)} \in \mathbb{R}^{m \times r}$ and $\Omega_s \subset \{1, 2, \cdots, m\}$, GRASTA finds x and $f_{i_{\Omega_s}}$ via (26) and then updates $U_{\Omega_s}^{(i+1)}$. This process continues until the video frames are exhausted.

Comparison between ALM and GRASTA. 1. At each step of GRASTA, the background and the sparse foreground are given as $U_{\Omega_s} x$ and $a_{i_{\Omega_s}} - U_{\Omega_s} x$, respectively and then one has to update the basis U_{Ω_s}. In contrast, (19) solves a supervised batch video background estimation problem. In our model, once we obtain the basis set from the QR decomposition of the background matrix A_1, we do not update the basis further. 2. GRASTA lacks a convergence analysis which is harder to obtain as the objective function (25) in their set-up is only convex in each component. [27]. Our objective function in (6) and in (20) are convex and therefore allow us to propose a thorough convergence analysis for ALM.

7.3. Cost of One Iteration

We discuss the complexity of one iteration of Algorithm 1 when A_1 is of full rank, that is, $\mathrm{rank}(A_1) = r$. The complexity of the QR decomposition at the initialization

step is $\mathcal{O}(2mr^2 - \frac{2}{3}r^3)$. Because $r \leq r_{\max}$, the maximum number of available training frames, the above cost can be controlled by the user. Next, the complexity of one iteration of Algorithm 1 is $\mathcal{O}(mnr)$. In contrast, the cost of each iteration of GRASTA is $\mathcal{O}(|\Omega_s|r^3 + Kr|\Omega_s| + mr^2)$, where K is the number of inner iterations and $|\Omega_s|$ is the cardinality of the index set $\Omega_s \subset \{1, 2, \cdots, m\}$ from which each video frame $a_i \in \mathbb{R}^m$ is subsampled at a percentage s (see Section 7.2).

7.4. Stopping Criteria

Define $L_k := L(S^{(k)}, F_2^{(k)}, Y^{(k-1)}, \mu_{k-1})$. With the notations above, for a given $\epsilon > 0$, Algorithm 1 converges if $\|A_2 - QS^{(k)} - F_2^{(k)}\|_F / \|A_2\|_F < \epsilon$, or $|L_k - L_{k-1}| < \epsilon$, or if the maximum iteration is reached.

7.5. Remarks on the Behaviour of ALM

In this section, we propose the convergence of Algorithm 1.

Lemma 1. *The sequence $\{Y^{(k)}\}$ is bounded.*

Proof. By the optimality condition of $F_2^{(k+1)}$ we have,

$$0 \in \partial_{F_2} L(S^{(k+1)}, F_2, Y^{(k)}, \mu_k).$$

Therefore,

$$0 \in \partial \|F_2^{(k+1)}\|_{\ell_1} - \mu_k (A_2 - QS^{(k+1)} - F_2^{(k+1)} + \frac{1}{\mu_k} Y^{(k)}),$$

which implies $Y^{(k+1)} \in \partial \|F_2^{(k+1)}\|_{\ell_1}$. By using Theorem 4 in [31] (see also [54]), we conclude that the sequence $\{Y^{(k)}\}$ is bounded by the dual norm of $\|\cdot\|_{\ell_1}$, that is, the $\|\cdot\|_\infty$ norm. □

Theorem 2. *There is a constant γ such that*

$$\|A_2 - QS^{(k)} - F_2^{(k)}\| \leq \frac{\gamma}{\mu_k}, \;\; k = 1, 2, \cdots.$$

Proof. By using (23) we have

$$A_2 - QS^{(k)} - F_2^{(k)} = \frac{1}{\mu_{k-1}} (Y^{(k)} - Y^{(k-1)}).$$

The result follows by applying Lemma 1.

