Big Data and Information Analytics

http://www.aimspress.com/journal/BDIA

BDIA, 3(2): 1–23.
DOI:10.3934/bdia.2018004
Received: 08 October 2018
Accepted: 16 October 2018
Published: 09 November 2018

Research article

Regularized kernel matrix decomposition for thermal video multi-object detection and tracking

Ioannis D. Schizas[1,*], Vasileios Maroulas[2] and Guohua Ren[1]

[1] Department of Electrical Engineering, University of Texas at Arlington, Arlington, TX, 76019, USA

[2] Department of Mathematics, University of Tennessee at Knoxville, TN, 37996, USA

* **Correspondence:** schizas@uta.edu; Tel: +18172723467.

Abstract: This paper derives a novel algorithm for joint detection and tracking of multiple moving objects in thermal videos. The problem of determining multiple objects in a frame sequence is formulated as the task of factorizing a properly defined kernel covariance matrix into sparse factors. The support of these factors will point to the indices of the pixels that form each object. A coordinate descent approach is utilized to determine the sparse factors, and extract the object pixels. A centroid pixel is estimated for each object which is subsequently tracked via Kalman filtering. A novel interplay between the sparse kernel covariance factorization scheme along with Kalman filtering is proposed to enable joint object detection and tracking, while a divide and conquer strategy is put forth to reduce computational complexity and enable efficient tracking. Extensive numerical tests on both synthetic data and thermal video sequences demonstrate the effectiveness of the novel approach and superior tracking performance compared to existing alternatives.

Keywords: Multi-object tracking; kernel learning; sparse matrix decomposition; Kalman filtering; thermal video

1. Introduction

Tracking of moving objects in videos is a fundamental problem in computer vision, and a plethora of approaches have been put forth to address the tracking problem using RGB (red, green, blue) cameras, see e.g., [4, 15, 19, 29]. Nonetheless, there are many challenges that still need to be addressed such as object/camera motion, varying appearances of the objects, different illumination conditions and occlusions. Further, the presence of a changing number of multiple objects in a frame sequence makes tracking still an extremely challenging problem.

Recently, uncooled thermal sensors have become affordable and achieve improved resolution capability [7]. Further, there is an increasing interest in utilizing thermal sensors to facilitate vision tasks, such as face recognition, and human-robot interaction, [5, 31]. Moreover, in moving object tracking applications like outdoor surveillance, where usually the background temperature is largely different from the moving objects, thermal imaging becomes crucial in detecting and tracking those objects that radiate thermal energy such as humans, animals or vehicles. It is noteworthy that thermal imaging is not affected by shadow and light illumination, which normally is a bottleneck for RGB cameras, rendering it more suitable for moving object tracking in both daytime and nighttime [22]. In [34], a sparse representation technique is utilized to extract features for the video objects. Compressed feature vectors are first obtained by the sparse representation technique, then a Bayes binary classifier is designed to track the object. A subspace model is learned in [3] to model the object of interest in videos, though it is an offline tracking approach. [22] proposed to use a particle filter to track object motion features preprocessed from the Wigner distribution. Support vector machines and Kalman filtering are combined toward identifying and tracking pedestrians in [35]. In [36], a scheme is developed to detect the pedestrian head, and pedestrian legs which are later tracked by local search. The aforementioned approaches are limited in the sense that they cannot jointly detect and track multiple objects, while they have to impose certain pixel intensity thresholding or statistical/structural assumptions for the objects present.

Thermal cameras output corresponds usually to gray scale imaging, which results in a lower data processing complexity, in contrast to the triple data load produced by RGB cameras. Also, there are some research efforts that propose fusion of thermal and RGB visible data, e.g., [6, 8, 11]. The work in [6] relies on the contour saliency map, to fuse together object locations and contours from both thermal and color sensors and eventually extract the object silhouette features, thus obtaining improved tracking performance. However, the method is computationally expensive since it aims at constructing a complete object contour. In [11], data fusion is implemented to fuse thermal and visible data, resulting in an illumination-invariant face image. In the latter work, decision fusion combines the matching score generated from individual face recognition models. Indeed, modal fusion enables better tracking performance since more data are utilized. However, in many practical scenarios where only one of the imaging modalities is available to use, tracking systems can benefit from the utility of thermal data due to the computational cost savings introduced.

