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

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

Xin Li
UCF, USA
xin.li@ucf.edu

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

Abstract

Principal component pursuit (PCP) is a state-of-the-art approach to 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 incrementally. We build a batch-incremental background estimation model by using a special weighted low-rank approximation of matrices. Through experiments with real and synthetic video sequences, we demonstrate that our model is superior to the existing state-of-the-art background estimation algorithms such as GRASTA, ReProCS, incPCP, and GFL.

1. Introduction

Background estimation and moving object detection are two important steps in many computer vision systems and video-surveillance applications. In the past decade, one of the prevalent approaches used for background estimation treats it as a low-rank and sparse matrix decomposition problem [1, 3, 22, 29]. Oliver et al. [24] 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. [6, 21, 34], which is considered 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 \(a_i \in \mathbb{R}^m\) being vectorized, let the data matrix \(A = (a_1, a_2, \ldots, a_n) \in \mathbb{R}^{m \times n}\) be the concatenation of all the video frames. The foreground is usually sparse if its size is relatively small compared to the frame size [6, 21, 34]. Therefore, one can naturally 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. Therefore, RPCA solves:

\[
\min_B \| A - B \|_{\ell_1} + \lambda \| B \|_*,
\]

where \(\| \cdot \|_{\ell_1}\) and \(\| \cdot \|_*\) denote the \(\ell_1\) norm and the nuclear norm (sum of the singular values) of matrices, respectively, and \(\lambda > 0\) is a balancing parameter.

In contrast, 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. [13] 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 \(A_G = (\hat{B}_1 \hat{B}_2)\) such that

\[
(\hat{B}_1 \hat{B}_2) = \arg \min_{B = (B_1, B_2), \; \| B_1 \|_{\ell_1} \leq r, \; \| B_2 \|_{\ell_1} \leq r} \| A - B \|_F^2,
\]

where \(\| \cdot \|_F\) denotes the Frobenius norm of matrices. That is, Golub et al. required a few columns, \(A_1\), of \(A\) to be preserved to find 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 \(\hat{B}\) such that

\[
\hat{B} = \arg \min_{B} \| A - B \|_F^2.
\]

As it is well known, this problem is equivalent to principal component analysis (PCA) [18] and has a closed form solution that uses 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 \(\hat{B} = H_r(A) := PD_rQ^T\), 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** [13] Assume \(\text{rank}(A_1) = k\) and \(r \geq k\), the solution \(\hat{B}_2\) in (2) is given by

\[
\hat{B}_2 = \arg \min_{B_2} \| A_2 - B_2 \|_F^2 \quad \text{subject to} \quad \| B_2 \|_{\ell_1} \leq r - k,
\]
\[ \hat{B}_2 = P_{A_1}(A_2) + H_{r-k} \left( P_{A_1}^\perp(A_2) \right), \]

where \( P_{A_1} \) and \( P_{A_1}^\perp \) are the projection operators to the column space of \( A_1 \) and its orthogonal complement, respectively.

If we assume that some pure background frames are known, we can apply GHS by using these background frames as the first block matrix \( A_1 \). Similarly, recently, Xin et al. [35] proposed a supervised learning model called generalized fused Lasso (GFL) which solves:

\[ \min_{B} \quad \text{rank}(B) + \| A - B \|_{gfl}, \]

where \( \| \cdot \|_{gfl} \) denotes a norm that is a combination of the \( \ell_1 \) norm and a local spatial total variation norm (to encourage connectivity of the foreground). To solve GFL problem (5), Xin et al. [35] further specialized the above model by requiring \( \text{rank}(B) = \text{rank}(A_1) \). Note that, with this specialization, problem (5) can be viewed as a constrained low-rank approximation problem as in GHS problem (2), and it can be formulated as follows:

\[ \min_{B} \quad \| A - B \|_{gfl}, \]

\[ \text{rank}(B) \leq r \]

\[ B = (B_1, B_2) \]

\[ B_1 = A_1 \]

