A Batch-Incremental Video Background Estimation Model using Weighted Low-Rank Approximation of Matrices

Aritra Dutta
King Abdullah University of Science and Technology (KAUST)
Thuwal 23955-6900, Kingdom of Saudi Arabia
aritra.dutta@kaust.edu.sa

Xin Li
University of Central Florida, USA
4000 Central Florida Blvd, Orlando, FL-32816
xin.li@ucf.edu

Peter Richtárik
King Abdullah University of Science and Technology (KAUST), KSA
University of Edinburgh, Scotland
peter.richtarik@kaust.edu.sa

Abstract

Principal component pursuit (PCP) is a state-of-the-art approach for background estimation problems. Due to their higher computational cost, PCP algorithms, such as robust principal component analysis (RPCA) and its variants, are not feasible in processing high definition videos. To avoid the curse of dimensionality in those algorithms, several methods have been proposed to solve the background estimation problem in an incremental manner. We propose a batch-incremental background estimation model using a special weighted low-rank approximation of matrices. Through experiments with real and synthetic video sequences, we demonstrate that our method is superior to the state-of-the-art background estimation algorithms such as GRASTA, ReProCS, incPCP, and GFL.

1. Introduction

Background estimation and moving object detection is an important step in many computer vision systems and video-surveillance applications. In the past decade, one of the prevalent approaches used for background estimation is to treat it as a low-rank and sparse matrix decomposition problem [1, 2, 24]. Oliver *et al*. [19] showed that when the camera motion is small, the background is not expected to change much throughout the video frames and they assumed it to be low-rank. The seminal work of Lin *et al*., Wright *et al*., and Candès *et al*. [5, 18, 27], which is referred as robust principal component analysis (RPCA), solves the problem of background estimation and moving object detection in a single framework.

Given a sequence of n video frames with each frame $\mathbf{a}_i \in \mathbb{R}^m$ being vectorized, let the data matrix $A = (\mathbf{a}_1, \mathbf{a}_2, \cdots, \mathbf{a}_n) \in \mathbb{R}^{m\times n}$ be the concatenation of all the video frames. The foreground is usually sparse if its size is relatively small compared to the frame size [5, 18, 27]. Therefore, it is natural to consider a matrix decomposition problem by writing A as the sum of its background and foreground:

$$A = B + F,$$

where $B, F \in \mathbb{R}^{m\times n}$ are the low-rank background and sparse foreground matrices, respectively. RPCA solves:

$$\min_B \|A - B\|_{\ell_1} + \lambda\|B\|_*, \tag{1}$$

where $\|\cdot\|_{\ell_1}$ and $\|\cdot\|_*$ denote the ℓ_1 norm and the nuclear norm (sum of the singular values) of matrices, respetively.

Consider a situation when a few, say k, principal directions are already specified and one wants to find a rank r approximation of the data, where $k \leq r$. In 1987, Golub *et al*. [11] formulated the following constrained low-rank approximation problem (to be referred as GHS from now on) to address this situation: Given $A = (A_1\ A_2) \in \mathbb{R}^{m\times n}$ with $A_1 \in \mathbb{R}^{m\times k}$ and $A_2 \in \mathbb{R}^{m\times(n-k)}$, find $A_G = (\tilde{B}_1\ \tilde{B}_2)$ such that

$$(\tilde{B}_1\ \tilde{B}_2) = \arg \min_{\substack{B=(B_1\ B_2)\\ B_1=A_1\\ \mathrm{rank}(B)\le r}} \|A-B\|_F^2, \tag{2}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of matrices. That is, Golub *et al*. required a few columns, A_1, of A be preserved when looking for a low rank approximation of $(A_1\ \ A_2)$. When $A_1 = \emptyset$, we are back to the standard problem of low-rank approximation: find $\tilde{B}$ such that

$$\tilde{B} = \arg \min_{\substack{B\\ \mathrm{rank}(B)\le r}} \|A-B\|_F^2. \tag{3}$$

As it is well known, this problem is equivalent to principal component analysis (PCA) [15] and has a closed form solution using the singular value decomposition (SVD) of A: if $A = PDQ^t$ is a SVD of A with unitary matrices P, Q and diagonal matrix D (of non-ascending diagonal entries), then the solution to (3) is given by $\tilde{B} = H_r(A) := PD_rQ^t$, where D_r is a diagonal matrix obtained from D by only keeping the r largest entries and replacing the rest by 0. The operator H_r is referred to as the hard thresholding operator. Using the thresholding operator, GHS problem (2) has a closed form solution as the following theorem explains.