□

Theorem 3. *The sequence $\{L_k\}$ is bounded above and*

$$L_{k+1} - L_k \leq O\left(\frac{1}{\mu_{k-1}}\right), \;\; k = 1, 2, \cdots.$$

Proof. We have,

$$\begin{aligned}
L_{k+1} &= L(S^{(k+1)}, F_2^{(k+1)}, Y^{(k)}, \mu_k) \\
&\leq L(S^{(k+1)}, F_2^{(k)}, Y^{(k)}, \mu_k) \\
&\leq L(S^{(k)}, F_2^{(k)}, Y^{(k)}, \mu_k) \\
&= \|F_2^{(k)}\|_{\ell_1} + \langle Y^{(k)}, A_2 - QS^{(k)} - F_2^{(k)} \rangle \\
&\quad + \frac{\mu_k}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2 \\
&= \|F_2^{(k)}\|_{\ell_1} + \langle Y^{(k-1)}, A_2 - QS^{(k)} - F_2^{(k)} \rangle \\
&\quad + \frac{\mu_{k-1}}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2 \\
&\quad + \langle Y^{(k)} - Y^{(k-1)}, A_2 - QS^{(k)} - F_2^{(k)} \rangle \\
&\quad + \frac{\mu_k - \mu_{k-1}}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2 \\
&\overset{\text{(using (23))}}{=} L_k + \mu_{k-1} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2 \\
&\quad + \frac{\mu_k - \mu_{k-1}}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2 \\
&= L_k + \frac{\mu_k + \mu_{k-1}}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2.
\end{aligned}$$

Therefore,

$$L_{k+1} - L_k \leq \frac{\mu_k + \mu_{k-1}}{2} \|A_2 - QS^{(k)} - F_2^{(k)}\|_F^2, \;\; k = 1, 2, \cdots.$$

By using (23) we have for $k = 1, 2, \cdots$

$$L_{k+1} - L_k \leq \frac{\mu_k + \mu_{k-1}}{\mu_{k-1}^2} \|Y^{(k)} - Y^{(k-1)}\|_F^2 = \frac{1 + \rho}{\mu_{k-1}} \|Y^{(k)} - Y^{(k-1)}\|_F^2.$$

Next by using the boundedness of $\{Y^{(k)}\}$ we find

$$L_{k+1} - L_k \;\; \leq \;\; O\left(\frac{1}{\mu_{k-1}}\right), \;\; k = 1, 2, \cdots,$$

which is what we set out to prove. □

Theorem 4. *We have*

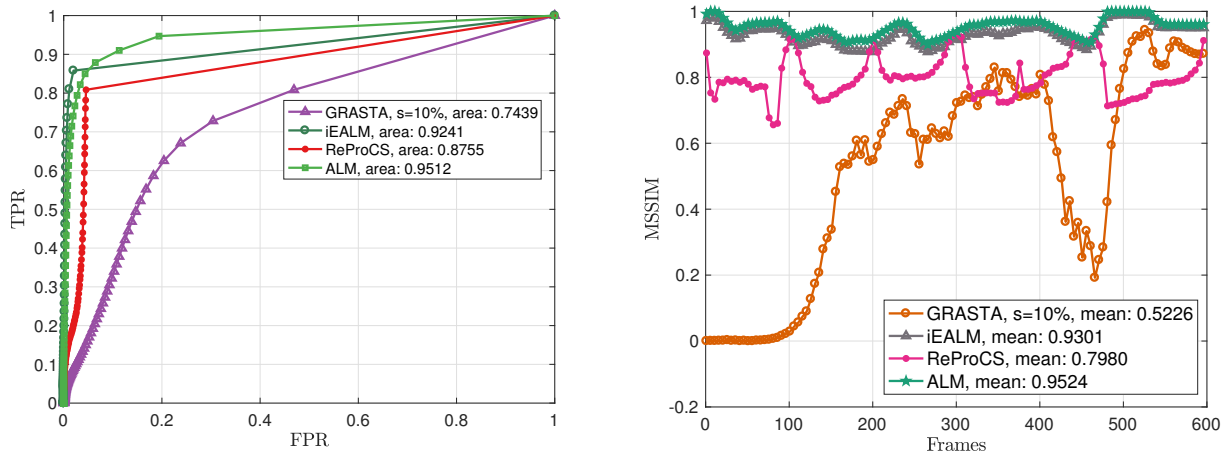
$$f^* - \|F_2^{(k)}\|_{\ell_1} \leq O\left(\frac{1}{\mu_k}\right),$$

where $f^* = \min_{A_2 = QS + F_2} \|F_2\|_{\ell_1}$.