1.1. Incremental Methods

Conventional PCA [18] is an essential tool to numerically solve both RPCA and GFL problems. PCA operates at a cost of \( \min \{ O(m^2n), O(mn^2) \} \), 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 \( O((m+n)r) \), where \( r \) is the rank of the low-rank approximation matrix in each iteration, which is increasing. And a high resolution video sequence characterized by very large \( m \), is computationally extremely expensive 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 \( 300 \times 400 \) 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 \( [A \ A] \) by 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 [37]. Similar to the IPCA, several methods have been proposed to solve the background estimation problem in an incremental manner [12, 20]. In 2012, He et al. [16] 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. [14] proposed another online algorithm for separating sparse and low dimensional subspace. For initial sequence of training background video frames, Guo et al. devised a recursive projected compressive sensing algorithm (ReProCS) for background estimation (see also [15, 25]). Following a modified framework of the conventional RPCA problem, Rodriguez et al. [28] formulated the incremental principal component pursuit (incPCP) algorithm which processes one frame at a time incrementally and uses only a few frames for initialization of the prior (see also [26, 27]). To the best of our knowledge, these are the state-of-the-art incremental background estimation models.\(^1\)

1.2. Contributions

In this paper, we propose an adaptive batch-incremental model for background estimation. Our model finds the background frame indexes robustly and incrementally 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 naturally.

Before describing our model, let us revisit the idea of Golub et al. Inspired by (2) and by applications in which \( A_1 \) may contain noise, we require \( \| A_1 - B_1 \|_F \) small not \( B_1 = A_1 \) as in (2). This leads Dutta et al. [9, 10, 11] to consider the following more general weighted low-rank (WLR) approximation problem:

\[ \min_{X=(X_1, X_2)} \quad \| (A_1 \ A_2) - (X_1 \ X_2) \|_F^2, \]

where \( W \in \mathbb{R}^{m \times n} \) is a matrix with non-negative entries and \( \odot \) denotes the Hadamard product. Using \( W = (W_1 \ I) \), Dutta et al. [9, 11] applied (7) 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 \( \ell_1 \) norm used in conventional and in the incremental methods, a weighted Frobenius norm used in [9, 11] to make WLR robust to the outliers for background estimation problems [9, 11]. Our batch model is as fast as incPCP and ReProCS also, our model can deal with high quality video sequences as well as incPCP and ReProCS. Some conventional algorithms, such as supervised GFL or ReProCS require an initial training sequence, which does not contain any foreground object. Our experimental results for 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 this algorithm is well suited for real time high-definition video surveillance and for panning motions of the camera where the background slowly evolves.

\(^1\)We refer the readers to [17, 30].
Algorithm 1: WLR Algorithm

1 Input: $A = (A_1 \ldots A_2) \in \mathbb{R}^{m \times n}$ (the data matrix), $W = (W_1 \ldots) \in \mathbb{R}^{m \times n}$ (the weight), threshold $\epsilon > 0$;

2 Initialize: $(X_1)_{i,0}, C_0, B_0, D_0$;

3 While not converged do

4 $E_p = A_1 \oplus W_1 \odot W_1 + (A_2 - B_p D_p) C_p^T$;

5 For $i = 1 : m$ do

6 $(X_1(i,i))_{p+1} = (E(i,i))_p (\text{diag}(W_1^2(i,1) W_1^2(i,2) \ldots W_1^2(i,k)) + C_p C_p^T)^{-1}$;

7 $C_{p+1} = ((X_1)_{p+1}^T, (X_1)_{p+1}^{-1}) (X_1)_{p+1} (A_2 - B_p D_p)$;

8 $B_{p+1} = (A_2 - (X_1)_{p+1} C_p) D_p (D_p^T D_p)^{-1}$;

9 $D_{p+1} = (B_{p+1}^T B_{p+1})^{-1} B_{p+1} (A_2 - (X_1)_{p+1} C_{p+1})$;

10 $p = p + 1$;

End do

End while

11 Output: $(X_1)_{p+1}, (X_1)_{p+1} C_{p+1} + B_{p+1} D_{p+1}$.

1.3. The WLR algorithm

We now briefly overview the WLR algorithm proposed by Dutta et al. [7, 10, 11]. Let rank($X_1) = k$. Then any $X_2$ such that rank($X_1 \times X_2$) $\leq r$ can be given in the form, $X_2 = X_1 C + B D,$ for some matrices $B \in \mathbb{R}^{m \times (r-k)}, D \in \mathbb{R}^{(r-k) \times (n-k)}$, and for $C \in \mathbb{R}^{k \times (n-k)}$. Therefore, problem (7) with $W = (W_1 1)$ of compatible block partition is reduced to:

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

(8)

The complexity of one iteration of Algorithm 1 is $O(mk^3 + mn)$ [10].