In this paper we propose a novel approach to perform joint detection and tracking of multiple moving objects in thermal videos. Having no prior information on the objects present in the video frames, the object detection problem is formulated as the problem of factorizing a pertinent kernel covariance matrix into sparse factors. The pixels consisting of an object will be determined by estimating the support of these sparse factors and employing clustering of the nonzero entries to separate individual objects. Each object will be tracked via an alternative implementation of Kalman filtering, which has been used extensively in target tracking using sensor networks [16, 20, 24–26], and the proposed kernel matrix sparse factorization scheme. The idea of sparse covariance factorization was first explored in [28] to determine informative sensors in a network. However, in [28], linear data models are considered which is not the case in the video object tracking setting considered here. Further, the approach in [28] focuses in detecting stationary and static sources, whereas in the proposed work here nonlinear inter-pixel correlations are extracted and utilized along with multiple object dynamics to achieve accurate multi-object tracking. Coordinate descent techniques [2, 32], are employed to decompose the formulated kernel covariance matrix in a recursive way. Moreover, the implementation of a computationally efficient 'divide-and-conquer' based scheme mitigates the high computational burden of factorizing large kernel covariance matrices resulting from frames having large dimensions and acquired at fast rates. The Kalman filter [17] is further combined with the aforementioned sparse kernel covariance factorization scheme to allow precise tracking of the detected objects in videos.

The paper is structured as follows. The problem setting is introduced in Sec. II. The problem of clustering pixels according to the object they belong to is formulated as a sparse kernel covariance factorization problem and is detailed in Sec. III. A computational efficient method is derived based on coordinate descent approaches. Kalman filtering is employed to track the estimated centroid pixel of the detected objects in Sec. IV. A divide and conquer implementation is described in Sec. V, along with the interplay of Kalman filtering and our proposed sparse kernel covariance factorization algorithm. Numerical results in different scenarios in Secs. VI, VII, VIII are carried out to corroborate the effectiveness of the proposed multiple object tracking scheme in thermal videos, while demonstrating the superiority of the novel approach over existing alternatives.

2. Problem setting and preliminaries

Consider a sequence of frames forming a video in which the frames contain multiple nonstationary/moving objects of interest that need to be detected/identified and tracked. Let $\mathbf{F}_t$ be the frame of the available video sequence at time instant t of dimensions $f_x \times f_y$ whose entries are real numbers, i.e., $\mathbf{F}_t \in \mathbb{R}^{f_x \times f_y}$. Further, let $x_t^{m,n}$ denote the pixel intensity of the (m, n)-th pixel of frame $\mathbf{F}_t$ at time instant t where $m = 1, \ldots, f_x$ and $n = 1, \ldots, f_y$. For simplicity in exposition $\mathbf{x}_t \in \mathcal{R}^{p \times 1}$, with $p = f_x \cdot f_y$ denotes a vector that contains all the pixels of frame $\mathbf{F}_t$ placed in there after traversing them from top to bottom and left to right. For the sake of simplicity later on we will omit the $\mathbf{f}$ index in $\mathbf{x}_t$.

There is an *unknown* number of objects in the video that we are interested in tracking, and M denotes the maximum number of objects that can be present in a frame. Let $\mathcal{P}_m^t$ denote the set of pixel indices corresponding to the mth object at time instant t, i.e., $\mathcal{P}_m^t := \{[x_{m,1,t}, y_{m,1,t}], \ldots, [x_{m,N_m,t}, y_{m,N_m,t}]\}$ indicate the coordinates of the pixels of the mth object at time instant t, with N_m indicating the number of pixels of object m. The pixels corresponding to an object at time instant t, say $\mathcal{P}_m^t$, are not known. In order to model the movement of each of the objects we will focus on how the coordinates of the centroid pixel of each object evolve in time. The centroid pixel for the mth object at time instant t is defined as

$$\mathbf{c}_m^t := \lfloor N_m^{-1} \sum_{i=1}^{N_m} [x_{m,n,t}, y_{m,n,t}] \rfloor,$$

with $\lfloor\rfloor$ denoting the floor operator. The centroid pixel intuitively describes the center point of the mth moving object.