Theorem 1 *[11] Assume* $\mathrm{rank}(A_1) = k$ *and* $r \ge k$*, the solution* $\tilde{B}_2$ *in (2) is given by*

$$\tilde{B}_2 = P_{A_1}(A_2) + H_{r-k}\left(P^{\perp}_{A_1}(A_2)\right), \tag{4}$$

where P_{A_1} *and* $P^{\perp}_{A_1}$ *are the projection operators to the column space of* A_1 *and its orthogonal complement, respectively.*

Assuming some pure background frames are known, GHS can be applied by using these background frames as the first block matrix A_1. Along a similar line, recently, Xin *et al*. [28] proposed a *supervised* learning model called generalized fused Lasso (GFL) which solves:

$$\min_{\substack{B\\ B=(B_1\ B_2)\\ B_1=A_1}} \mathrm{rank}(B) + \|A-B\|_{gfl}, \tag{5}$$

where $\|\cdot\|_{gfl}$ denotes a norm that is a combination of the ℓ_1 norm and a local spatial total variation norm (to encourage connectivity of the foreground). To solve GFL problem (5), Xin *et al*. [28] further specialized the above model by requiring $\mathrm{rank}(B) = \mathrm{rank}(A_1)$. Note that, with this specialization, problem (5) can be viewed as a constrained low-rank approximation problem as in GHS problem (2) and can be formulated as follows:

$$\min_{\substack{B=(B_1\ B_2)\\ \mathrm{rank}(B)\le r\\ B_1=A_1}} \|A-B\|_{gfl}. \tag{6}$$

1.1. Incremental Methods

Conventional PCA [15] is an essential tool in numerically solving both RPCA and GFL problems. PCA operates at a cost of $\min\{\mathcal{O}(m^2n), \mathcal{O}(mn^2)\}$ which is due to the SVD of an $m \times n$ data matrix. For RPCA algorithms, the space complexity of an SVD computation is approximately $\mathcal{O}((m+n)r)$, where r is the rank of the low-rank approximation matrix in each iteration, which is increasing. For a high resolution video sequence characterized by very large m, this results in high computational cost and memory usage for the RPCA and GFL algorithms. For example, the accelerated proximal gradient (APG) algorithm runs out of memory to process 600 video frames each of size 300×400 on a computer with 3.1 GHz Intel Core i7-4770S processor and 8GB memory. In the past few decades, incremental PCA (IPCA) was developed for machine learning applications to reduce the computational complexity of performing PCA on a huge data set. The idea is to produce an efficient SVD calculation of an augmented matrix of the form $[A\ \ \tilde{A}]$ using the SVD of A, where $A \in \mathbb{R}^{m\times n}$ is the original matrix and $\tilde{A}$ contains r newly added columns [29]. Similar to the IPCA, several methods have been proposed to solve the background estimation problem in an incremental manner [10, 17]. In 2012, He *et al*. [14] proposed the Grassmannian robust adaptive subspace estimation (GRASTA), a robust subspace tracking algorithm, and showed its application in background estimation problems. More recently, Guo *et al*. [12] proposed another online algorithm for separating sparse and low dimensional subspace. Given an initial sequence of training background video frames, Guo *et al*. devised a recursive projected compressive sensing algorithm (ReProCS) for background estimation (see also [13, 20]). Following a modified framework of the conventional RPCA problem, Rodriguez *et al*. [21] formulate the incremental principal component pursuit (incPCP) algorithm which processes one frame at a time in an incremental fashion and uses only a few frames for initialization of the prior (see also [22, 23]). To the best of our knowledge, these are the state-of-the-art incremental background estimation models.

1.2. Contributions

In this paper, we propose an adaptive batch-incremental model for background estimation. The strength of our model lies in finding the background frame indexes in a robust and incremental manner to process the entire video sequence. Unlike the models described previously, we do not require any training frames. The model we use allows us to use the background information from previous batch in a natural way.

Before describing our main contribution, let us take a pause here and revisit the idea of Golub *et al*. Inspired by (2) and motivated by applications in which A_1 may contain

noise, it makes more sense if we require $\|A_1 - B_1\|_F$ small instead of asking for $B_1 = A_1$ as in (2). This leads Dutta *et al*. [7, 8, 9] to consider the following more general weighted low-rank (WLR) approximation problem:

$$\min_{\substack{X=(X_1\ X_2)\\ \mathrm{rank}(X)\le r}} \|\left((A_1\ A_2) - (X_1\ X_2)\right) \odot W\|_F^2, \tag{7}$$

where $W \in \mathbb{R}^{m\times n}$ is a matrix with non-negative entires and $\odot$ denotes the Hadamard product. Using $W = (W_1\ \mathbb{1})$ in [7], the model (7) was applied to solve background estimation problems. Here we propose a batch-incremental background estimation model using the WLR algorithm of Dutta *et al*. to gain robustness. Similar to the ℓ_1 norm used in conventional and the incremental methods, the use of a weighted Frobenius norm makes WLR robust to the outliers for background estimation problems [7, 9]. Our batch method is fast and can deal with high quality video sequences similar to incPCP and ReProCS. Some conventional algorithms, for example, supervised GFL or ReProCS, require an initial training sequence which does not contain any foreground object. Our experimental results on both synthetic and real video sequences show that unlike the supervised GFL and ReProCS, our model does not require a prior instead, it can estimate its own prior robustly from the entire data. We believe the adaptive nature of the algorithm is suitable for real time high-definition video surveillance and for panning motion of the camera where the background is slowly evolving.

