Subspace Tracking under Dynamic Dimensionality for Online Background Subtraction

Matthew Berger
Air Force Research Laboratory
Matthew.Berger.1@us.af.mil

Lee M. Seversky
Air Force Research Laboratory
Lee.Seversky@us.af.mil

Abstract

Long-term modeling of background motion in videos is an important and challenging problem used in numerous applications such as segmentation and event recognition. A major challenge in modeling the background from point trajectories lies in dealing with the variable length duration of trajectories, which can be due to such factors as trajectories entering and leaving the frame or occlusion from different depth layers. This work proposes an online method for background modeling of dynamic point trajectories via tracking of a linear subspace describing the background motion. To cope with variability in trajectory durations, we cast subspace tracking as an instance of subspace estimation under missing data, using a least-absolute deviations formulation to robustly estimate the background in the presence of arbitrary foreground motion. Relative to previous works, our approach is very fast and scales to arbitrarily long videos as our method processes new frames sequentially as they arrive.

1. Introduction

Background subtraction of video is a longstanding problem in computer vision. The basic problem is to distinguish portions of the scene which are moving from those parts which remain still. For scenes in which the camera is static, there exists a number of approaches for pixel-wise modeling of the background [9, 12]. For a dynamic camera, however, pixel-wise modeling is no longer possible without motion compensation. In these cases, the image motion provides an extremely useful cue for distinguishing portions of the scene which appear to be moving due to the camera movement from parts which are truly moving.

In this vein, factorization-based methods [24] have seen much development in motion segmentation from point trajectories, where trajectories are typically produced via optical flow. A requirement of such methods is for the trajectory durations to all be equivalent. However, in many real-world scenes, trajectories typically have widely varying durations due to trajectories entering and leaving the frame as a result of the camera moving or from self-occlusion of objects at different depth layers. While recent works have explicitly dealt with the problem of long-term point trajectories of varying duration for motion segmentation [5, 15, 18], along with extensions to background modeling [11], such methods face significant scalability issues in the number of trajectories. Indeed, as dense optical flow algorithms continue to scale and approach interactive rates for large resolution video [21, 22], background modeling of point trajectories should also be able to scale just as gracefully.

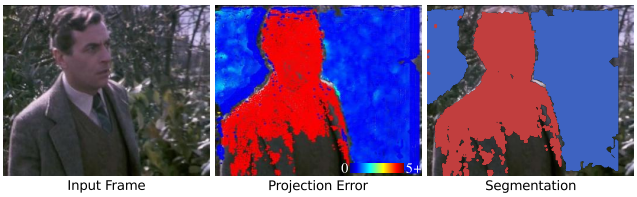

Figure 1. Given a frame from a video with a moving camera, we show the color-mapped error of the trajectory coordinates projected onto our dynamically tracked subspace, capable of handling trajectories entering and leaving the frame. Given the subspace we can segment background from foreground, shown on the right.

In this paper we address the problem of online background modeling of dense point trajectories. Assuming that camera motion is well-modeled by a low-dimensional linear subspace [23], our goal is to determine this subspace in the presence of trajectories dynamically entering and leaving the video. Unlike previous robust factorization-based approaches which use a sliding window [20], we employ subspace tracking to find the subspace at every frame. In particular, we show that as the ambient space dimensionality (number of trajectories) of the subspace changes, the frame-to-frame change in the subspace is well-bounded. This permits us to draw from subspace tracking techniques which handle missing data [12, 7], where we consider trajectories entering and leaving the video as missing data. By employing a suitable robust energy, we are able to simultaneously

separate the foreground from the background and update the subspace at every frame even in the presence of significant frame-to-frame changes in the number of trajectories. An overview of our approach is shown in Figure 1.

Our main contribution is a simple, fast, and scalable method for estimating background point trajectories under a moving camera. We demonstrate improved performance over existing methods which estimate the foreground in a sliding window fashion. Our method is also competitive with recent global offline motion segmentation methods while achieving several orders of magnitude improvement in computational time.

1.1. Related Work

Background modeling of point trajectories can be considered an instance of motion segmentation in the context of multibody factorization methods [25, 24, 19, 10]. Such methods segment motion by factorizing the matrix composed of all point trajectories using the fact that an orthographic camera model results in the trajectory matrix having rank at most 4 for a single rigid object, while for the background it is at most rank 3 [23]. However, as such methods rely on a matrix factorization, it is required that the point trajectories are equivalent length. Although some methods can handle small variability in the trajectory lengths [19, 10], in practice they tend to scale poorly in the number of trajectories.

More recent methods have considered ways of directly estimating the background motion under an orthographic camera model in a scalable manner with support for varying trajectory duration. The work of [20] performs matrix factorization in a sliding window fashion to identify background trajectories using RANSAC to robustly estimate the subspace. Another way of estimating the background is to decompose the point trajectory matrix into a low rank component to represent the background and a group-sparse component to represent the foreground [8] – such an approach can also use a temporal sliding window. Choosing the window size, however, is nontrivial in both cases, as there is a delicate tradeoff in supporting a large number of trajectories and having sufficient evidence to determine background trajectories.

To generalize the orthographic camera model, the work of [14] uses a Bayesian filtering framework to estimate appearance and motion models for background subtraction at the pixel level. This approach requires a high-quality segmentation at the first frame in order to reliably track subsequent frames. This can be problematic for long sequences containing complicated camera motion, due to the need to repeatedly reset the appearance and motion models and recompute the segmentation.