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). Our algorithm fully exploits 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 $A$. First, we partition the original video sequence into $p$ batches: $A = (A^{(1)} \ldots A^{(p)})$, 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 coarsely estimates the possible background frame indices of $A^{(1)}$: we run the classic singular value thresholding (SVT) of Cai et al. [5] on $A^{(1)}$ to obtain a low rank component (containing the estimations of the background frames) $F_{1n}$ and let $\tilde{F}_{1n} = A^{(1)} - B_{1n}^{(1)}$ be the estimation of the foreground matrix (Step 2). From the above estimates, 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 obtain the background $B^{(j)}$ (Step 9). To determine the indices of the frames that contain the least information of the foreground we identify the “best background frames” by using a modified version of the percentage score model by Dutta et al. [8] (Step 5). Using this modified model allows us to estimate $k$, $r$, and the first block $A_1$ which contains the background prior knowledge (Steps 6-7). Weight matrix $W = (W_1 1)$ is chosen by randomly picking the entries of the first block $W_1$ from an interval $[\alpha, \beta]$ by 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 [9, 10]. 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 $A^{(j)}$ does not grow with $j$. The output of the algorithm is the background estimations for each batch 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 does not change much. However, when the camera is panning and the background is continuously evolving, our inWLR could be proven very robust as new frames are entering in
1. Apply WLR on matrix $A^{(1)} = (A^{(1)}_1)$ in each incremental step (Step 9) of Algorithm 50 is approximately 1 tends to slow down with higher overhead than 4 for 1 is used on the entire 100 150 0 2 does (see Table 5).

Now, we analyze the complexity of Algorithm 2 for equal batch size. Primarily, the cost of the SVT algorithm in Step 2 is only $O(\frac{mn^2}{p})$. Next, in Step 9, the complexity of implementing Algorithm 1 is $O(mk^{3} + \frac{mnr}{p})$. Note that $r$ and $k$ are linearly related and $k \leq k_{max}$. Once we obtain 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 justify as follows: the estimated background frames $\hat{A}_i$ are not necessarily exact background; they are only estimations of background. Thus, 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 $O(m(k^{3}p + nr))$. Note that the complexity of Algorithm 2 is dependent on the partition $p$ of the original data matrix.

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

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

2.1. Complexity analysis

Due to the availability of ground truth frames for each foreground mask, we use 600 frames of the Basic scenario of the Stuttgart artificial video sequence [4] to analyze them quantitatively and qualitatively. 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. Qualitative and quantitative analysis

3.1. Comparison with GHS

Because 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 [11]. The structural similarity index (MSSIM) is used to quantitatively evaluate the overall image quality as it mostly agrees with the human visual perception [32]. To calculate the MSSIM of each recovered foreground video frame, we consider a $11 \times 11$ Gaussian window with standard deviation ($\sigma$) 1.5. We perceive the information of how the high-intensity regions of the images come through the noise, and consequently, we pay much less attention to the low-intensity regions. We remove the noisy components
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.

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.

from the foreground recovered by inWLR, \( F \) by using a threshold \( \epsilon_1 \) (calculated implicitly in Step 5 of Algorithm 2 to choose the background frames, see [8]), such that we set the components below \( \epsilon_1 \) 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 \times 176. On the other hand, the GHS inspired model and WLR model take approximately 273.8382 and 64.5 seconds, respectively. The MSSIM presented in Figure 2 indicates that the inWLR and GHS inspired model produce the 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 that 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. [35]\(^2\). For both models, we use 200 frames of the Basic sequence, each frame resized to [144, 176]. The background recovered and the SSIM map in Figures 4 and 11 show that both methods are very competitive. However, it is worth mentioning that inWLR is extraordinarily time efficient compare with 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 linspace(0, 255, 100) to compare pixel-wise predictive analysis between each recovered foreground frames.

