JA-POLS: a Moving-camera Background Model via Joint Alignment and Partially-overlapping Local Subspaces

Irit Chelly
Ben-Gurion University
tohamy@post.bgu.ac.il

Vlad Winter
Ben-Gurion University
winterv@post.bgu.ac.il

Dor Litvak
Ben-Gurion University
dorlit@post.bgu.ac.il

David Rosen
Massachusetts Institute of Technology
dmrosen@mit.edu

Oren Freifeld
Ben-Gurion University
orenfr@cs.bgu.ac.il

Abstract

Background models are widely used in computer vision. While successful Static-camera Background (SCB) models exist, Moving-camera Background (MCB) models are limited. Seemingly, there is a straightforward solution: 1) align the video frames; 2) learn an SCB model; 3) warp either original or previously-unseen frames toward the model. This approach, however, has drawbacks, especially when the accumulative camera motion is large and/or the video is long. Here we propose a purely-2D unsupervised modular method that systematically eliminates those issues. First, to estimate warps in the original video, we solve a joint-alignment problem while leveraging a certifiably-correct initialization. Next, we learn multiple partially-overlapping local subspaces and how to predict alignments. Lastly, in test time, we warp a previously-unseen frame, based on the prediction, and project it on a subset of those subspaces to obtain a background/foreground separation. We show the method handles even large scenes with a relatively-free camera motion (provided the camera-to-scene distance does not change much) and that it not only yields State-of-the-Art results on the original video but also generalizes gracefully to previously-unseen videos of the same scene. Our code is available at https://github.com/BGU-CS-VIL/JA-POLS.

1. Introduction

Background modeling is an important video-analysis tool with applications such as tracking and change detection. In the static-camera case, the problem has been solved successfully[42, 11, 5, 1, 19, 6]. Our focus is on the more challenging case, where the camera is moving. There, the success has been more modest, as existing methods are limited to highly-restricted motions (e.g.: translations; small motions; jitter) and/or a small accumulative motion across the video; moreover, they do not directly generalize to previously-unseen misaligned videos. This raises a natural question: why not employ the following seemingly-simple 3-step solution? 1) align all the video frames (via, e.g., [4, 27, 22, 8, 9, 21]); 2) learn a Static-camera Background (SCB) model of the global scene from the aligned frames; 3) warp previously-unseen frames toward the SCB model and apply the latter to the former, where the warping is done using, e.g., either classical tools (see [44] and references therein) or methods such as PoseNet [25]. Unfortunately, this logical approach, whose first two steps are exemplified by the clever PRPCA [30], suffers from severe drawbacks (related to, among other things, scalability and optimization challenges) that hinder its applicability, especially for large scenes (i.e., a large accumulative camera motion) and/or long videos. A more popular
alternative focuses on incremental model updates to perform background/foreground (BG/FG) separation of the next frame in the video stream. The lack-of-memory property of the approach, however, prevents an effective use of all the previously-acquired data (especially when the camera returns to regions covered earlier, possibly at new orientations/positions). We also note that existing methods, whether global (e.g., PRPCA) or incremental, target BG/FG separation in the original video and lack a readily-available mechanism to do so for previously-unseen unaligned frames (unless it is the next frame right at the end of the original video). Our approach is different. Particularly, we propose a novel method, self-coined JA-POLS (short for Joint Alignment and Partially-overlapping Local Subspaces), for unsupervised learning of a Moving-camera Background (MCB) model; see Fig. 1. JA-POLS, a purely-2D modular method, allows for large camera motions (either accumulative ones or between consecutive frames) and provides a mechanism for warping previously-unseen frames toward the model. The model itself scales gracefully since rather than trying to capture the background of the entire scene using a single low-dimensional global (“panoramic-size”) subspace, it consists of multiple smaller Partially-overlapping Local Subspaces (POLS). As we show, JA-POLS not only yields State-of-the-Art (SOTA) results on the original data but also generalizes to previously-unseen unaligned videos. Our key contributions are as follows: 1) a novel MCB method that allows for a substantial and relatively-free camera motion; 2) the alignment of the original frames is done jointly (not pairwise) and utilizes an efficient initialization with theoretical guarantees; 3) all the alignment-related computations are done in 2D directly from image measurements, obviating the need of an explicit 3D-scene reconstruction, of constructing a global panoramic image, and of camera calibration. 4) The POLS model overcomes the issues that prevent a single global model from handling large scenes and/or long videos; 5) unlike competing methods, which focus only on BG/FG separation in the original video and/or the next frame, JA-POLS also provides a mechanism for aligning frames taken from new videos (covering the same scene but taken at possibly-different times and from possibly-different camera poses).