Algorithm 1: WLR Algorithm

Input : $A = (A_1\ \ A_2) \in \mathbb{R}^{m\times n}$ (the given matrix), $W = (W_1\ \ \mathbb{1}) \in \mathbb{R}^{m\times n}$ (the weight), threshold $\epsilon > 0$;
Initialize: $(X_1)_0, C_0, B_0, D_0$;
while *not converged* **do**
  $E_p = A_1 \odot W_1 \odot W_1 + (A_2 - B_pD_p)C_p^T$;
  for $i = 1 : m$ **do**
    $(X_1(i,:))_{p+1} = (E(i,:))_p(\mathrm{diag}(W_1^2(i,1)\ W_1^2(i,2)\cdots W_1^2(i,k)) + C_pC_p^T)^{-1}$;
  end
  $C_{p+1} = ((X_1)_{p+1}^T(X_1)_{p+1})^{-1}(X_1)_{p+1}^T(A_2 - B_pD_p)$;
  $B_{p+1} = (A_2 - (X_1)_{p+1}C_{p+1})D_p^T(D_pD_p^T)^{-1}$;
  $D_{p+1} = (B_{p+1}^TB_{p+1})^{-1}B_{p+1}^T(A_2 - (X_1)_{p+1}C_{p+1})$;
  $p = p + 1$;
end
Output : $(X_1)_{p+1}, (X_1)_{p+1}C_{p+1} + B_{p+1}D_{p+1}$.

Algorithm 2: Incremental Background Estimation using WLR (inWLR)

Input : p, $A = (A^{(1)}\ \ A^{(2)}\dots A^{(p)}) \in \mathbb{R}^{m\times n}$, $\tau > 0$ (for SVT), $\alpha, \beta > 0$ (for weights), threshold $\epsilon > 0$, k_{max}, $i_r \in \mathbb{N}$;
Run SVT on $A^{(1)}$ with parameter τ to obtain: $A^{(1)} = B_{In}^{(1)} + F_{In}^{(1)}$;
Initialize the background block by $B = B_{In}^{(1)}$ and $A^{(0)} = A^{(1)}$;
for $j = 1 : p$ **do**
  Identify the indices S of at most k_{max} columns of $A^{(j-1)}$ that are closest to background using B and $F = A^{(j-1)} - B$;
  Set $k = \#(S), r = k + i_r$;
  Set the first block: $\tilde{A}_1 = (A^{(j-1)}(:,i))_{m\times k}$ with $i \in S$;
  Define $W = (W_1\ \mathbb{1})$ with $W_1 \in \mathbb{R}^{m\times k}$ where $(W_1)_{ij}$ are randomly chosen from $[\alpha, \beta]$;
  Apply **Algorithm 1** on $\tilde{A}^{(j)} = (\tilde{A}_1\ A^{(j)})$ using threshold ϵ and weight W to obtain its low rank component $\tilde{B}^{(j)}$ and define $\tilde{F}^{(j)} = \tilde{A}^{(j)} - \tilde{B}^{(j)}$;
  Take the sub-matrix of $\tilde{B}^{(j)}$ corresponding to the $A^{(j)}$ block such that $A^{(j)} = B^{(j)} + F^{(j)}$;
  Update the background block: $B = \tilde{B}^{(j)}$;
end
Output : $B = (B^{(1)}, B^{(2)}, ..., B^{(p)})$.

1.3. The WLR algorithm

We now give a brief overview of the WLR algorithm proposed by Dutta *et al*. [8, 9]. Let $\mathrm{rank}(X_1) = k$. Then any X_2 such that $\mathrm{rank}(X_1\ \ X_2) \le r$ can be given in the form

$$X_2 = X_1C + BD,$$

for some matrices $B \in \mathbb{R}^{m\times(r-k)}$, $D \in \mathbb{R}^{(r-k)\times(n-k)}$, and $C \in \mathbb{R}^{k\times(n-k)}$. Therefore, problem (7) with $W = (W_1\ \mathbb{1})$ of compatible block partition is reduced to:

$$\min_{X_1,C,B,D} \|(A_1 - X_1) \odot W_1\|_F^2 + \|A_2 - X_1C - BD\|_F^2. \tag{8}$$

The complexity of one iteration of Algorithm 1 is $\mathcal{O}(mk^3 + mnr)$ [8].

2. An incremental model using WLR

In this section, we propose an incremental weighted low-rank approximation (inWLR) algorithm for background estimation based on WLR (see Algorithm 2 and Figure 1).