Proof. By using the triangle inequality we have

$$\begin{aligned}
\|F_2^{(k)}\|_{\ell_1} &\geq \|A_2 - QS^{(k)}\|_{\ell_1} \\
&\quad - \|A_2 - QS^{(k)} - F_2^{(k)}\|_{\ell_1} \\
&\overset{\text{(using (23))}}{\geq} f^* - \frac{1}{\mu_{k-1}} \|Y^{(k)} - Y^{(k-1)}\|_{\ell_1}.
\end{aligned} \tag{27}$$

The result follows by applying boundedness of the multipliers $Y^{(k)}$. □



(a) (b)

Figure 12: (a) ROC curve to compare between ALM, iEALM, GRASTA, and ReProCS on `Basic` video, frame size 144×176. (b) Comparison of Mean SSIM (MSSIM) of ALM, iEALM, GRASTA, and ReProCS on `Basic` video. ALM has the best MSSIM. To process 600 frames each of size 144×176, iEALM takes 164.03 seconds, GRASTA takes 20.25 seconds, ReProCS takes 14.20 seconds, and ALM takes 13.13 seconds.

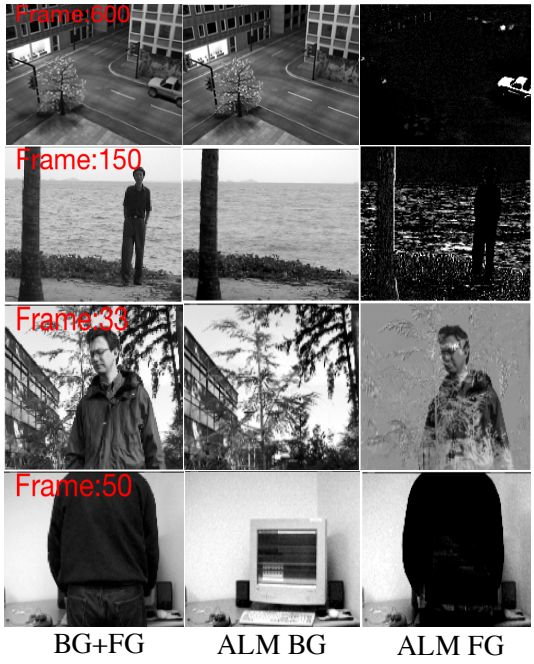


Figure 13: Background and foreground recovered by ALM. The videos have static foreground and dynamic background.

8. Smooth Optimization of ℓ_1 Regression with Parallel Coordinate Descent Methods [19]

Imagine a situation when one processes a very low-resolution video sequence with a huge number of available training frames. That is, when there are more training frames r than the number of pixels m, the method used in [19] to solve (8) for each i could be more effective. In this scenario we propose to solve each ℓ_1 regression problem in (8) by using the parallel coordinate descent methods on their smooth variants [19]. Note that each $f_i(s_i)$ is a non-smooth continuous convex function on a compact set E_1. By using Nesterov's smoothing technique [37] one can find a smooth approximation $f_i^{\mu}(s_i)$ of $f_i(s_i)$ for any $\mu > 0$. Fercoq et al. [19] minimized $f_i^{\mu}(s_i)$ to approximately solve the original ℓ_1 regression problem that contains $f_i(s_i)$.

9. Additional numerical experiments demonstrating the effectiveness of ALM

To demonstrate the robustness of the ALM in batch mode, we compare ALM with other state-of-the-art batch background estimation methods, such as, iEALM [31] of RPCA, GRASTA [27], and ReProCS [24] on the `Basic` scenario. We use 15 training frames for ALM. Figure 12a shows that ALM covers the maximum area under the ROC curve. Additionally, in Figure 12b, our ALM has the best mean SSIM (MSSIM) among all other methods. Moreover, in batch mode, ALM takes the least computational time. The background and foreground recovered by ALM in batch mode also shows its effectiveness in supervised background estimation (see Figure 13)

References

[1] https://sparselab.stanford.edu/.

[2] http://sbmi2015.na.icar.cnr.it/SBIdataset.html.

[3] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. *ACM Transactions on Graphics*, 23:294–302, 2004.

[4] T. Boas, A. Dutta, X. Li, K. P. Mercier, and E. Niderman. Shrinkage function and its applications in matrix approximation. *Electronic Journal of Linear Algebra*, 32:163–171, 2017.