When the video sequence is acquired at a high frame rate, it can be assumed that the objects' centroid pixels move from frame to frame according to a constant velocity moving model [1]. The velocity can be kept constant for a certain number of frames and then changed if necessary. Specifically, the m-th object's state vector is denoted as $\mathbf{s}_m(t) := [(\mathbf{c}_m^t)^T, \mathbf{v}_m^t]$, where $\mathbf{v}_m^t$ is a 2×1 vector that contains the velocity across the horizontal and vertical axis. The state vector $\mathbf{s}_m(t)$ is assumed to evolve according to the following model:

$$\mathbf{s}_m(t) = \mathbf{A}\mathbf{s}_m(t-1) + \mathbf{u}_m(t), \quad m = 1, \ldots, M \tag{2.1}$$

where $\mathbf{A} \in \mathcal{R}^{4\times4}$ is the state transition matrix, while $\mathbf{u}_m(t)$ denotes zero-mean Gaussian noise with covariance $\mathbf{\Sigma}_u$. The matrices $\mathbf{A}$ and $\mathbf{\Sigma}_u$ have the following structure (e.g., see [1])

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{2.2}$$

$$\mathbf{\Sigma}_u = \sigma_u^2 \begin{bmatrix} (\Delta T)^3/3 \cdot \mathbf{I}_2 & (\Delta T)^2/2 \cdot \mathbf{I}_2 \\ (\Delta T)^2/2 \cdot \mathbf{I}_2 & \Delta T \cdot \mathbf{I}_2 \end{bmatrix}, \tag{2.3}$$

where ΔT corresponds to the inter-frame time interval, σ_u^2 is a nonnegative constant controlling the variance of the noise entries in $\mathbf{u}_m(t)$, while $\mathbf{I}_2$ denotes the 2×2 identity matrix. The pixel coordinates $[x_{m,1,t}, y_{m,1,t}]$ take integer values, however the state noise $\mathbf{u}_m(t)$ is assumed Gaussian causing the state to take real values. Though, we can control the state noise standard deviation such that $3\sigma_u$ ($3 - \sigma$ bounds) is equal to a small number corresponding to the number of pixels that the state is deviating from the constant velocity movement. Since the noise will take values within the interval $[-3\sigma_u, 3\sigma_u]$ with probability 99.7%, the state noise can model at an acceptable level of accuracy the movement of the video objects from frame to frame despite the fact that the centroid $\mathbf{c}_m(t)$ can have real values.

2.1. Kernel-based object pixel correlations

As stated earlier the pixels corresponding to an object are unknown, thus it is essential to identify the objects before attempting to track them. To cluster the pixels of interest according to the object they belong to, we will utilize statistical correlations that pixels belonging to the same object exhibit. Pixels of an object are expected to have similar intensity (different from the background pixels) which subsequently makes them correlated.

Pixels belonging to the same object exhibit nonlinear dependencies in general in the sense of being nonlinearly correlated [14], thus employing a linear covariance matrix will not identify correlated components. To this end, we account for the nonlinear inter-pixel correlations of frame $\mathbf{F}_t$, namely $\mathbf{x}_t := \text{vec}(\mathbf{F}_t)$ where vex(.) is the vectorization operator, by utilizing nonlinear mappings $\boldsymbol{\phi}_x(\mathbf{x}_t)$ that are applied row-wise across the entries of $\mathbf{x}_t$ and map each pixel in $\mathbf{x}_t$, in a higher dimensional space where linear correlations can be exploited. Specifically, the mapping $\boldsymbol{\phi}_x$ results a $f_x f_y \times D$ matrix

$$\boldsymbol{\phi}_x(\mathbf{x}_t) := [\boldsymbol{\phi}_x(\mathbf{x}_t(1)), \ldots, \boldsymbol{\phi}_x(\mathbf{x}_t(f_x f_y))]^T, \tag{2.4}$$

where D corresponds to the dimensionality of the transformed pixel vector $\boldsymbol{\phi}_x(\mathbf{x}_t(i))$ for $i = 1, \ldots, f_x f_y$.