The works of [5, 18] model motion by constructing an affinity matrix over all trajectories, measuring commonality in translational [5] or similarity [18] transformations, and performing spectral clustering to arrive at a segmentation. These approaches handle variable-length trajectories by only measuring similarity across trajectories sharing common time intervals. This was recently extended to the online setting for background subtraction in [11], where Laplacian eigenmaps from the affinity matrix are used to represent the trajectories. A merging of the segmented trajectory clusters is then performed to identify segments which belong to the background. However, this affinity matrix will necessarily be dense for common point trajectories, hence the need to perform spectral clustering or construct Laplacian eigenmaps limits such methods to sparse trajectories.

Our approach is based on recent work related to online subspace tracking in the presence of missing data. GROUSE [1] tracks a subspace from incomplete data vectors via optimization on the Grassmanian manifold through efficient rank-1 subspace updates. A crucial element of GROUSE is that it is possible to estimate with high probability the residual error of projecting an incomplete data vector onto a subspace, as studied in [2]. This was extended in GRASTA [12] to handle outliers in the (incomplete) data vectors and subsequently applied to the problem of pixel-wise background estimation from video. Our approach is similar to GRASTA, as we also robustly track a subspace in the presence of outliers (i.e. foreground). However, there are two main challenges in leveraging GRASTA for trajectory modeling: 1) under dynamic dimensionality the subspace representation no longer lies on the Grassmanian, and 2) the subspace update needs modification as processing trajectories results in an update which is no longer rank-1.

Our method is inspired by PETRELS [7], which similarly tracks a subspace from incomplete data, but performs updates per-dimension. We extend this to the case where dimensions are dynamically added and removed.

2. Subspace Tracking with Changing Dimensions

Our approach for distinguishing foreground motion from background motion relies on maintaining a stable background model via subspace tracking. Given an estimate of the background subspace, we can project the set of trajectory coordinates at each frame onto the current subspace estimate in order to determine which trajectories are foreground and which are background. In this section, we discuss how to track a linear subspace when the ambient dimension is continually changing – the following section discusses how to estimate the foreground from the subspace via sparse estimation.

2.1. Subspace Tracking of Point Trajectories

Assume that the set of input point trajectories consists entirely of background motion and assume that this background motion follows an orthographic camera motion model. If we are tracking p points in each frame over f frames, the tracked pixel coordinates can be stacked into a matrix $\mathbf{P} \in \mathbb{R}^{p \times 2f}$, where $\mathbf{P}_{i,2j}$ and $\mathbf{P}_{i,2j+1}$ are respectively the x and y component of the i'th pixel coordinate at frame j – this corresponds to batch processing of trajectories. Assuming that the points are centered, under an orthographic camera model this matrix is at most rank 3 and admits the following factorization [23]:

$$\mathbf{P} = \mathbf{X}\mathbf{C}^T = \begin{bmatrix} x^1 & y^1 & z^1 \\ x^2 & y^2 & z^2 \\ \vdots & \vdots & \vdots \\ x^p & y^p & z^p \end{bmatrix} \begin{bmatrix} v_1^1 & w_1^1 & \cdots & v_f^1 & w_f^1 \\ v_1^2 & w_1^2 & \cdots & v_f^2 & w_f^2 \\ v_1^3 & w_1^3 & \cdots & v_f^3 & w_f^3 \end{bmatrix},$$

where $\mathbf{X}$ and $\mathbf{C}$ respectively represent the shape and camera parameters of the trajectories. Up to a linear transformation, $\mathbf{X}$ represents the corresponding 3D coordinates of the trajectories.

In the case where trajectories are dynamically going in and out of frame, obtaining such a factorization becomes challenging. We instead propose to dynamically *track* the rows of $\mathbf{X}$, rather than assume they remain fixed. Although it should only require 3 measurements to obtain a valid representation of each row, updating $\mathbf{X}$ via subspace tracking allows us to update each row so that either: a) it converges, b) it updates in order to accommodate the change in a row which has been added or removed, or c) its motion changes so that it no longer lies in the background.

More specifically, suppose that at time t we have our current shape subspace estimation $\mathbf{X}_{t-1} \in \mathbb{R}^{d \times 3}$, and we want to update to $\mathbf{X}_t \in \mathbb{R}^{(d+a) \times 3}$ based on the (mean-centered) coordinates of the trajectories at t, denoted $\mathbf{x}_t \in \mathbb{R}^{d+a}$ and $\mathbf{y}_t \in \mathbb{R}^{d+a}$, where a is the number of trajectories added at frame t. Furthermore, assume that r point trajectories were removed at frame t. We define our diagonal *sampling matrix* $\mathbf{S}_t$, where $\mathbf{S}_t$ is 0 for a point which has either been added or removed, and 1 otherwise. Hence, $\mathbf{S}_t\mathbf{x}_t$ and $\mathbf{S}_t\mathbf{y}_t$ effectively zeros out coordinates which have been added or removed since frame $t-1$.