2. Related Work

The global approach to MCB modeling starts with building a representation of the entire scene. This usually involves, as preprocessing, aligning the frames of the original video, thereby reducing the MCB problem to a typically-large SCB problem with missing data [30].

Image alignment. In [10, 30], homographies between consecutive frames are estimated, while [24] uses a multi-layer homography. Works such as [48, 29] generate an adaptive panoramic image, while [45] assumes a PTZ camera. Also related is video stabilization; e.g., [14] finds an optimal steady-camera path using pairwise transformations between consecutive frames, while [28] minimizes a global cost based on the warped frames. Most of the works above assume a calibrated camera and/or a highly-restricted camera motion (e.g., small motions or PTZ). Moreover, transformation estimation is usually done pairwise and sequentially; this is prone to accumulative errors as well as perspective distortions when the scene is wide. AutoStitch [4] uses bundle adjustment upon computing pairwise geometric matches. Available implementations of [4] do not scale well with the number of input frames and typically handle, at most, only a few hundreds of frames. Other alignment methods use 3D data, e.g., the (non-visual) Simultaneous Localization and Mapping (SLAM) [31, 26] which estimates a model of the environment together with a dynamic camera pose. PoseNet [25] is a neural net that estimates camera poses from images and uses ground-truth 3D poses in its training. Such methods rely on depth data (in some SLAM methods) and/or expensive 3D reconstruction procedures such as Structure-from-Motion [47] (as in, e.g., [25]). Our method includes a regression net which is, conceptually, akin to PoseNet; the differences are that ours is purely 2D-based and that what it predicts are invertible affine transformations.

Background models. For already-aligned images, SCB models have been researched extensively. Earlier methods focused on pixelwise models [42, 18, 52]. Thurnhofer-Hemsi et al. [45] use a competitive-learning net that learns receptive fields in the panoramic scene. Another main approach, closer to ours, is learning a low-dimensional subspace. Principal Component Analysis (PCA) can be used but only if it can be assumed that the data contains neither foreground objects nor outliers. Otherwise, Robust PCA (RPCA) methods are preferred. The first RPCA in computer vision was proposed in [11]. Later, Candès et al. [5] and similar works [51, 16] used a “low-rank and sparse” (“L + S”) data decomposition. The low-rank part represents the background while the sparse part models outliers. Unfortunately, all these models [11, 5, 51, 16] do not scale. A scalable RPCA was proposed in [17] based on Trimmed Grassmann Averages (TGA); see also [6]. Works such as [1, 19, 15] use L + S decompositions within subspace tracking; despite the word “tracking”, these methods, which focus on subspace updates, are more suitable for SCB than MCB models. A related MCB approach, t-GRASTA [20], relies on [34] and alternates between motion estimation and subspace learning. DECOLOR [50] is a similar
MCB approach focusing on moving-object detection. These MCB methods linearly approximate the motions and thus cannot handle a large camera motion. The incPCP-PTI method [7] continuously estimates a rigid-body transformation between each new frame and the low-rank component of the previous one, and then applies it to the whole low-rank matrix. This method targets a PTZ camera. Several works focus on moving-object detection in a moving camera: e.g., [49] segments optical flow into BG/FG while [38] classifies feature-based trajectories. These works, which solve a related but slightly-different problem from ours, cannot detect static changes and cannot handle long sequences; see also [2]. Lastly, all works mentioned in this section lack a direct way to handle previously-unseen misaligned frames (unless the frame is next consecutive one).