The nonlinear mapping $\boldsymbol{\phi}_x$ should be selected such that the covariance matrix of the transformed frames exhibits a block diagonal structure. For example Figure 1 depicts the nonzero entries (black dots) of a kernel (the Gaussian kernel will be used) covariance matrix obtained from a sequence of frames in which a single white object moves within black background. Clearly, the covariance matrix has a block diagonal structure after proper permutation of the rows and columns that contain the nonzero entries.

After properly selecting a kernel (see details later on) the covariance of the transformed data $\boldsymbol{\phi}_x(\mathbf{x}_t)$ can be written as

$$\begin{aligned}&\mathbb{E}[(\boldsymbol{\phi}_x(\mathbf{x}_t) - \mathbb{E}[\boldsymbol{\phi}_x(\mathbf{x}_t)])(\boldsymbol{\phi}_x(\mathbf{x}_t) - \mathbb{E}[\boldsymbol{\phi}_x(\mathbf{x}_t)])^T]\\ &= \mathbf{P}_r \mathrm{bdiag}(\mathbf{B}_{1,t}, \mathbf{B}_{2,t}, \ldots, \mathbf{B}_{M,t})\mathbf{P}_c,\end{aligned} \tag{2.5}$$

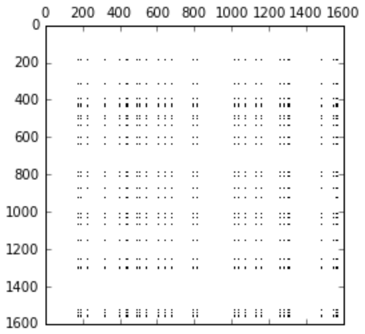

Figure 1. Sparse structure of a kernel covariance matrix.

where bdiag() refers to a block diagonal matrix, with $\mathbf{B}_{m,t}$ denoting the mth diagonal block of size $N_m \times N_m$ indicating how the N_m pixels of object m are correlated, while pixels belonging to different component are assumed to be uncorrelated. It is also possible that pixels from different objects may be correlated if the objects have similar pixel intensity or texture. In that case one diagonal block, namely $\mathbf{B}_{j,t}$, may be associated with more than one objects and we will see later on how the pixels can be separated. Further, matrices $\mathbf{P}_r$ and $\mathbf{P}_c$ corresponds to an arbitrary unknown permutation of the rows and columns.

The first challenge will be to locate the pixels of each object, which pertains to identifying where the entries of each of the M diagonal blocks are located in the transformed covariance matrix in (2.5), which boils down to estimating the size of each diagonal block N_m, as well as the indices of the pixels that belong to the mth diagonal block. In the following section we will formulate this as a sparse matrix factorization problem, while properly selecting the kernel to induce a covariance matrix with block diagonal structure. Then, once the pixels of an object have been determined we will proceed with tracking the estimated centroid of each of the objects.

3. Multi-object pixel clustering

3.1. Kernel covariance estimation

In order to estimate the covariance matrix of the transformed data in (2.5) we will rely on sample-averaging, where after applying a proper transformation to the pixel vectors to obtain $\boldsymbol{\phi}_x(\mathbf{x}_t)$ we estimate the covariance of the transformed data as

$$\begin{aligned}\hat{\boldsymbol{\Sigma}}_{\phi_x,t} &= \frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}(\boldsymbol{\phi}_x(\mathbf{x}_\tau) - \bar{\boldsymbol{\phi}}_{x,t}) \cdot (\boldsymbol{\phi}_x(\mathbf{x}_\tau) - \bar{\boldsymbol{\phi}}_{x,t})^T\\ &= \frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}[\boldsymbol{\phi}_x(\mathbf{x}_\tau)\boldsymbol{\phi}_x^T(\mathbf{x}_\tau) - \boldsymbol{\phi}_x(\mathbf{x}_\tau)\bar{\boldsymbol{\phi}}_{x,t}^T\\ &- \bar{\boldsymbol{\phi}}_{x,t}\boldsymbol{\phi}_x^T(\mathbf{x}_\tau) + \bar{\boldsymbol{\phi}}_{x,t}\bar{\boldsymbol{\phi}}_{x,t}^T],\end{aligned} \tag{3.1}$$