We aim to find $\mathbf{X}_t$ by minimizing the following energy:

$$\mathbf{X}_t = \underset{\mathbf{X}}{\operatorname{argmin}} \sum_{i=1}^{t} \lambda^{t-i} \min_{\mathbf{v}_i, \mathbf{w}_i} (\|\mathbf{S}_i(\mathbf{X}\mathbf{v}_i - \mathbf{x}_i)\|_2^2 + \|\mathbf{S}_i(\mathbf{X}\mathbf{w}_i - \mathbf{y}_i)\|_2^2), \quad (1)$$

where $0 < \lambda \leq 1$ represents the discount factor: the smaller the λ, the less influence previous subspace estimates have on estimating $\mathbf{X}_t$.

Similar to PETRELS [7], we alternate between finding the camera parameters $\mathbf{v}_t$ and $\mathbf{w}_t$, and subsequently updating $\mathbf{X}_t$ based on the camera parameters. Note that the update is at the *pixel coordinate* level: we estimate $\mathbf{v}_t$ and $\mathbf{w}_t$ from $\mathbf{X}_{t-1}$, and $\mathbf{X}_t$ from $\mathbf{v}_t$ and $\mathbf{w}_t$, rather than separately performing these steps sequentially. We first solve for the following, where we assume that a additional rows have been added to $\mathbf{X}_{t-1}$ at indices where $\mathbf{S}_t = 0$:

$$(\mathbf{v}_t, \mathbf{w}_t) = \underset{\mathbf{v},\mathbf{w}}{\operatorname{argmin}} \|\mathbf{S}_t(\mathbf{X}_{t-1}\mathbf{v} - \mathbf{x}_t)\|_2^2 + \|\mathbf{S}_t(\mathbf{X}_{t-1}\mathbf{w} - \mathbf{y}_t)\|_2^2. \quad (2)$$

We note that points which have been added or removed are not considered – deleted trajectories should play no role in the estimation, whereas new trajectories do not yet have a subspace representation, i.e. it is unclear what their corresponding rows in $\mathbf{X}_{t-1}$ should be. Nonetheless, as illustrated in previous works [2, 1, 7], this estimation of $\mathbf{v}$ and $\mathbf{w}$ is an excellent approximation to the scenario of having complete knowledge of the removed coordinates and the subspace representation of the added rows.

We may now estimate each row of $\mathbf{X}_t$ that has not been removed, denoted $\mathbf{X}_t^j$ for row j, as:

$$\mathbf{X}_t^j = \underset{\mathbf{X}^j}{\operatorname{argmin}} \sum_{i=t^j}^{t} \lambda^{t-i}((\mathbf{v}_i^\intercal \mathbf{X}^j - x_i^j)^2 + (\mathbf{w}_i^\intercal \mathbf{X}^j - y_i^j)^2), \quad (3)$$

where t^j is the frame index when the row first arrived. Upon differentiating, we obtain the following linear system:

$$\sum_{i=t^j}^{t} \lambda^{t-i}(\mathbf{v}_i\mathbf{v}_i^\intercal + \mathbf{w}_i\mathbf{w}_i^\intercal)\mathbf{X}^j = \sum_{i=t^j}^{t} \lambda^{t-i}(x_i^j\mathbf{v}_i + y_i^j\mathbf{w}_i). \quad (4)$$

We slightly depart from Equation 1 by also updating the newly added rows, via a least-norm solution. This serves as a robust means of bootstrapping the subspace for added trajectories. In the next frame we use these rows as part of the subspace, while trajectories which have been removed are discarded from the subspace. This linear system may be recursively defined solely based on the current frame, hence we need only solve a 3×3 linear system for each row.

2.2. Conditions for Successful Tracking

There are two main conditions for subspace tracking to perform well: 1) the subspace should be sufficiently *incoherent*, and 2) the subspace should be slowly changing. We recall that the incoherence ϕ of a subspace $\mathbf{X}$ is:

$$\phi(\mathbf{X}) = \frac{d}{3} \max_i \|P_U \mathbf{e}_i\|_2^2, \quad (5)$$

where P_U is the subspace projection operator and $\mathbf{e}_i$ is the canonical basis vector: 1 at index i and 0 elsewhere. Empirically, we find that the shape subspace $\mathbf{X}$ is highly incoherent, that is, $\phi(\mathbf{X})$ tends to be very small – this corroborates

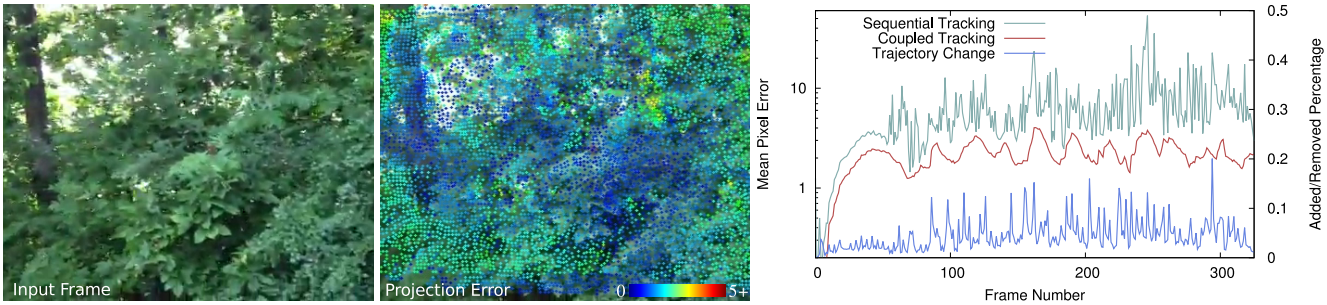

Figure 2. We show the performance of our subspace tracker in tracking exclusively background motion. Here we show a frame from a handheld camera video, the color-mapped projection error, and the mean projection error (corresponding to the left y-axis), comparing sequential coordinate updates versus coupling the pixel coordinates in the tracker. Note the stability of our method as trajectories are added/removed (corresponding to the right y-axis).