3. The Proposed Method: JA-POLS

The outline of the proposed method, which treats the original video as (unlabeled) training data, is as follows. Phase 1 (§ 3.1): Given a training video, we solve an unsupervised joint-alignment problem using: a novel smart initialization with theoretical guarantees; a Lie-algebraic parameterization/regularization; a Spatial Transformer Net (STN). See [41] for benefits of a Lie-algebraic parameterization (via the matrix exponential) of affine maps within STNs. Phase 2 (§ 3.2): Upon the joint alignment, we learn two tasks independently: 1) alignment prediction; 2) learning multiple local low-dimensional robust background models over partially-overlapping areas, yielding a set of local linear subspaces, each associated with a different region of the scene. Phase 3 (§ 3.3): In test time, a new frame is first warped toward the global scene via the (refined) predicted alignment, and then projected on the Partially-overlapping Local Subspaces (POLS). The average of the projections results in an effective BF/FG separation. See flowcharts in Fig. 2. Importantly, our method requires neither the creation of a single global model for an entire panoramic scene nor 3D reconstruction; rather, it employs a decentralized and localized approach and is purely 2D-based. As our method uses Lie groups/algebras, our Sup. Mat. contains all the relevant required background used below.

Notation. Let SE(2) and Aff(2) denote the Special Euclidean and affine groups in 2D, respectively. Both groups can be seen as nonlinear spaces of 3-by-3 matrices acting on \( \mathbb{R}^2 \) (in homogeneous coordinates) and SE(2) \( \subsetneq \) Aff(2). Let \( \text{aff}(2) \) denote the Lie algebra of Aff(2): \( \text{aff}(2) \) is a 6D linear space of 3-by-3 matrices. Let vec : \( \text{aff}(2) \to \mathbb{R}^6 \) denote a linear bijection. The matrix exponential and logarithm, \( \exp : \text{aff}(2) \to \text{Aff}(2) \) and \( \log : \text{Aff}(2) \to \text{aff}(2) \), connect the algebra to the group. If \( \theta \in \mathbb{R}^6 \), then \( \theta^T = \exp(\text{vec}^{-1}(\theta)) \in \text{Aff}(2) \) is the affine transformation parameterized by \( \theta \) and \( d(T^\theta,\text{SE}(2)) \) (see Sup. Mat.) measures how far \( \theta^T \) is from SE(2). Let \( \Omega_{\text{scene}} \) be the domain of the panoramic-size image, and let \( D \) be the number of its pixels.

3.1. Unsupervised Joint Alignment

Given training frames, \( (x_i)_i \in \mathbb{N} \), we seek \( \theta_i^\theta \) \( i \in \mathbb{N} \) \( \subset \) Aff(2) that minimize the (robustified) variance of all RGB values over the warped images, \( (x_i^\theta)_i \in \mathbb{N} \), where \( x_i^\theta = x_i \circ T_\theta \). Let \( d_{\theta}^\theta < D \) be the number of pixels in \( \Omega_i^\theta \), the domain of \( x_i^\theta \). Note that \( \Omega_i^\theta \subseteq \Omega_{\text{scene}} = \bigcup_{i=1}^N \Omega_i^\theta \). Let \( w_i^l \) denote a binary weight indicating whether or not pixel \( l \) in \( \Omega_{\text{scene}} \) is also in \( \Omega_i^\theta \), and let \( x_i^\theta \) denote pixel \( l \) of \( x_i^\theta \) (if \( w_i^l = 0 \), then \( x_i^\theta \) is undefined). Consider the following minimization of a robust joint-alignment loss,

\[
\min_{(\theta_i^\theta)_i} \sum_{i=1}^D \sum_{l=1}^D w_i^l \rho(x_i^\theta l - \mu_l, \sigma) \leq \text{aff}(2) \\
\mu_l = \frac{\sum_{i=1}^N w_i^l x_i^\theta l}{\sum_{i=1}^N w_i^l}, \quad T_\theta^\theta = \exp(\text{vec}^{-1}(\theta)) \in \text{Aff}(2) \quad (1)
\]

where \( \theta_i^\theta \in \mathbb{R}^6 \), \( \mu_l \) is the average of pixel stack \( l \), \( \{x_i^\theta l : w_i^l = 1\} \), and \( \rho(\cdot, \sigma) \) is Huber’s loss [3] of parameter \( \sigma > 0 \). This loss is akin to those used by others for joint alignment. In our setting, however, that loss is specially-hard to minimize: since usually \( d_{\theta}^\theta < D \), and
Given \((g_{ij})_{i,j} \subseteq \text{SE}(2)\), find
\[
(g_{ij})_{i=1}^{N} = \arg \min_{t_i \in \mathbb{R}^2, R_i \in \text{SO}(2)} \sum_{(i,j) \in \mathcal{E}} |\kappa_{ij} R_j - R_i \tilde{R}_{ij}|^2_F + \tau_{ij} \|t_j - t_i - R_i \tilde{t}_{ij}\|^2_{\mathcal{E}}
\]
where \(\|\cdot\|^2_F\) is the Frobenius norm. \(\tilde{g}_{ij} = \left[\begin{array}{c} R_{ij} \ t_{ij} \\ 0_{1 \times 2} \ 1 \end{array}\right] \in \text{SE}(2), \quad \kappa_{ij} > 0, \quad \tau_{ij} > 0, \quad \text{and } g_i = \left[\begin{array}{c} R_i \ t_i \\ 0_{1 \times 2} \ 1 \end{array}\right] \in \text{SE}(2).