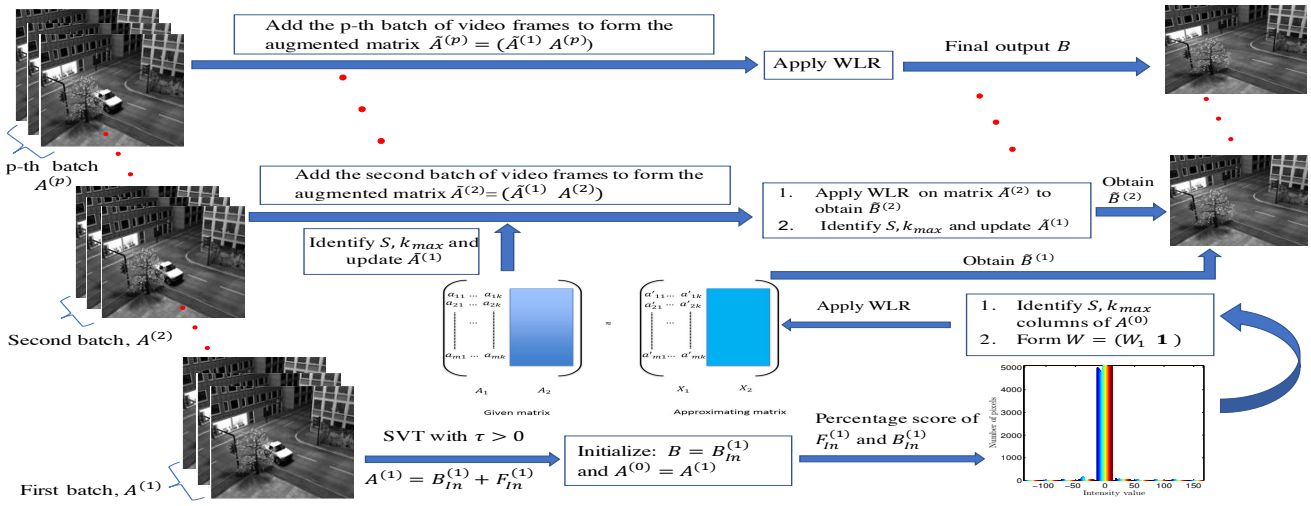


Figure 1: A flowchart for WLR inspired background estimation model proposed in Algorithm 2.

Our algorithm takes the full advantage of WLR in which a prior knowledge of the background space can be used as an additional constraint to obtain the low rank (thus the background) estimation of the data matrix A. Indeed, we start by partitioning the original video sequence into p batches: $A = (A^{(1)}\ \ A^{(2)} \dots A^{(p)})$, where the batch sizes do not need to be equal. Instead of working on the entire video sequence, the algorithm incrementally works through each batch. To initialize, the algorithm performs a coarse estimation of the possible background frame indices of $A^{(1)}$: we run the classic singular value thresholding (SVT) of Cai *et al.* [4] on $A^{(1)}$ to obtain a low rank component (containing the estimations of background frames) $B_{In}^{(1)}$ and let $F_{In}^{(1)} = A^{(1)} - B_{In}^{(1)}$ be the estimation of the foreground matrix (Step 2). From the above, we obtain the initialization for B and $A^{(0)}$ (Step 3). Then we go through each batch $A^{(j)}$ using the estimates of the background from the previous batch as prior for the WLR algorithm to get the background $\tilde{B}^{(j)}$ (Step 9). The identification of the "best background frames" is obtained by a modified version of the percentage score model by Dutta *et al.* [6] to determine the indices of frames that contain the least information of the foreground (Step 5). This allows us to estimate k, r, and the first block $\tilde{A}_1$ which contains the background prior knowledge (Steps 6-7). Weight matrix $W = (W_1\ \mathbb{1})$ is chosen by randomly picking entries of the first block W_1 from an interval $[\alpha, \beta]$ using an uniform distribution, where $\beta > \alpha > 0$ are large (Step 8). To understand the effect of using a large weight in W_1 we refer the reader to [7, 8]. Finally, we collect background information for next iteration (Steps 10-11). Note that the number of columns of the weight matrix W_1 is k which is controlled by bound k_{max} so that the column size of $\tilde{A}^{(j)}$ is not growing with j. The output of the algorithm is the background estimations for each batch

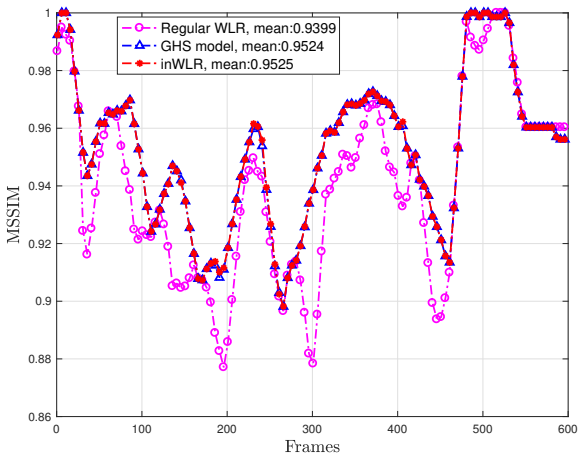


Figure 2: Comparison of MSSIM of WLR acting on all frames, inWLR, and GHS inspired background estimation model with frame size $[144, 176]$ and $p = 6$.

collected in a single matrix B. When the camera motion is small, updating the first block matrix $\tilde{A}_1$ (Step 7) has trivial impact on the model since it is not changing much. However, when the camera is panning and the background is continuously evolving, this could be proven very robust as new frames are entering in the video.