the results of [6], illustrating that in practice most subspaces should be incoherent. Regarding the second condition, we have the following result:

Theorem 2.1 *Let $\mathbf{X}_{t-1} \in \mathbb{R}^{d\times 3}$ and $\mathbf{X}_t \in \mathbb{R}^{(d+a)\times 3}$ be the ground truth shape subspaces composed of orthonormal columns, such that there exists a new points at time t, and without loss of generality, these correspond to the last a rows of $\mathbf{X}_t$. Furthermore, let $\hat{\mathbf{X}}_{t-1} \in \mathbb{R}^{(d+a)\times 3}$ have its first d rows be $\mathbf{X}_{t-1}$ and its last a rows be 0. Then:*

$$\left\|(\mathbf{I} - \hat{\mathbf{X}}_{t-1}\hat{\mathbf{X}}_{t-1}^{\intercal})\mathbf{X}_t\right\|_F^2 \leq \frac{3a\phi(\mathbf{X})}{d}. \tag{6}$$

Proof Let $\mathbf{S}$ be a diagonal sampling matrix where $\mathbf{S}_{i,i} = 1$ for $i \leq d$ and $\mathbf{S}_{i,i} = 0$ for $d < i \leq a$. Note that the first d rows of $\mathbf{X}_t$ span the same subspace as $\mathbf{X}_{t-1}$. Hence for a given column $\mathbf{u}$ of $\mathbf{X}_t$, we have:

$$\left\|(\mathbf{I} - \hat{\mathbf{X}}_{t-1}\hat{\mathbf{X}}_{t-1}^{\intercal})\mathbf{u}\right\|_2^2 = \tag{7}$$

$$\left\|(\mathbf{I} - \hat{\mathbf{X}}_{t-1}\hat{\mathbf{X}}_{t-1}^{\intercal})(\mathbf{S}\mathbf{u} - (\mathbf{I} - \mathbf{S})\mathbf{u})\right\|_2^2 = \sum_{i=d+1}^{d+a} u_i^2.$$

Applying this to each column of $\mathbf{X}_t$ and summing up, we obtain:

$$\left\|(\mathbf{I} - \hat{\mathbf{X}}_{t-1}\hat{\mathbf{X}}_{t-1}^{\intercal})\mathbf{X}_t\right\|_F^2 = \sum_{i=d+1}^{d+a} \|\mathbf{X}_t^i\|_2^2. \tag{8}$$

Note that $\phi(\mathbf{X})$ can also be expressed as: $\frac{d}{3}\max_i \|\mathbf{X}_t^i\|_2^2$, hence the bound follows. We also note that a similar (and tighter) bound can be obtained in projecting $\hat{\mathbf{X}}_{t-1}$ onto the orthogonal complement of $\mathbf{X}_t$. ∎

Hence, we see that projecting $\mathbf{X}_t$ onto the orthogonal complement of $\mathbf{X}_{t-1}$ – a common measure of proximity between subspaces [2] – is bound by the incoherence, so we impose no additional conditions on the subspace as traditional subspace tracking methods which deal with missing data. We simply require that the number of dimensions which are added and removed are bounded.

2.3. Tracking Camera Motion

We first demonstrate our method's capability in tracking a scene consisting of strictly background motion. We have taken a video from a camera phone of an outdoor scene consisting of varying degrees of motion, where the scene also contains different depth layers. At each frame, after updating the subspace, we project the trajectory coordinates onto the subspace and measure error by computing the distance from the ground truth point to its projected point, where we plot the mean error over all trajectories at each frame. In order to show the benefit of performing the subspace update at the pixel coordinate level, we also show the performance of subspace tracking through sequentially updating the subspace by the x coordinates, followed by the y coordinates.

See Figure 2 for the results, where we have set $\lambda = 0.98$. Note the benefit of coupling the x and y coordinates in the update, rather than processing them separately in sequence. Furthermore, note that our method remains quite stable when the percentage of trajectories being added and removed is rather large. Since the discount factor λ is somewhat high, this is indicative that the camera motion results in a slowly changing subspace, and demonstrates our ability to capture the overall motion.

3. Robust Dynamic Subspace Tracking

The energy defined by Equation 1 is inappropriate for handling foreground motion. Assuming that foreground trajectories are poorly modeled by the background subspace, their incorporation into the subspace update will pollute the background model. Hence we use a least absolute deviations formulation for updating the background, similar to the approach of [12]. This allows us to simultaneously update the background motion and estimate the foreground trajectories. We depart from [12] by enforcing sparsity at the coordinate level and use a more robust norm.

More specifically, at frame t suppose we have our subspace from the previous frame $\mathbf{X}_{t-1}$, and we want to obtain

a robust estimate of $\mathbf{v}_t$ and $\mathbf{w}_t$ from the new coordinates $\mathbf{x}_t$ and $\mathbf{y}_t$. We cannot assume that coordinates are mean-centered, as foreground trajectories will result in an erroneous center. To account for this, note that for uncentered trajectories the subspace is at most dimension 4 with the vector of all ones in its column space. Hence, throughout we assume that $\mathbf{X}_t \in \mathbb{R}^{d\times 4}$ with its last column comprised of all ones and $\mathbf{v}_t, \mathbf{w}_t \in \mathbb{R}^4$, where the last entries of $\mathbf{v}_t$ and $\mathbf{w}_t$ comprise the center, denoted $\mathbf{m}_t$.