To solve it, we employ SE-Sync \cite{tsui2021synchronization}, an efficient and certifiably-correct algorithm for synchronization over SE(2) (more generally, SE(n)). SE-Sync recovers certifiably globally-optimal solutions provided that the noise corrupting the \((\tilde{g}_{ij})\) is not too large; moreover, even when exact optimality fails to hold, SE-Sync still produces a reasonable approximated solution, together with an upper bound on that solution’s (global) suboptimality. SE-Sync thus provides us with a good estimate of \((g_{ij})_{i=1}^{N} \subseteq \text{SE}(2)\) to jointly align \((x_i)_{i=1}^{N}\), where the alignments are restricted to SE(2). As mentioned above, we subsequently refine the transformations over the larger group, Aff(2)\(^N\), using the STN. An instance of SE-Sync is modeled as a sparse and nonlinear undirected graph \((\mathcal{V}, \mathcal{E})\); the nodes, \(\mathcal{V}\), correspond to (the coordinate systems of the) input frames, \(x_i\), while the edges, \(\mathcal{E}\), correspond to a set of noisy estimates of relative transformations; i.e., \(\mathcal{E}_{ij}\) corresponds to \(\tilde{g}_{ij}\), the estimated transformation from coordinate system \(j\) to coordinate system \(i\). We build \(\mathcal{E}\) by connecting each node only with the next five ones; the rationale is that relative transformations between frames in such a short batch are usually small. The estimation of \(\tilde{g}_{ij} \in \text{SE}(2)\), is done via established vision tools; see our Supp. Mat.

For each \((x_i, t_j)\) image pair, the result of that estimation procedure is not only the (estimated) relative transformation, \(\tilde{g}_{ij} = \left[\begin{array}{c} R_{ij} \ t_{ij} \\ 0_{1 \times 2} \ 1 \end{array}\right]\), but also (estimated) precisions, \(\tau_{ij}\) and \(\kappa_{ij}\), of \(t_{ij} \in \text{SO}(2)\) and \(R_{ij} \in \text{SO}(2)\), respectively, used in Eq. (3).