2.1. Complexity analysis

Now, we analyze the complexity of Algorithm 2 for equal batch size. Primarily, the cost of the SVT algorithm in Step 2 is only $\mathcal{O}(\frac{mn^2}{p^2})$. Next, in Step 9, the complexity of implementing Algorithm 1 is $\mathcal{O}(mk^3 + \frac{mnr}{p})$. Note that r and k are linearly related and $k \leq k_{max}$. Once we obtain

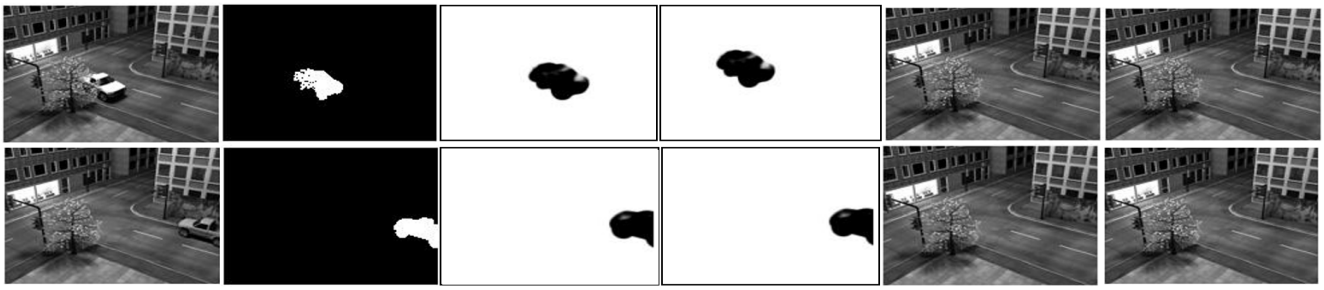


Figure 3: SSIM map of inWLR and GHS inspired background estimation model, frame size $[144, 176]$, and $p = 6$. Top to bottom: Frame 420 with dynamic foreground, frame 600 with static foreground. Left to right: Original, ground truth, inWLR SSIM, GHS SSIM, inWLR background, and GHS background. SSIM index of the methods are 0.95027 and 0.96152, respectively.

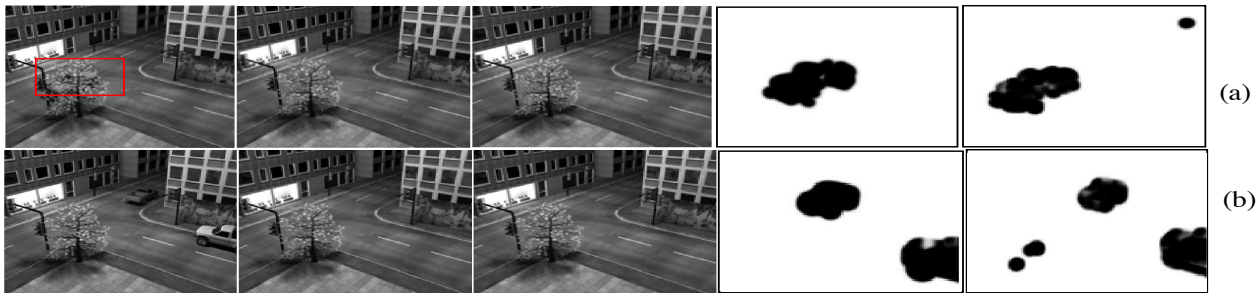


Figure 4: *Basic* scenario frame: (a) 50, (b)100. Left to right: Original, inWLR background, GFL background, inWLR SSIM, and GFL SSIM. The MSSIM of inWLR on two frames are 0.9595 and 0.9457, and that of GFL are 0.9534 and 0.9443, respectively.

a refined estimate of the background frame indices S as in Step 5, and, form an augmented matrix by adding the next batch of video frames, a very natural question in proposing our WLR inspired Algorithm 2 is: why do we use Algorithm 1 in each incremental step (Step 9) of Algorithm 2 instead of using a closed form solution (4) of GHS? We give the following justification: the estimated background frames $\tilde{A}_1$ are not necessarily exact background, only estimations of background. So, GHS inspired model may be forced to follow the wrong data while inWLR allows enough flexibility to find the best fit to the background subspace. This is confirmed by our numerical experiments (see Section 3.1 and Figure 2). Thus, to analyze the entire sequence in p batches, the complexity of Algorithm 2 is approximately $\mathcal{O}(m(k^3p + nr))$. Note that, the complexity of Algorithm 2 is dependent on the partition p of the original data matrix. Our numerical experiments suggest for video frames of varying sizes, the choice of p plays an important role and is empirically determined.

Unlike Algorithm 2, if Algorithm 1 is used on the entire data set, and if the number of possible background frame indices is k', then the complexity is $\mathcal{O}(mk'^3 + mnk')$. When k' grows with n and becomes much bigger than k_{max} in order to achieve competitive performance, we see that Algorithm 1 tends to slow down with higher overhead than Algorithm 2 does (see Table 1).

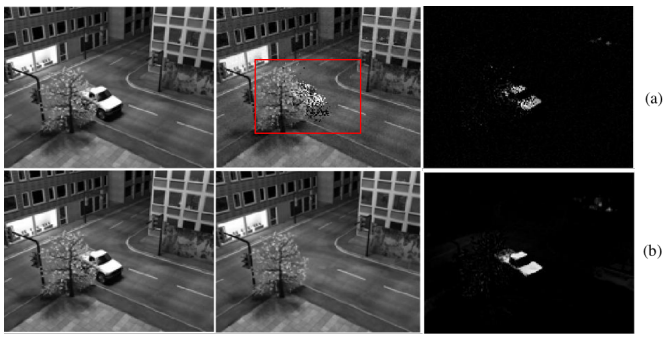


Figure 5: *Basic* scenario frame 420: (a) GRASTA, (b) inWLR. Left to right: Original, background, and foreground. GRASTA with subsample rate 10% recovers a fragmentary foreground and degraded background.