where $\bar{\boldsymbol{\phi}}_{x,t} := F^{-1}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_\tau)$ corresponds to the sample-average estimate of the mean of the transformed frame pixels, while F corresponds to the number (different than $\mathbf{F}_t$ that denotes the frame at time t) of frames that the objects are virtually stationary and occupy the same area in the frames. The higher the sampling rate is, the larger F can be chosen while assuming the objects are stationary within the time interval $[(t-1)F+1, tF]$.

Applying the kernel trick (assuming a proper nonlinear mapping $\boldsymbol{\phi}_x(\cdot)$ is used; see details in [12, 18, 27, 33]) the inner products involved in calculating the entries of $\boldsymbol{\phi}_x(\mathbf{x}_\tau)\boldsymbol{\phi}_x^T(\mathbf{x}_\tau)$ can be found using a proper positive definite kernel function $K(\mathbf{x}_1(i), \mathbf{x}_2(j))$ whose two arguments correspond to pixels i and j from frame pixel vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively, i.e., the kernel trick implies that the inner product $\langle\boldsymbol{\phi}_x(\mathbf{x}_1(i)), \boldsymbol{\phi}_x(\mathbf{x}_2(j))\rangle$ can be evaluated from a proper scalar kernel function $K(\mathbf{x}_1, \mathbf{x}_2)$. Utilizing this property the first term in Eq (3.1) can be rewritten as

$$\frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_\tau)\boldsymbol{\phi}_x^T(\mathbf{x}_\tau) = \frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\mathbf{K}(\mathbf{x}_\tau, \mathbf{x}_\tau), \tag{3.2}$$

where $\mathbf{K}$ is a $f_xf_y \times f_xf_y$ matrix whose (i, j)-th entry is given as $[\mathbf{K}]_{i,j} = K(\mathbf{x}_\tau(i), \mathbf{x}_\tau(j))$. Similarly, the second and third summation terms in Eq (3.1) can be expressed as

$$\begin{aligned}&\frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_\tau)\bar{\boldsymbol{\phi}}_{x,t}^T\\ &= \frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_\tau)\frac{1}{F}\Sigma_{\tau'=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x{}^T(\mathbf{x}_{\tau'})\\ &= \frac{1}{F}\cdot\frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\Sigma_{\tau'=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_\tau)\boldsymbol{\phi}_x{}^T(\mathbf{x}_{\tau'})\\ &= \frac{1}{F^2}\Sigma_{\tau,\tau'=(t-1)\cdot F+1}^{t\cdot F}\mathbf{K}(\mathbf{x}_\tau, \mathbf{x}_{\tau'}),\end{aligned} \tag{3.3}$$

while the fourth summation term in Eq (3.1) gives

$$\begin{aligned}&\frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\bar{\boldsymbol{\phi}}_{x,t}\bar{\boldsymbol{\phi}}_{x,t}^T\\ &= \frac{1}{F^2}\Sigma_{\tau'=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x(\mathbf{x}_{\tau'})\Sigma_{\tau''=(t-1)\cdot F+1}^{t\cdot F}\boldsymbol{\phi}_x^T(\mathbf{x}_{\tau''})\\ &= \frac{1}{F^2}\Sigma_{\tau''=(t-1)\cdot F+1}^{t\cdot F}\ \mathbf{K}(\mathbf{x}_{\tau'}, \mathbf{x}_{\tau''}).\end{aligned} \tag{3.4}$$

Notice that the matrix in Eq (3.4) is the same with the one obtained in (3.3), thus the covariance matrix of the transformed data can be calculated with the aid of the kernel function $K(\cdot, \cdot)$ as follows