[5] D. D. Bolci, A. Pennisi, and L. Iocchi. Parallel multimodal background modeling. *Pattern Recognition Letters*, 96:45–54, 2017.

[6] T. Bouwmans. Traditional and recent approaches in background modeling for foreground detection: An overview. *Computer Science Review*, 11–12:31 – 66, 2014.

[7] T. Bouwmans, L. Maddalena, and A. Petrosino. Scene background initialization: A taxonomy. *Pattern Recognition Letters*, 2017.

[8] T. Bouwmans, A. Sobral, S. Javed, S. K. Jung, and E.-H. Zahzah. Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset. *Computer Science Review*, 2016.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[10] S. Brutzer, B. Höferlin, and G. Heidemann. Evaluation of background subtraction techniques for video surveillance. *IEEE Computer Vision and Pattern Recognition*, pages 1568–1575, 2012.

[11] C. S. Burrus, J. A. Barreto, and I. W. Selesnick. Iterative reweighted least-squares design of fir filters. *IEEE Transaction on Signal Processing*, 42(11):2926–2936, 1994.

[12] E.J. Candès, M.B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

[13] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Gunturk. Iteratively reweighted least squares minimization for sparse recovery. *Communications on Pure and Applied Mathematics*, 63:1–38, 2010.

[14] A. Dutta, B. Gong, X. Li, and M. Shah. Weighted singular value thresholding and its application to background estimation, 2017. arXiv:1707.00133.

[15] A. Dutta and X. Li. A fast algorithm for a weighted low rank approximation. In *15th IAPR International Conference on Machine Vision Applications (MVA)*, pages 93–96, 2017.

[16] A. Dutta and X. Li. On a problem of weighted low-rank approximation of matrices. *SIAM Journal on Matrix Analysis and Applications*, 38(2):530–553, 2017.

[17] A. Dutta and X. Li. Weighted low rank approximation for background estimation problems. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1853–1861, 2017.

[18] A. Dutta, X. Li, and P. Richtárik. A batch-incremental video background estimation model using weighted low-rank approximation of matrices. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1835–1843, 2017.

[19] O. Fercoq and P. Richtárik. Smooth minimization of nonsmooth functions with parallel coordinate descent methods. *arXiv:1309.5885*, 2013.

[20] G. H. Golub, A. Hoffman, and G. W. Stewart. A generalization of the Eckart-Young-Mirsky matrix approximation theorem. *Linear Algebra and its Applications*, 88(89):317–327, 1987.

[21] M. Grant and S. Boyd. Graph implementations for nonsmooth convex programs. In *Recent Advances in Learning and Control*, Lecture Notes in Control and Information Sciences, pages 95–110. Springer-Verlag Limited, 2008.

[22] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `http://cvxr.com/cvx`, 2014.

[23] M. D. Gregorio and M. Giordano. Background estimation by weightless neural networks. *Pattern Recognition Letters*, 96:55–65, 2017.

[24] H. Guo, C. Qiu, and N. Vaswani. An online algorithm for separating sparse and low-dimensional signal sequences from their sum. *IEEE Transactions on Signal Processing*, 62(16):4284–4297, 2014.

[25] H. Guo, C. Qiu, and N. Vaswani. Practical REPROCS for seperating sparse and low-dimensional signal sequences from their sum-part 1. In *IEEE International Conference on Acoustic, Speech and Signal Processing*, pages 4161–4165, 2014.

[26] E.T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: methodology and convergence. *SIAM Journal on Optimization*, 19:1107–1130, 2008.

[27] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. *IEEE Computer Vision and Pattern Recognition*, pages 1937–1944, 2012.

[28] T. Huang, G. Yang, and G. Tang. A fast two-dimensional median filtering algorithm. *IEEE Trans. Acoustic, Speech, Signal Processing*, 27(1):13–18, 1979.

[29] I. T. Jolliffee. Principal component analysis, 2002. Second edition.

[30] L. Li, W. Huang, I.-H. Gu, and Q. Tian. Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing*, 13(11):1459–1472, 2004.

[31] Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, 2010. arXiv1009.5055.