3. Qualitative and quantitative analysis

Due to the availability of ground truth frames for each foreground mask, we use 600 frames of the *Basic* scenario

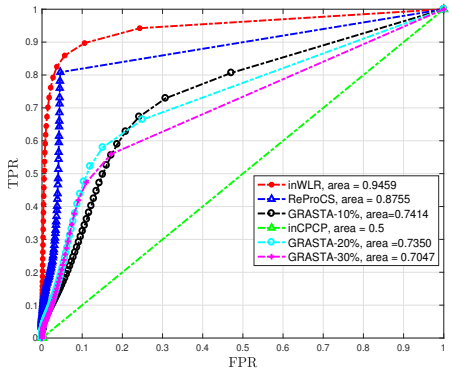


Figure 6: ROC curves on Stuttgart *Basic* scenario to compare between GRASTA, inWLR, incPCP, and ReProCS.

of the Stuttgart artificial video sequence [3] for quantitative and qualitative comparisons. To capture an unified comparison against each method, we resize the video frames to [144,176] and for inWLR set $p = 6$, that is, we add a batch of 100 new video frames in every iteration until all frames are exhausted.

3.1. Comparison with GHS

Since the *Basic* scenario has no noise, once we estimate the background frames, GHS can be used as a baseline method in comparing the effectiveness of Algorithm 2. To demonstrate the benefit of using an iterative process as in Algorithm 1, we first compare the performance of Algorithm 2 against the GHS inspired models. We also compare regular WLR acting on all 600 frames with the parameters specified in [9]. The structural similarity index (MSSIM) is used to quantitatively evaluate the overall image quality as it mostly agrees with the human visual perception [26]. To calculate the MSSIM of each recovered foreground video frame, we consider a 11×11 Gaussian window with standard deviation ($\sigma = 1.5$). We perceive the information how the high-intensity regions of the image are coming through the noise, and consequently, we pay much less attention to the low-intensity regions. We remove the noisy components from the foreground recovered by inWLR, F, by using a threshold ϵ_1 (calculated implicitly in Step 5 of Algorithm 2 to choose the background frames, see [6]), such that we set the components below ϵ_1 in F to 0. The average computation time of inWLR is approximately in the range 17.829035 seconds to 19.5755 seconds in processing 600 frames each of size 144×176. On the other hand, the GHS inspired model and WLR take approximately 273.8382 and 64.5 seconds, respectively. The MSSIM presented in Figure 2, indicates that inWLR and GHS inspired model produce same result with inWLR being more time efficient than GHS. Next in Figure 3, the SSIM map of two sample video frames of the *Basic* scenario show both methods recover the similar quality background and foreground frames. Figure 2 shows that to work on a high resolution video, inWLR is more accurate than GHS and WLR.

3.2. Comparison with GFL

We compare the performance of inWLR with the GFL model of Xin *et al.* [28][1]. For both models we use 200 frames of the *Basic* sequence, each frame resized to $[144, 176]$. From the background recovered and the SSIM map in Figures 4 and 11, it is clear that both methods are very competitive. However, it is worth mentioning that inWLR is extraordinarily time efficient compare to the GFL model.

3.3. Comparison with other state-of-the-art models

In this section, we compare the performance of inWLR against other incremental background estimation models such as, GRASTA, incPCP, and ReProCS on 600 frames of the *Basic* scenario of the Stuttgart sequence. For quantitative measure we use the receiver operating characteristic (ROC) curve, the recall and precision (RP) curve, and the MSSIM. For ROC curve and RP curve, we use a uniform threshold vector $\text{linspace}(0, 255, 100)$ to compare pixel-wise predictive analysis between each recovered foreground frame and the corresponding ground truth frame.

3.3.1 Comparison with GRASTA [14]

At each time step i, GRASTA solves the following optimization problem: For a given orthonormal basis U_{Ω_s} solve

$$\min_{x} \|U_{\Omega_s} x - A_{\Omega_s}(:, i)\|_{\ell_1}, \tag{9}$$

where each video frame $A(:, i) \in \mathbb{R}^n$ is subsampled over the index set $\Omega_s \subset \{1, 2, \cdots, n\}$ following the model: $A_{\Omega_s}(:, i) = U_{\Omega_s} x + F_{\Omega_s}(:, i) + \epsilon_{\Omega_s}$, such that, $x \in \mathbb{R}^{|\Omega_s|}$ is a weight vector and ϵ_{Ω_s} is a Gaussian noise vector of same size. After updating x, one has to update U_{Ω_s}. We set the subsample percentage s to 0%, 10%, 20%, and 30% respectively, estimated rank 60, and keep the other parameters same as in [14]. The GRSTA code is obtained from author's website.[2] Note that, for a lower estimated rank GRASTA does not perform well. Referring the qualitative result in Figure 5, we only provide the ROC curve and RP curve to compare GRASTA with different subsamples s and inWLR (see Figure 6 and 8a). The ROC curves and RP curves clearly show the superior performance of inWLR on the Stuttgart *Basic* scenario.