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{\phi}_x,t} &= \frac{1}{F}\Sigma_{\tau=(t-1)\cdot F+1}^{t\cdot F}\mathbf{K}(\mathbf{x}_\tau, \mathbf{x}_\tau)\\ &\quad - \frac{1}{F^2}\Sigma_{\tau',\tau''=(t-1)\cdot F+1}^{t\cdot F}\ \mathbf{K}(\mathbf{x}_{\tau'}, \mathbf{x}_{\tau''}).\end{aligned} \tag{3.5}$$

A kernel utilized in image pixel classification successfully [9, 23], is the Gaussian radial basis function (RBF) in which the (i, j) entry of matrix $\mathbf{K}$ used earlier can be expressed as

$$K(\mathbf{x}_\tau(i), \mathbf{x}_\tau(j)) = \exp\left(-\frac{(\mathbf{x}_\tau(i) - \mathbf{x}_\tau(j))^2}{2\sigma^2}\right), \tag{3.6}$$

where the variance σ^2 is a crucial parameter that controls the degree of inter-pixel correlation. Details on how to select this parameters will be given in Sec. 6.

3.2. Kernel covariance sparse factorization

Next, we will try to determine sparse factors $\mathbf{m}_1, \ldots, \mathbf{m}_M$ such that $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}_x,t} \approx \sum_{m=1}^{M} \mathbf{m}_m \mathbf{m}_m^T = \mathbf{M}_t \mathbf{M}_t^T$, while the support of each of the factors $\mathbf{g}_m$ will indicate the indices of the entries belonging to a block of correlated pixels in $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\phi}_x,t}$ that subsequently belong to the same object. The idea of utilizing sparse matrix decomposition to identify correlated data was first proposed in [28] and here it is generalized under the realm of kernel-based nonlinear data transformations.

Single Object:

We start with the case where there is only one moving object in the frames of the video sequence. A standard least-squares based matrix decomposition scheme would minimize the Frobenius norm-based cost $||\boldsymbol{\Sigma}_{\boldsymbol{\phi}_x,t} - \mathbf{M}_t \mathbf{M}_t^T||_F^2$ with respect to the factor estimates $\mathbf{M}_t \in \mathcal{R}^{p\times 1}$. However, such a formulation does not take into account the sparse structure of $\mathbf{M}_t$. To this end, the following minimization framework is proposed:

$$\hat{\mathbf{M}}_t := \arg\min_{\mathbf{M}_t} ||\boldsymbol{\Sigma}_{\boldsymbol{\phi}_\mathbf{x},t} - \mathbf{M}_t \mathbf{M}_t^T||_F^2 + \lambda ||\mathbf{M}_t||_1, \tag{3.7}$$

where the norm-one term $||\cdot||_1$ is utilized to induce sparsity in the column vector $\mathbf{M}_t$, see e.g., [30, 37], in the column of $\mathbf{M}_t$ whose support will point to those pixels in a collection of F frames that contain the object of interest within interval $[(t-1)F + 1, t\cdot F]$. The parameter λ is the sparsity controlling coefficient that determines the number of zeros in $\mathbf{M}_t$, i.e., the larger λ is, the more zero entries will be contained in the optimal solution $\hat{\mathbf{M}}_t$.

The cost in Eq (3.7) is nonconvex with respect to (wrt) $\mathbf{M}_t$. To overcome this obstacle an iterative minimization scheme is derived next using coordinate descent strategies [2]. The cost in Eq (3.7) is minimized recursively wrt one entry of $\mathbf{M}_t$, namely $\mathbf{M}_t(j)$, while keeping all other entries in $\mathbf{M}_t$ fixed to their latest updates.

Minimization of the cost in Eq (3.7) wrt $\mathbf{M}_t(j)$ while fixing the remaining variables to their latest update during coordinate cycle k gives the following solution for updating $\hat{\mathbf{M}}_t^k(j)$:

$$\begin{aligned}\hat{\mathbf{M}}_t^k(j) = \arg\min_{\mathbf{M}_t(j)} 2 \cdot &\sum_{\mu=1,\mu\neq j}^{p} [\boldsymbol{\Sigma}_{\boldsymbol{\phi}_x,t}(j,\mu) - \mathbf{M}_t(j)\hat{\mathbf{M}}_t^{k-1}(\mu)]^2 \\ &+ \lambda|\mathbf{M}_t(j)| + [\boldsymbol{\Sigma}_{\boldsymbol{\phi}_x,t}(j,j) - \mathbf{M}_t^2(j)]^2.\end{aligned} \tag{3.8}$$