For notational clarity and without loss of generality, at frame i assume that $\mathbf{X}_i$, $\mathbf{x}_i$, and $\mathbf{y}_i$ have already taken into account the sampling matrix $\mathbf{S}_i$. Let $\mathbf{c}_i = [\mathbf{v}_i \ \mathbf{w}_i]$ and for a given point index j let $\mathbf{p}_i^j = [x_i^j \ y_i^j]^\intercal$. Our least absolute deviations energy is defined as:

$$\mathbf{X}_t = \underset{\mathbf{X}}{\operatorname{argmin}} \sum_{i=1}^{t} \lambda^{t-i} \min_{\mathbf{c}_i} \left(\sum_{j=1}^{d} \|\mathbf{c}_i^\intercal \mathbf{X}^j - \mathbf{p}_i^j\|_2^p \right), \quad (9)$$

where $p < 1$ defines the choice of norm. Note that by coupling the x and y coordinates, this ensures a sparse solution at the pixel coordinate level. Furthermore, in using a l_p norm with $p < 1$ we can not only detect the foreground trajectories, but also reduce the impact of these foreground trajectories in estimating $\mathbf{v}_t$ and $\mathbf{w}_t$, giving us a more robust background representation.

Similar to the least squares formulation, we alternate between estimating the camera parameters $\mathbf{v}$ and $\mathbf{w}$, followed by updating the subspace. The group least absolute deviations energy for estimating the camera parameters at frame t is:

$$(\mathbf{v}_t, \mathbf{w}_t) = \underset{\mathbf{c}}{\operatorname{argmin}} \sum_{j=1}^{d} \|\mathbf{c}^\intercal \mathbf{X}_{t-1}^j - \mathbf{p}_t^j\|_2^p. \quad (10)$$

We minimize this energy by employing the alternating direction method of multipliers (ADMM) [3]. To this end, Equation 10 can be reformulated as the following energy by introducing a pair of sparsity vectors $\mathbf{s}_x$ and $\mathbf{s}_y$:

$$\min \sum_{j=1}^{d} \left(\sqrt{(\mathbf{s}_x^j)^2 + (\mathbf{s}_y^j)^2} \right)^p \quad \text{s. t.} \quad (11)$$
$$\mathbf{X}_{t-1}\mathbf{v} + \mathbf{s}_x = \mathbf{x}_t$$
$$\mathbf{X}_{t-1}\mathbf{w} + \mathbf{s}_y = \mathbf{y}_t.$$

The augmented Lagrangian of this formulation is then:

$$L(\mathbf{v}, \mathbf{w}, \mathbf{s}_x, \mathbf{s}_y, \boldsymbol{\alpha}_x, \boldsymbol{\alpha}_y) = \sum_{j=1}^{d} \left(\sqrt{(\mathbf{s}_x^j)^2 + (\mathbf{s}_y^j)^2} \right)^p$$
$$+\boldsymbol{\alpha}_x^\intercal(\mathbf{X}_{t-1}\mathbf{v} + \mathbf{s}_x - \mathbf{x}_t) + \boldsymbol{\alpha}_y^\intercal(\mathbf{X}_{t-1}\mathbf{w} + \mathbf{s}_y - \mathbf{y}_t)$$
$$+\frac{\rho}{2}(\|\mathbf{X}_{t-1}\mathbf{v} + \mathbf{s}_x - \mathbf{x}_t\|_2^2 + \|\mathbf{X}_{t-1}\mathbf{w} + \mathbf{s}_y - \mathbf{y}_t\|_2^2).$$

We minimize the Lagrangian through ADMM [3], where we use the group shrinkage operator for the l_p norm defined in [16], which works at the pixel coordinate level. Note that the shrinkage operator results in $\mathbf{v}$ and $\mathbf{w}$ influencing one another, where a foreground trajectory will result in both $\mathbf{s}_x$ and $\mathbf{s}_y$ to be of large magnitude.

In updating the subspace, we note that by fixing $\mathbf{v}$ and $\mathbf{w}$, minimizing Equation 9 also results in another least absolute deviations problem. However, its solution cannot be recursively defined, hence it would be prohibitively expensive to solve at each frame. We find that solving for the subspace via Equation 4, with the coordinates subtracted by the mean $\mathbf{m}_t$, still provides a robust and stable solution, and in practice is comparable to the solution of Equation 9. This is due to the per-row update scheme: trajectories which are in the background will receive the same update in this scenario, while foreground trajectories will receive a slightly noisier update. Note that if a trajectory in the foreground eventually stops moving in the scene it will gradually become a background trajectory through this update scheme, depending on the discount factor λ.

3.1. Trajectory-Level Segmentation

At each frame the likelihood that a given point belongs to the foreground or the background can be estimated by associating each point $\mathbf{p}_i^j$ the following score based on the point's projection onto the current subspace estimate:

$$s(\mathbf{p}_i^j) = \|\mathbf{c}_i^\intercal \mathbf{X}_i^j - \mathbf{p}_i^j\|_2. \quad (12)$$

However, using this directly to segment trajectories may result in spurious outliers: a trajectory which should be in the background may occasionally have points that are in the foreground due to poor optical flow estimation, while a trajectory which should be in the foreground may have points that are in the background due to insufficient foreground motion evidence. To cope with these cases, we employ a Markov Random Field (MRF) model to obtain a more robust segmentation in an online fashion by considering the full history of point scores for a given trajectory.