\(^{2}\) http://idm.pku.edu.cn/staff/wangyizhou/
frame and the corresponding ground truth frame.

3.3.1 Comparison with GRASTA [16]

At each time step \( i \), GRASTA solves the following optimization problem: For a given orthonormal basis \( U_{\Omega_s} \in \mathbb{R}^{[\Omega_s] \times d} \) solve

\[
\min_x \| U_{\Omega_s} x - A_{\Omega_s}(; i) \|_{\ell_1},
\]

where each video frame \( A(; i) \in \mathbb{R}^m \) is subsampled over the index set \( \Omega_s \subset \{1, 2, \cdots, m\} \) following the model: \( A_{\Omega_s}(; i) = U_{\Omega_s} x + F_{\Omega_s}(; i) + \epsilon_{\Omega_s} \), such that, \( x \in \mathbb{R}^d \) is a weight vector and \( \epsilon_{\Omega_s} \in \mathbb{R}^{[\Omega_s]} \) is a Gaussian noise vector. After updating \( x \), one has to update \( U_{\Omega_s} \). We set the subsample percentage \( s \) at 0%, 10%, 20%, and at 30% respectively, estimated rank 60, and keep the other parameters the same as those in [16]. The GRASTA code is obtained from the author’s website.\(^3\) Note that for a lower estimated rank GRASTA does not perform well in Basic scenario. Referring to 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 show the superior performance of inWLR on the Stuttgart Basic scenario.

3.3.2 Comparison with ReProCS [14]

ReProCS is a two stage algorithm. In the first stage, given a sequence of training background frames, say \( t \), the algorithm 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 that 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 support our claim quantitatively for the Basic sequence. Although the average computation time for ReProCS is 15.644460 seconds, which is better than inWLR.

3.3.3 Comparison with incPCP [28]

incPCP follows a modified framework of PCP but is built 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:

\[
\min_{B_k, F_k} \| B_k + F_k - A_k \|_F^2 + \lambda \| F_k \|_{\ell_1},
\]

where \( A_k = [A_{k-1} A(; k)] \) and \( B_k = [U_r \Sigma_r V^T_r B(; k)] \) such that \( U_r \Sigma_r V^T_r \) is a partial SVD of \( B_{k-1} \). According to [28], 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 the author’s website.\(^4\) From the MSSIM presented in Figure 8c and the background recovered by both methods in Figure 9, both methods appear to perform equally well on the Basic scenario. However, when the foreground is static (as in frames 551-600 of the Stuttgart sequen-

\(^3\) https://sites.google.com/site/hejunzz/grasta

\(^4\) https://sites.google.com/a/istec.net/prodrig/Home/en/pubs/incpcp
4. Results on real world sequences

To further validate the robustness of inWLR, we tested it on some challenging real world video sequences containing occlusion, dynamic background, and static foreground. We use I2R and SBI dataset [19, 23, 31] for this purpose. 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 against ReProCS on two complex sequences: 80 frames of Lake, sequence), the $\ell_1$ norm in incPCP cannot capture the foreground object, thus resulting in the presence of the static car as a part of the background (see Figure 10). On the other hand, our inWLR successfully detects and removes the static foreground from the background.

5. Conclusion

In this paper we proposed a novel model for background estimation. Our model operates on the entire data in a batch-incremental way and adaptively determines the background frames without requiring any prior estimate. Furthermore, our model does not require much storage and allows slow changes in the background. Our extensive qualitative and quantitative comparison on real and synthetic video sequences demonstrate the robustness of our model. The batch sizes and the parameters in our model are still empirically selected. Therefore, we plan to propose a more robust estimate of the parameters and explore the possibilities that our algorithm can handle videos of more dynamic background.

The MATLAB codes for MSSSIM and CQM are downloaded from http://sbmi2015.na.icar.cnr.it/SBIdataset.html
References