### 3.2 Learning

**POLLS learning.** Given the original frames and their estimated global affine transformations, \((x_i, T_{ij})_{i=1}^{N}\), a seemingly-straightforward approach is to use the warped images, \((x_i)^{T_{ij}}\)\(_{i=1}^{N}\), to learn a subspace-based SCB model (e.g., some \(k\)-dimensional subspace such as PCA or one of its robust variants \cite{ma2019deep, tsui2018deep, tsui2019deep, tsui2021deep}) whose domain is \(\Omega_{\text{scene}}\). The subspace would be represented by an orthogonal \(D \times k\) matrix \(V_{\text{scene}}\). We note, however, that doing so for very large scenes: 1) can be very expensive as \(D\) can be huge; 2) requires learning a model where, in each example, most of the data is missing, as typically \(d_{ij}^m \ll D\); 3) requires k
large enough to capture the variability the dynamic background exhibits in the entire scene; 4) can have (for long videos) a prohibitively-large memory footprint. We thus argue that, especially for large scenes and/or long videos, there must be a better way: if all we want is BG/FG separation in a small region (e.g., the size of a single frame), why should we do everything at a whole-scene scale? As an alternative, we propose learning portions of the scene individually. Particularly, we split $\Omega_{\text{scene}}$ into $M$ partially-overlapping domains; see Fig. 3c. Let $m \in \{1, \ldots, M\}$. In our experiments, each such domain, $\Omega_m$, is a 250-by-420 rectangle, obtained via a raster scan of 30-pixel horizontal/vertical strides (this determines $M$). See Sup. Mat. for the effect of using other window sizes. Let $n_m < N$ denote the number of warped training images whose domain overlaps $\Omega_m$ by more than 85%. We denote these images by $\{\tilde{x}_i^\theta_q\}_{q=1}^{n_m} \subset \{\tilde{x}_i^\theta_q\}_{i=1}^N$. Let $d_m = d$ denote the (constant) number of pixels in $\Omega_m$. We form a (3d)-by-$n_m$ local-data matrix, $Y_m$, whose generic column, $y_q \in \mathbb{R}^{3d}$, contains the RGB values of $\tilde{x}_i^\theta_q$ in indices corresponding to overlapping pixels between $\Omega_m$ and $\tilde{\Omega}_q^\theta_q$ (the domain of $\tilde{x}_i^\theta_q$), and NaNs (to indicate missing data) in indices corresponding to pixels that are in $\Omega_m$ but not in $\tilde{\Omega}_q^\theta_q$. We can now apply, to each $Y_m$, any off-the-shelf method for linear dimensionality reduction suitable for background modeling. Let $V_m$ denote the subspace learned from $Y_m$. Note that: 1) each $Y_m$ is much smaller, in either dimension, than an analogous global-data matrix, $Y_{\text{scene}}$, that consists of all of the pixels in all of the warped frames (as well as many NaNs); 2) the $(V_m)$’s can be learned in parallel (our distributed implementation exploits that); 3) the relative portion of missing data in $Y_m$ is much smaller than that in $Y_{\text{scene}}$. To summarize, a POLS model is highly scalable and suffers less (than a global model) from missing data.

Alignment learning. Given a test frame, we seek to align it w.r.t. the global scene. In theory, one can use any off-the-shelf tool for pairwise alignment. However, this would suffer from two issues. 1) One would have to explicitly reconstruct a panoramic-size image. Tools for creating panoramas do not scale to a very large scene (e.g., thousands of frames). Moreover, committing to specific values of the panorama is error prone, and these mistakes can hurt alignment to it. 2) For a very large panorama, it is hard and/or time consuming to estimate the alignment using standard tools. Inspired by PoseNet [25], we prefer a deep-net approach. However, PoseNet’s training relies on expensive and error-prone 3D reconstruction pipeline. In contrast, we propose a purely-2D approach that reuses the estimated training warps. Given the already-available training pairs, $(x_i, T_{\theta_i})_{i=1}^N$, our regression net learns to map $x_i$, to $\theta_i \in \mathbb{R}^6 \cong \text{aff}(2)$, the (Lie-algebraic) parameter of an affine transformation in 2D. Learning transformation parameters between a pair of images directly has been done, e.g., in [13, 32]. Here, however, we are interested in learning the transformation for each frame to a global coordinate system. As sometimes a training video might consists of only tens or hundreds of frames, a too-low number for training, we resort to standard approaches for handling data scarcity: 1) Transfer learning [39]: in training, we merely fine-tune a GoogLeNet [43] pre-trained on ImageNet [12]. 2) Data augmentation (e.g., [40]): we generate synthesized frames by applying affine warps to the training data. For more details, see our Sup. Mat.

3.3. Test

In test time, the regression net takes an input image, $x$, and predicts $\theta = \theta(x)$. We warp $x$ by $T^\theta$ to produce a warped image in the global coordinate system, $x' = x \circ T^\theta$. The predicted alignment is sometimes imperfect (in terms of pixel-level accuracy), but it still locates $x$ very close to its desired destination. Thus, it is easy to refine the prediction as follows (the more expensive PoseNet would have also needed a refinement due to
similarly-imprecise results; see our Sup. Mat.). Let \( \Omega' \) be the domain of \( x' \). We select only the training pixel stacks that are either in \( \Omega' \) or near it (e.g., of distance < 10 pixels). We compute the average image of these pixel stacks. Then, we apply a standard tool for estimating the small residual transformation between \( x' \) and the average image (see Sup. Mat.). Let \( \tilde{x} \) denote the result of applying the refined warp, and let \( \tilde{\Omega} \) denote its domain (Fig. 3d, green border). Among the local subspaces we use only those whose domain has some overlap with \( \tilde{\Omega} \). We project \( \tilde{x} \) on the subspace of each such model as follows. Let \( m \in \{1, \ldots, M\} \) and let \( x_m \) be the following image, of domain \( \Omega_m \) (Fig. 3d, red border): \( x_m \) coincides with \( \tilde{x} \) (Fig. 3d, green border) on \( \Omega_m \cap \tilde{\Omega} \), and takes zero values on \( \Omega_m \setminus \tilde{\Omega} \) (Fig. 3d, black pixels). Recall there are \( d \) pixels in \( \Omega_m \). Let \( \mu_m \) denote the (vectorized) mean image associated with the \( k \)-dimensional subspace \( V_m \) (some of the subspace methods we experimented with do not use a mean image; for these, just take \( \mu_m \) to be the zero vector). Let \( b_m \) denote the projection of \( x_m \), the vectorized version of \( x_m \), on \( V_m \). It is defined by