3.3.2 Comparison with ReProCS [12]

ReProCS is a two stage algorithm. In the first stage, given a sequence of training background frames, say t, the al-

[1] http://idm.pku.edu.cn/staff/wangyizhou/

[2] https://sites.google.com/site/hejunzz/grasta

Figure 7: *Basic* scenario frame 123. Left to right: Original, inWLR background, ReProCS background, inWLR foreground, ReProCS foreground, and ground truth. Both methods recover similar quality background, however, ReProCS foreground has more false positives than inWLR.

Dataset	ReProCS	GRASTA	inWLR	incPCP	WLR	GHS
Basic	15.8122	22.39	17.829035	58.4132	64.0485	273.8382
Fountain	-	-	3.709931	-	7.135779	4.327475
Waving Tree	4.548824	-	3.3873	-	13.751490	42.302412

Table 1: Computational time comparison. All experiments were performed on a computer with 2.7 GHz Intel Core i7 processor and 16 GB memory. The best and the 2^{nd} best results are colored with red and blue, respectively. For frame numbers, frame size, and p for inWLR see Section 3 and 4.

gorithm finds an approximate basis which is ideally of low-rank. After estimating the initial low-rank subspace, in the second stage, the algorithm recursively estimates F_{t+1}, B_{t+1}, and the subspace in which B_{t+1} lies. We use 200 background frames of the *Basic* sequence for initialization of ReProCS. Figure 7 shows both methods recover similar quality background. However, ReProCS foreground contains more false positives than inWLR foreground. The ROC curve, RP curve, and MSSIM in Figure 6, 8a, and 8b comply with our claim quantitatively for the *Basic* sequence. Though the average computation time for ReProCS is 15.644460 seconds which is better than inWLR.

3.3.3 Comparison with incPCP [21]

incPCP follows a modified framework of PCP but with the assumption that the partial rank r SVD of first $k-1$ background frames B_{k-1} is known and using them A_{k-1} can be written as $A_{k-1} = B_{k-1} + F_{k-1}$. Therefore for a new video frame $A(:,k)$ one can solve the optimization problem as follows:

$$\min_{\substack{B_k, F_k \\ \text{rank}(B_k) \le r}} \|B_k + F_k - A_k\|_F^2 + \lambda \|F_k\|_{\ell_1},$$

where $A_k = [A_{k-1} \;\; A(:,k)]$ and $B_k = [U_r \Sigma_r V_r^T \;\; B(:,k)]$ such that $U_r \Sigma_r V_r^T$ is a partial SVD of B_{k-1}. According to [21], the initialization step can be performed incrementally. For the Stuttgart sequence, the algorithm uses the first video frame for initialization. The incPCP code is downloaded from author's website[3]. From the MSSIM presented in Figure 8c and the background recovered by both methods in Figure 9, it seems that both methods perform equally well on the *Basic* scenario. However, when the foreground is static (as in frames 551-600 of the Stuttgart sequence), the ℓ_1 norm in incPCP is unable to capture the foreground object, resulting the presence of the static car as a part of the background (see Figure 10). On the other hand, our inWLR successfully detects the static foreground.

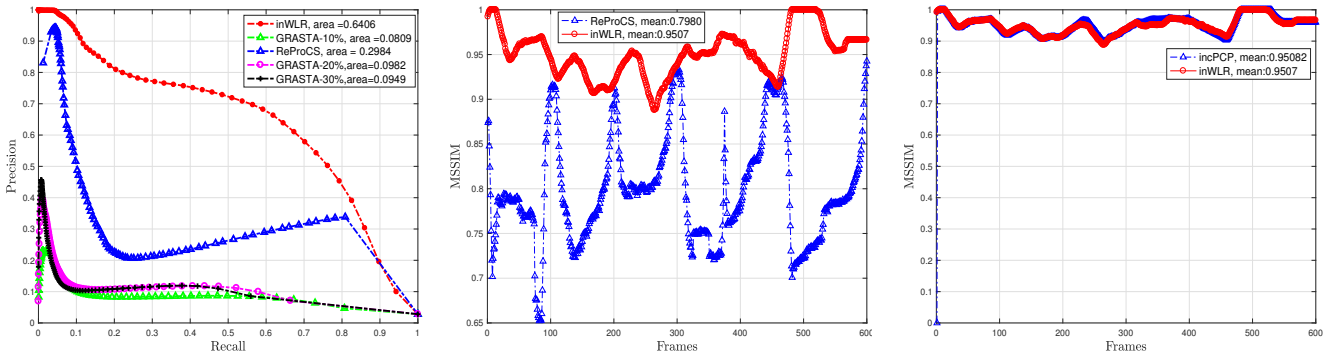
[3] https://sites.google.com/a/istec.net/prodrig/Home/en/pubs/incpcp

4. Results on real world sequences

In this section, we demonstrate the performance of inWLR on five challenging real world video sequences [16, 25], containing occlusion, dynamic background, and static foreground. In Figure 11 we compare inWLR against GFL and ReProCS on 60 frames of *Waving Tree* sequence. ReProCS and GFL use 220 and 200 pure background frames respectively as training data. In Figure 12 we compare inWLR only against ReProCS on two complex sequences: 80 frames of *Lake*, frame size $[72, 90]$, and, 50 frames of *Person*, frame size $[120, 160]$. In those sequences, for inWLR, we set $p = 8$ and 5, respectively. Due to the absence of ground truth we only provide qualitative comparison. In Figure 13 we demonstrate the performance of inWLR on two data sets with dynamic background and semi-static foreground. In almost every video sequence, inWLR performs reasonably well. See Table 1 for the comparisons between computational time.