Discarding the terms that do not depend on $\mathbf{M}_t(j)$ and applying proper algebraic manipulations, the cost in Eq (3.8) can be rewritten as:

$$\begin{aligned}J^k(j) = &(\mathbf{M}_t(j))^4 + \lambda|\mathbf{M}_t(j)| \\ &+ (\mathbf{M}_t(j))^2[2\sum_{\mu=1,\mu\neq j}^{p} [\hat{\mathbf{M}}_t^{k-1}(\mu)]^2 - 2\delta^k(j,j)] \\ &- \mathbf{M}_t(j)[4\sum_{\mu=1,\mu\neq j} \delta^k(j,\mu)\hat{\mathbf{M}}_t^{k-1}(\mu)]\end{aligned} \tag{3.9}$$

where $\delta^k(j,\mu) := \boldsymbol{\Sigma}_{\boldsymbol{\phi}_x,t}(j,\mu)$ for $j,\mu = 1,\ldots,p$. Given the most recent update $\hat{\mathbf{M}}_t^{k-1}$ from coordinate cycle $k-1$, as shown in Appendix A, the update for $\hat{\mathbf{M}}_t^k(j)$ will be the value which achieves the minimum cost in Eq (3.9) among the following candidates: i) 0; ii) the real positive roots of the third-degree polynomial:

$$
\begin{aligned}
4 \cdot h^3 + 4(\sum_{\mu=1,\mu\neq j} [\hat{\mathbf{M}}_t^{k-1}(\mu)]^2 - \delta^k(j,j)) \cdot h \\
- 4(\sum_{\mu=1,\mu\neq j}^{p} \delta^k(j,\mu)\hat{\mathbf{M}}_t^{k-1}(\mu)) + \lambda = 0
\end{aligned}
\tag{3.10}
$$

iii) the real negative roots of the third-degree polynomial:

$$
\begin{aligned}
4 \cdot h^3 + 4(\sum_{\mu=1,\mu\neq j} [\hat{\mathbf{M}}_t^{k-1}(\mu)]^2 - \delta^k(j,j)) \cdot h \\
- 4(\sum_{\mu=1,\mu\neq j}^{p} \delta^k(j,\mu)\hat{\mathbf{M}}_t^{k-1}(\mu)) - \lambda = 0
\end{aligned}
\tag{3.11}
$$

To obtain the roots for the above two third-degree polynomial, we utilized companion matrices, [13]. The proposed sparsity-aware kernel matrix decomposition algorithm is tabulated as Algorithm 1. In fact, convergence to at least a stationary point of the cost in Eq (3.7) is established in Appendix B.

Algorithm 1 Sparse Kernel Covariance Factorization

```
1: Using frames within time interval $[(t-1) \cdot F + 1, t \cdot F]$:
2: Form the kernel covariance matrix using Eq (3.5).
3: Initialize $\mathbf{M}_t(j)$'s with 0's.
4: for $k = 1, 2, \ldots, \kappa$ do
5:     Evaluate $\delta^k(j,\mu)$ for $j, \mu = 1, \ldots, p$.
6:     Determine the updates $\{\hat{\mathbf{M}}_t^k(j)\}$ after determining the positive roots of Eq (3.10) and the negative roots of Eq (3.11).
7:     If $||\mathbf{M}_t^k - \mathbf{M}_t^{k-1}|| \leq \epsilon$, where $\epsilon$ is the desired error threshold then break.
8: end for
```

After determining the sparse factor $\hat{\mathbf{M}}_t^k$, the nonzero entries' indices (support) of $\hat{\mathbf{M}}_t^k$ will point to the moving object pixels within the frame sequence during time interval $[(t-1)F + 1, tF]$. Next, we generalize the pixel classification framework in the presence of multiple objects.