Let $T = \{t_j\}_{j=1}^m$ be the set of all trajectories up to and including frame t and let $l_j \in \{b, f\}$ be the background or foreground label associated with a trajectory t_j. The trajectory-level segmentation can be found at frame t by finding the labeling l with smallest MRF energy:

$$E(l) = \sum_{t_j \in T} \bar{\phi}_t(l_j) + \sum_{t_j, t_k \in T} \bar{\psi}_t(l_j, l_k), \quad (13)$$

where $\bar{\phi}_t(l_j)$ is the unary cost of assigning label l_j to trajectory t_j and $\bar{\psi}_t(l_j, l_k)$ is the pairwise cost of assigning labels to different trajectories. Both terms represent the average of

the $1 \ldots t$ per-frame trajectory costs:

$$\bar{\phi}_t(l_j) = \frac{1}{t}\sum_{i=1}^{t}\phi_i(l_j), \quad \bar{\psi}_t(l_j, l_k) = \frac{1}{t}\sum_{i=1}^{t}\psi_i(l_j, l_k), \quad (14)$$

where $\phi_i(l_j)$ and $\psi_i(l_j, l_k)$ are respectively the *frame-specific* unary and pairwise costs for t_j (and t_k for ψ_i) at frame i.

Unary Cost: To construct a reliable cost for label assignment, we observe that the distribution of scores from Equation 12 are consistently similar across all frames – it is low for points in the background and rapidly increases for points in the foreground. This motivates the following cost function:

$$\phi_i(l_j) = \delta_b + (\delta_f - \delta_b)e^{\frac{-\left(H(s(\mathbf{p}_i^j)) - \beta_k\right)^2}{2\sigma^2}}, \quad (15)$$

where $\delta_f = \delta(l_j = \text{f})$ and $\delta_b = \delta(l_j = \text{b})$ for delta function δ, $H(x)$ is x for $x > 0$ and 0 otherwise, and β_k is the k^{th} smallest score in frame i. Hence scores less than β_k are considered likely to be in the background; otherwise, likelihood is determined via a Gaussian centered at β_k with variance $\sigma = 10(\beta_k - \beta_0)$.

Pairwise Cost: The pairwise term encourages spatially adjacent trajectories to have the same label. For point $\mathbf{p}_i^j$ associated with trajectory t_j at frame i, let $\mathbf{p}_i^k$ be the point associated with trajectory t_k; then the pairwise term is simply: $\psi_N(t_i, t_j) = e^{\frac{-||\mathbf{p}_i^j - \mathbf{p}_i^k||^2}{2\sigma_p^2}}$, where $\sigma_p = 8$. Note that maintaining a running average in $\bar{\psi}_t$ encourages spatiotemporal coherency in label assignment.

At any frame a segmentation can be obtained by updating the unary and pairwise costs and minimizing the energy in Equation 13 via graph cuts [4]. We use the approach of [13] to efficiently compute the labeling in an online fashion, where after each frame, the previous labeling and graph are updated to reflect the current frame information and trajectories which are no longer present are removed.

4. Results

We demonstrate the benefits of our approach, denoted STUDD for *Subspace Tracking under Dynamic Dimensionality*, in several ways. We use the Berkeley motion segmentation dataset [5] to illustrate how our approach produces high-quality background estimation, scales to a large number of trajectories, and benefits dense trajectory segmentation for producing full image segmentation. Furthermore, we have produced a dataset intended to show how our method excels at background subtraction for long duration scenes containing very complex camera motion and foreground.

Name	RANSAC	LRGS	STUDD
people1	0.992	0.999	**0.999**
people2	0.989	0.991	**0.999**
cars1	0.875	0.585	**0.944**
cars2	0.960	0.973	**0.987**
cars3	0.933	0.984	0.976
cars4	0.983	0.994	0.992
cars5	0.967	0.982	**0.989**
cars6	0.973	0.956	**0.998**
cars7	0.980	0.989	**0.993**
cars8	0.935	0.655	**0.989**
cars9	0.953	0.763	0.952
cars10	0.810	0.604	**0.969**
marple1	0.775	0.765	**0.973**
marple2	0.832	0.844	**0.933**
marple3	0.840	0.819	**0.927**
marple4	0.820	0.819	**0.858**
marple5	0.919	0.922	**0.941**
marple6	0.668	0.668	**0.714**
marple7	0.746	0.795	**0.848**
marple8	0.835	0.862	**0.921**
marple9	0.472	0.506	**0.557**
marple10	0.984	0.982	0.930
marple11	0.978	0.982	**0.985**
marple12	0.846	0.862	**0.944**
marple13	0.809	0.841	**0.894**
tennis	0.968	0.984	0.983

Table 1. A comparison between our method (STUDD) and the methods of [20] and [8], evaluated on the benchmark dataset of [5] for dense tracks. Here we list the average background f-measure.