[32] C. Lu, Z. Lin, and S. Yan. Smoothed low rank and sparse matrix recovery by iteratively reweighted least squares minimization. *IEEE Transactions on Image Processing*, 24(2):646–654, 2015.

[33] L. Maddalena and A. Petrosino. The SOBS algorithm: What are the limits? In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–26, 2012.

[34] L. Maddalena and A. Petrosino. Towards benchmarking scene background initialization. In *New Trends in Image Analysis and Processing – ICIAP 2015 Workshops*, pages 469–476, 2015.

[35] G. Mateos and G. Giannakis. Robust PCA as bilinear decomposition with outlier-sparsity regularization. *IEEE Transaction on Signal Processing*, 60(10):5176–5190, 2012.

[36] B. Millikan, A. Dutta, N. Rahnavard, Q. Sun, and H. Foroosh. Initialized iterative reweighted least squares for automatic target recognition. In *Proceedings of IEEE Military Communications Conference*, pages 506–510, 2015.

[37] Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programmming*, 103(1):127–152, 2005.

[38] Y. Nesterov. Introductory lectures on convex optimization: A basic course, 2014. First edition.

[39] N. Oliver, B. Rosario, and A. Pentland. A Bayesian computer vision system for modeling human interactions. In *International Conference on Computer Vision Systems*, pages 255–272, 1999.

[40] M. Osborne. Finite algorithms in optimization and data analysis, 1985. John Wiley & Sons, Inc.

[41] C. Qiu and N. Vaswani. Support predicted modified-CS for recursive robust principal components pursuit. In *IEEE International Symposium on Information Theory*, pages 668–672, 2011.

[42] V. Reddy, C. Sanderson, and B. C. Lovell. A low-complexity algorithm for static background estimation from cluttered image sequences in surveillance contexts. *Journal of Image Video Process.*, pages 1:1–1:14, 2011.

[43] P. Richtárik. *Some algorithms for large-scale convex and linear minimization in relative scale*. PhD thesis, Cornell University, 2007.

[44] P. Rodriguez and B. Wohlberg. A matlab implementation of a fast incremental principal component pursuit algorithm for video background modeling. In *IEEE International Conference on Image Processing*, pages 3414–3416, 2014.

[45] P. Rodriguez and B. Wohlberg. Translational and rotational jitter invariant incremental principal component pursuit for video background modeling. In *2015 IEEE International Conference on Image Processing*, pages 537–541, 2015.

[46] P. Rodriguez and B. Wohlberg. Incremental principal component pursuit for video background modeling. *Journal of Mathematical Imaging and Vision*, 55(1):1–18, 2016.

[47] J. Romberg. https://statweb.stanford.edu/ candes/l1magic/.

[48] S. Shalev-Shwartz and S. Ben-David. Understanding machine learning: From theory to algorithms, 2014. Cambridge University Press.

[49] J. Sigl. Nonlinear residual minimization by iteratively reweighted least squares, 2015. arXiv:1504.06815.

[50] K Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintainance. *Seventh International Conference on Computer Vision*, pages 255–261, 1999.

[51] N. Wang, T. Yao, J. Wang, and D.-Y. Yeung. A probabilistic approach to robust matrix factorization. In *Proceedings of 12th European Conference on Computer Vision*, pages 126–139, 2012.

[52] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transaction on Image Processing*, 13(4):600–612, 2004.

[53] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *37th IEEE Asilomar Conference on Signals, Systems, and Computers*, pages 1398–1402, 2003.

[54] G.A. Watson. Characterization of the subdifferential of some matrix norms. *Linear Algebra and its Applications*, 170:33–45, 1992.

[55] J. Wright, Y. Peng, Y. Ma, A. Ganseh, and S. Rao. Robust principal component analysis: exact recovery of corrputed low-rank matrices by convex optimization. *Proceedings of 22nd Advances in Neural Information Processing systems*, pages 2080–2088, 2009.

[56] B. Xin, Y. Tian, Y. Wang, and W. Gao. Background subtraction via generalized fused Lasso foreground modeling. *IEEE Computer Vision and Pattern Recognition*, pages 4676–4684, 2015.

[57] Y. Yalman and I. Erturk. A new color image quality measure based on yuv transformation and psnr for human vision system. *Turkish Journal of Electrical Engineering and Computer Sciences*, 21(2):603–612, 2013.