\[
b_m = \mu_m + V_m \alpha_m \tag{4}
\]

\[
\alpha_m = \arg \min_{\alpha \in \mathbb{R}^k} \left\| W_m^2 (V_m \alpha - (x_m - \mu_m)) \right\|_{l_2}^2 \tag{5}
\]

where \( W_m \) is a diagonal \( 3d \times 3d \) matrix whose diagonal elements are 1 if they correspond to pixels in \( \tilde{\Omega} \), and 0 otherwise. This is a linear weighted least-squares problem, so it has a standard closed-form solution. The background image of \( \tilde{x} \), according to model \( m \), is given by \( b_m \), the “unvectorized” version of \( b_m \). Next, we compute the pixelwise average of the background images and warp the result back to the domain of \( x \). Letting \( b \) denote the resulting “unwarped” average image, \( b \) and \( f \triangleq x - b \) are the BG/FG separation of \( x \) by the POLS model; see, e.g., Fig. 1 and Fig. 2.

### 3.4. Camera-motion Types Handled by the Method

Recall that while we initialize the STN using SE(2), we end up working with Aff(2). Thus, while SE(2) cannot handle, e.g., changes in scale, JA-POLS can handle some variations of those distances, as long as these variations are not substantial enough to completely break the initialization; this caveat is the main limitation of our method. The method has no problems with camera motions that are roughly parallel to the scene (e.g., arbitrary in-plane rotations or vertical/lateral translations). Thus, it enables relatively-free camera movements, in the sense described above and, particularly, handles large accumulative motions (which none of the competing methods can handle) that are roughly parallel to the scene. Moreover, not relying on motion cues, in test videos JA-POLS can handle even fast motions of camera/objects. It also detects changes due to removal/displacement of static background objects. Empirically, perspective effects come into play only in very short ranges (e.g., < 2 meters). At least conceptually the affine STN could have been replaced with a homographic STN to handle such cases as well.

### 4. Results

We show, via qualitative and quantitative evaluations, that JA-POLS consistently yields SOTA results on moving-camera videos. Please also see the videos in the Sup. Mat. Whenever possible (see below), we compare against PRPCA [30], DECOLOR [50], Prac-ReProCS [15] and incPCP-PTI [7] in camera-motion datasets, and t-GRASTA [20] in camera-jitter datasets. We consider 9 moving-camera benchmark videos: Tennis, Swing, Stroller, Stunt, Flamingo, Hike (DAVIS dataset [35], [36]), Horse (Freiburg-Berkeley dataset [33]), Sidewalk and ContinuousPan (CDNet 2014 dataset [46]); these videos come with Ground-Truth (GT) FG masks. We also add our own 5 real videos, captured by us: Jitter, GardenShort, Kitchen, FastMotion and GardenWideScene. To enable qualitative evaluation on those, we inserted synthesized FG objects into each of them (see the Sup. Mat. for some visual examples). Together, the 14 videos cover a variety of camera-motion types: jitter (Sidewalk, Jitter), short videos with fairly-free motion (Tennis, Stroller, Swing, Flamingo, Hike, Horse, GardenShort), wide-scenes and/or long videos (Stunt, ContinuousPan, GardenWideScene), fast motion (FastMotion), and an indoor scene (Kitchen). Some of the sequences contain occlusions and illumination variations. To quantify FG estimation, we set a fixed threshold on the estimated FG, and compute the \( F_{\text{measure}} \) index defined by

\[
F_{\text{measure}} = \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}
\]