Implementation Details For all examples, we have set the discount factor to $\lambda = .98$, the l_p norm to $p = .1$, and the threshold index k in β_k to 20% of the number of trajectories at the given frame. We found our method to be quite robust to variability in these parameter settings. We produce labels for trajectories only when their trajectory duration is at least 4, as we find their subspace representation to be stable beginning at this duration.

4.1. Berkeley Motion Segmentation Dataset

For this dataset, we first compare to factorization-based sliding window methods [20, 8], denoted RANSAC and LRGS respectively. A window size of 4 was used in order to perform a fair comparison with our method. The optical flow tracker of [21], with a subsampling factor of 2 was used to obtain dense point trajectories. To measure segmentation quality the background f-measure averaged over all ground truth frames was used and is defined as $f = \frac{2PR}{P+R}$ where P and R respectively represent the precision and recall of classifying background.

See Table 1 for the results. As shown, in general we obtain superior results across the dataset at comparable running times – see Table 2 for computational timings. We

Name	RANSAC	LRGS	STUDD
Time (s)	0.788	1.061	1.833

Table 2. The average per-frame running time in seconds for the results shown in Table 1.

Dataset	F-Measure		Time (s)	
	HOM	STUDD	HOM	STUDD
moseg	0.938	0.885	10.778	0.055
moseg*	0.937	0.911	10.726	0.052

Table 3. Comparison between [18] (HOM) and our method using averaged f-measures on the marple and tennis datasets [5] in the first row, and the same set with marple9 excluded in the second row. We obtain competitive results with 2 orders of magnitude improvement in running times, measured as average time per frame.

perform particularly well for longer sequences, i.e. the marple videos, demonstrating the long-term stability of our approach, whereas [20, 8] are unable to discern the background due to insufficient evidence in the temporal window.

We also compared our method to the high-order spectral clustering approach of [18]. As [18] produces a full motion segmentation, we take the background in each frame as the label which comprises the most trajectories. Due to scalability limitations of [18], we generate a sparse trajectories set via [21] using a subsampling factor of 8 for tracking. See Table 3 for the results. We obtain competitive, albeit slightly worse results. This is typically due to when a person begins to move in later frames, since our approach will initially classify the trajectory as background when they are not moving (i.e. marple9). However, note that our method is roughly 2 orders of magnitude faster, and does not require processing the trajectories in batch.

To demonstrate the impact of using sparse trajectories for dense image segmentation, we have taken the results of [18] at a trajectory subsampling factor of 8 and compared to our method for a subsampling factor of 2, using the method of [17] to obtain a full image segmentation. See Figure 3 for the results. We observe that our dense trajectory labeling produces superior image segmentation results compared to a sparser labeling, where there may be insufficient evidence for the foreground.

4.2. Long-Term Motion

Last, we have produced a dataset consisting of a long and complex video sequence. We have taken a nearly 2000 frame single camera shot sequence from the film "Magnolia", and labeled 22 of these images for ground truth. We compare our method to [20, 8] – see Table 4 for average f-measure results and Figure 4 for qualitative results. In order to test the sensitivity to window size, we evaluate these methods with windows of 4, 10, and 15. Interestingly, the best window size need not be the largest possible, demonstrating the sensitivity to the choice of a temporal window. Furthermore, we note that a larger window size

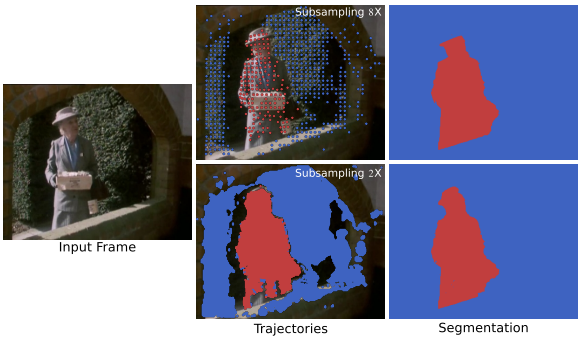

Figure 3. The method of [17] (top row) is applied to 8x subsampled segmented trajectories from [18] as compared with the 2x subsampled segmented trajectories from our method (bottom row). Note the utility of dense trajectories in supporting appearance-based dense image segmentation. The background f-measure for [18] is 0.990, and is 0.995 for STUDD.

STUDD	RANSAC			LRGS		
	w=4	w=10	w=15	w=4	w=10	w=15
0.877	0.841	0.843	0.827	0.857	0.868	0.791

Table 4. A comparison of the "Magnolia" sequence, in average f-measure, where for RANSAC and LRGS the window sizes are set to 4, 10, and 15. Note the sensitivity to the window size, whereas STUDD naturally adapts to the dynamic trajectories.

results in fewer trajectories labeled per-frame. By maintaining a continuously-changing subspace we can achieve stable and denser background segmentation results.

5. Conclusions

We have presented a method for segmenting foreground and background trajectories from moving camera videos via subspace tracking under changing dimensionality. Our method is highly scalable in terms of the number of trajectories and performs segmentation in an online manner, hence it can work alongside dense optical flow methods. A limitation of our approach is that certain scenes can violate the orthographic camera assumption, where our subspace will fail to capture the full background. We plan to investigate our subspace tracker in the context of multiple types of motion [25], in order to extend our method to arbitrary motion segmentation.

Acknowledgements

We thank Kevin Johnson for his help in early stage development and the reviewers for their useful feedback.