5. Conclusion

In this paper we propose a novel background estimation model which operates on the entire data set in a batch-incremental way and adaptively determines the background frames without requiring any prior estimate. The proposed



(a) (b) (c)

Figure 8: (a) Precision-Recall curves on Stuttgart *Basic* scenario to compare between ReProCS, inWLR, and GRASTA. MSSIM on Stuttgart *Basic* scenario to compare between: (b) ReProCS and inWLR, (c) incPCP and inWLR.

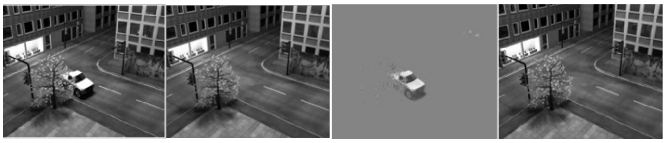


Figure 9: *Basic* scenario frame 420. Left to right: Original, incPCP background, incPCP foreground, and inWLR background. Both methods work equally well in detecting the dynamic foreground object.

model demands less on storage and allows slow change in background. Through extensive qualitative and quantitative comparison on real and synthetic video sequences, we establish our claim and demonstrate the robustness of our model. The batch sizes and the parameters in our model are still empirically selected. In future, we plan to propose a more robust estimate of the parameters and explore the possibilities in dealing with videos of more dynamic background using our algorithm.

References

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

[2] 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*, 23:1–71, 2017.

[3] 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.

[4] J. F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[5] E. J. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the Association for Computing Machinery*, 58(3):11:1–11:37, 2011.

[6] A. Dutta, B. Gong, X. Li, and M. Shah. Weighted singular value thresholding and its applications to background estimation, 2017.

[7] A. Dutta and X. Li. A fast algorithm for a weighted low rank approximation. *15 th IAPR International Conference on Machine Vision Applications*, 2017.

[8] 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.

[9] A. Dutta and X. Li. Weighted low rank approximation for background estimation problems. 2017.

[10] D. Farcas, C. Marghes, and T. Bouwmans. Background subtraction via incremental maximum margin criterion: a discriminative subspace approach. *Machine Vision and Applications*, 23(6):1083–1101, 2012.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

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

Figure 10: *Basic* scenario frame 600. Left to right: Original, incPCP background, incPCP foreground, inWLR background, inWLR SSIM, and incPCP SSIM. incPCP fails to detect the static foreground object, though a careful reader can detect a blurry reconstruction of the car in incPCP foreground. However, the SSIM map of both methods are equally good.

Figure 11: *Waving Tree* frame 247, frame size [120,160]. Left to right: Original, inWLR background ($p = 6$), GFL background, ReProCS background, inWLR SSIM (MSSIM: 0.9592), GFL SSIM (MSSIM: 0.9996), and ReProCS SSIM (MSSIM: 0.5221). inWLR and GFL recover superior quality background.

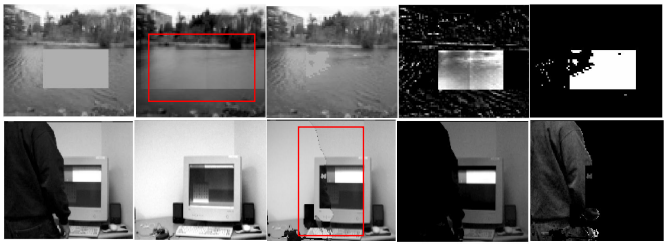


Figure 12: Left to right: Original, inWLR background, ReProCS background, inWLR foreground, and ReProCS foreground. In *Lake* sequence ReProCS performs better, and in *Person* sequence inWLR has better performance.

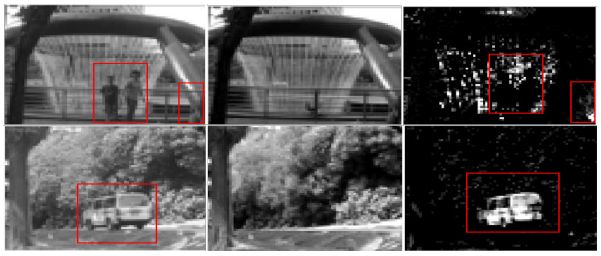


Figure 13: Top to bottom: *Fountain* 500 frames with $p = 5$, *Campus* 600 frames with $p = 6$, frame size [64,80]. Left to right: Original, inWLR background, and inWLR foreground.

[16] 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.

[17] Y. Li, L. Q. Xu, J. Morphett, and R. Jacobs. An integrated algorithm of incremental and robust pca. In *Proceedings 2003 International Conference on Image Processing*, volume 1, pages I–245–I–248, 2003.

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

[19] 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.

[20] 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.

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

[22] P. Rodrguez 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.

[23] P. Rodrguez 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.

[24] A. Sobral and A. Vacavant. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. *Computer Vision and Image Understanding*, 122:4–21, 2014.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] H. Zhao, P. C. Yuen, and J. T. Kwok. A novel incremental principal component analysis and its application for face recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(4):873–886, 2006.