where precision and recall are derived from the GT and thresholded FG binary masks. Each local subspace in our POLS model was learned using either TGA [17] (5: 60\% trimming) or the denoising RPCA (Denoising-RPCA) which was proposed as part of the pipeline in [30]. We found that both of them outperformed [5], that in small datasets (\(< 80\) frames) Denoising-RPCA was the best, and that in larger ones TGA was the winner; see Sup. Mat. for details. We evaluate all methods on the same training images. As our competitors lack a direct way to process new test videos, only JA-POLS can be evaluated on the latter. Table 1 compares the different methods (except t-GRASTA, which is not designed to handle most of the sequences there). PRPCA, which uses a global model, faces memory issues in wide scenes and could not run
on them, even though the machine we used, whose full specs are at the Sup. Mat., had 256GB RAM (to clarify, JA-POLS does not need so much RAM). Also, as PRPCA relies on pairwise alignments, it suffers from accumulative alignment errors, and these worsen as the video gets longer (e.g., >100 frames). DECOLOR, Prac-ReProCS and incPCP-PTI degrade sharply when the motion is fast, suggesting they cannot handle rapid subspace evolutions. Figures 4 and 5 visualize typical BG/FG separation results produced by each method. For more visual and qualitative results, see Sup. Mat. Table 2 compares the performance, on the Jitter dataset, of JA-POLS and t-GRASTA [20] using 3 levels of motion. Figure 6 shows results for frames from a test video
Observation | JA-POLS | t-GRASTA (batch) | t-GRASTA (online)
--- | --- | --- | ---
Frame 3 | | | |
Frame 64 | | | |

Figure 6: FG-extraction of JA-POLS vs. t-GRASTA (in its batch and online modes), on 2 typical frames from Jitter dataset. Note this is a test video. Since t-GRASTA cannot handle test videos, we ran it here from scratch (i.e., not treating it as test data). While such a comparison is biased against JA-POLS, the latter still clearly wins. It also detects FG details even of small objects, and provides a relatively-clearer background.

Figure 7: Change detection: JA-POLS captures not only moving objects such as the man and the dog (which appears in the last 3 frames as can be seen upon zooming in) but also removal of static objects (a chair). of the Jitter dataset (see the caption for details). Figure 7 shows frames from a test video of the GardenShort dataset. Note that a static object (a chair), which was part of the background, is moved to another location during the test video. JA-POLS detects that as it does not rely on motion cues.

JA-POLS evaluation on test videos. Leveraging the alignment predictor, we also evaluated JA-POLS’ performance on 3 test videos. The resulting F-measure (mean±std) values are as follows: Kitchen: 0.50 ± 0.03, GardenWideScene: 0.55 ± 0.03 and Jitter: 0.88 ± 0.01.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>SE-Sync+STN</th>
<th>STN Only</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F-measure</td>
<td>Loss</td>
</tr>
<tr>
<td>Jitter</td>
<td>0.83 ± 0.03</td>
<td>0.018</td>
</tr>
<tr>
<td>CP</td>
<td>0.67 ± 0.05</td>
<td>0.064</td>
</tr>
</tbody>
</table>

Table 3: F-measure performance (mean±std) and the same STN loss (alignment+regularization) of JA-POLS on the Jitter and ContinuousPan (CP) data, with and without the SE-Sync initialization.

Ablation studies. Table 3 quantifies the importance of the SE-Sync initialization, showing its clear utility, especially when the motions go beyond mere jitters. The Sup. Mat. contains two additional such ablation studies. The first shows the importance of our regularization term, while the second shows that the alignment prediction is indeed indispensable.

Timings. SE-Sync takes a few seconds. POLS learning is fast, especially when TGA is used, and usually takes a few minutes. Since training the predictor is based on transfer learning, that too takes only minutes. The bottleneck is the STN optimization, whose running time, which ranges from ~15 minutes to several hours, depends on the length and complexity of the training video. However, processing a new test frame through the entire pipeline takes less than 2 [sec].

5. Conclusion

We proposed a novel MCB model and showed it achieves SOTA results and that it is highly scalable. We also showed that our choices in each step were judicious; e.g., we demonstrated that POLS consistently outperforms a global model as well as the critical roles of the SE-Sync initialization, the novel regularization, and the predictor. While competing MCB models focus on BG/FG separation in the original data and/or incremental updates given the next frame, ours also generalizes to unseen misaligned videos (of the same scene, taken possibly at different times).
References