References

[1] L. Balzano, R. Nowak, and B. Recht. Online identification and tracking of subspaces from highly incomplete information. In *Allerton Conference on Communication, Control, and Computing*, pages 704–711. IEEE, 2010. 2, 3

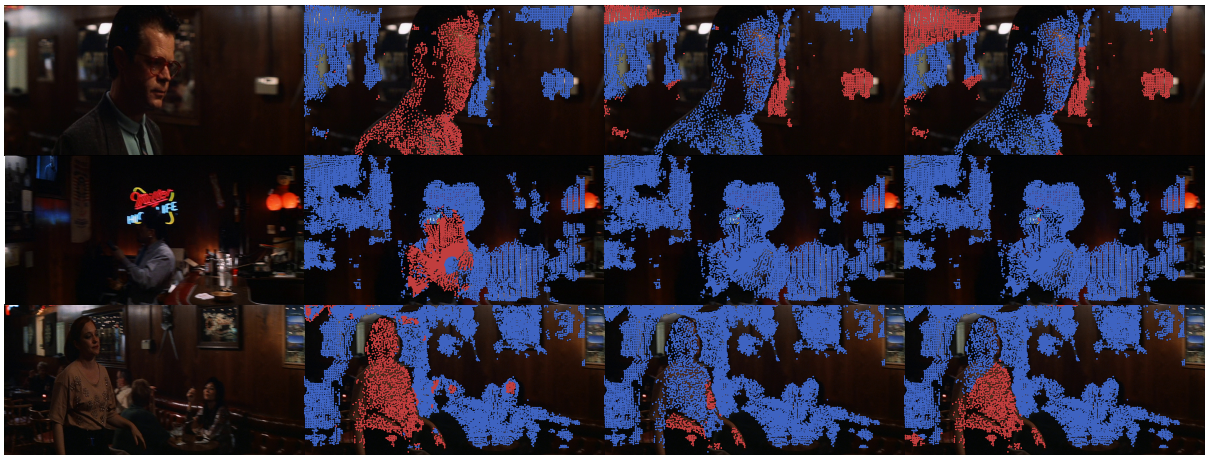

Figure 4. Background segmentation applied to a video sequence from the film "Magnolia". Our method is shown in the second column, RANSAC [20] in the third column, and LRGS [8] in the last column, both using a window of size 4.

[2] L. Balzano, B. Recht, and R. Nowak. High-dimensional matched subspace detection when data are missing. In *International Symposium on Information Theory Proceedings (ISIT)*, pages 1638–1642. IEEE, 2010. 2, 3, 4

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011. 5

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004. 6

[5] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, pages 282–295. Springer, 2010. 1, 2, 6, 7

[6] E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009. 4

[7] Y. Chi, Y. C. Eldar, and R. Calderbank. Petrels: Subspace estimation and tracking from partial observations. In *ICASSP*, pages 3301–3304. IEEE, 2012. 1, 2, 3

[8] X. Cui, J. Huang, S. Zhang, and D. N. Metaxas. Background subtraction using low rank and group sparsity constraints. In *ECCV*, pages 612–625. Springer, 2012. 2, 6, 7, 8

[9] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. In *ECCV*, pages 751–767. Springer, 2000. 1

[10] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, pages 2790–2797. IEEE, 2009. 2

[11] A. Elqursh and A. Elgammal. Online moving camera background subtraction. In *ECCV*, pages 228–241. Springer, 2012. 1, 2

[12] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *CVPR*, pages 1568–1575. IEEE, 2012. 1, 2, 4

[13] P. Kohli and P. H. Torr. Efficiently solving dynamic markov random fields using graph cuts. In *ICCV*, volume 2, pages 922–929. IEEE, 2005. 6

[14] S. Kwak, T. Lim, W. Nam, B. Han, and J. H. Han. Generalized background subtraction based on hybrid inference by belief propagation and bayesian filtering. In *ICCV*, pages 2174–2181. IEEE, 2011. 2

[15] J. Lezama, K. Alahari, J. Sivic, and I. Laptev. Track to the future: Spatio-temporal video segmentation with long-range motion cues. In *CVPR*, pages 3369–3376. IEEE, 2011. 1

[16] G. Marjanovic and V. Solo. On l_q optimization and matrix completion. *IEEE Transactions on Signal Processing*, 60(11):5714–5724, 2012. 5

[17] P. Ochs and T. Brox. Object segmentation in video: a hierarchical variational approach for turning point trajectories into dense regions. In *ICCV*, pages 1583–1590. IEEE, 2011. 7

[18] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *CVPR*, pages 614–621. IEEE, 2012. 1, 2, 7

[19] S. R. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *CVPR*, pages 1–8. IEEE, 2008. 2

[20] Y. Sheikh, O. Javed, and T. Kanade. Background subtraction for freely moving cameras. In *ICCV*, pages 1219–1225. IEEE, 2009. 1, 2, 6, 7, 8

[21] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *ECCV*, pages 438–451. Springer, 2010. 1, 6, 7

[22] M. Tao, J. Bai, P. Kohli, and S. Paris. Simpleflow: A non-iterative, sublinear optical flow algorithm. In *Computer Graphics Forum*, volume 31, pages 345–353, 2012. 1

[23] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *IJCV*, 9(2):137–154, 1992. 1, 2, 3

[24] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, pages 1–8. IEEE, 2007. 1, 2

[25] J. Yan and M. Pollefeys. A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate. In *ECCV*, pages 94–106. Springer, 2006. 2, 7