3.3. Multiple objects

In the presence of multiple objects in a frame sequence the sparse factorization formulation in Eq (3.7) can be used by introducing multiple columns in $\mathbf{M}_t$ and employing the same coordinate descent process described earlier. One challenge in the presence of multiple objects is the correlation among objects that have similar pixel intensities and/or texture. In this case, the sparse factorization framework may return sparse factors $\hat{\mathbf{M}}_t$ that contain nonzero values in entries corresponding to pixels of more than one objects. Thus, it may be necessary to do some extra clustering among these pixels to separate them according to the object they correspond to. This process will enable to split the objects that may appear in the same sparse factor and enable us to track them individually.

To split the objects that may be present in a sparse factor returned by the sparse factorization algorithm we rely on the property that pixels corresponding to the same object present in a sparse factor

$\hat{\mathbf{M}}_t$ should be neighboring and thus closer (in terms of Euclidean distance) compared to pixels corresponding a different object (placed at a different part of the frame).

Let $\mathcal{P}^t$ denote the nonzero entries of $\hat{\mathbf{M}}_t^k$ which indicates the moving objects' pixels, from which the coordinates $\mathbf{z}_i$ for each pixel $i \in \mathcal{P}^t$ can be further extracted. Then, we employ K-means clustering, see [10], aiming at partitioning the $\mathcal{P}^t$ pixels into Z_t clusters $\{\Xi_1, \ldots, \Xi_{Z_t}\}$ according to the similarity of their corresponding coordinates $\mathbf{z}_i, i \in \mathcal{P}^t$. K-means clusters the pixels by minimizing the following formulation

$$\arg\min_{\Xi_j} \sum_{j=1}^{Z_t} \sum_{\mathbf{z}_i \in \Xi_j} ||\mathbf{z}_i - \xi_j||^2, \tag{3.12}$$

where ξ_j corresponds to the centroid of cluster Ξ_j.
In this way, $\mathcal{P}^t$ pixels will be clustered into Z_t clusters, centered at ξ_j, $j = 1, \ldots, Z_t$ corresponding to the different moving objects contained within a sparse factor obtained via Alg. 1. If two clusters' centroid coordinates are too close then:

$$||\xi_j - \xi_{j'}||^2 \le \epsilon_d, \tag{3.13}$$

where ϵ_d is a predefined distance, we will decrease the initial number of clusters to $Z_t - 1$. By setting an upper limit M_{up} on the number of moving objects, we can adjust the required number of clusters Z_t in the aforementioned way to accurately estimate the unknown number of video objects.

4. Frame object tracking

Once the pixels $\mathcal{P}_m^t$ corresponding to object m have been determined, each object's centroid pixel can be determined as described earlier and Kalman filtering will be utilized to accurately track the location of each detected object within the video sequence. Recall that the state vector $\mathbf{s}_m(t)$ contains the location coordinates, as well as the velocity at which the object's centroid is moving along each of the two dimensions present in each frame. It should be emphasized that there may be some errors when clustering the pixels according to the objects they belong to, in which case let $\hat{\mathcal{P}}_m^t$ denote the estimated pixel locations corresponds to object m, while $\hat{\mathbf{c}}_m^t$ corresponds to the corresponding estimate of the object's centroid pixel. The following measurement model can be utilized to associate $\hat{\mathbf{c}}_m^t$ with the state vector $\mathbf{s}_m(t)$ as follows

$$\hat{\mathbf{c}}_m^t = \mathbf{H}(t)\mathbf{s}_m(t) + \mathbf{w}_m(t) = \mathbf{c}_m^t + \mathbf{w}_m(t), \tag{4.1}$$

with $m = 1, \ldots, M$, where

$$\mathbf{H}(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

while $\mathbf{w}_m(t)$ corresponds to the localization error that may be present in $\hat{\mathbf{c}}_m^t$ when utilizing the sparse factorization approach in Sec. III. It is assumed that the noise $\mathbf{w}_m(t)$ is zero mean with variance $\sigma_w^2 \cdot \mathbf{I}_{2\times 2}$. After numerical testing, the noise standard deviation σ_w is found to be less than 3 pixels.

Although, the distribution of the noise $w_m(t)$ is unknown in (4.1) and not necessarily Gaussian, the Kalman filter will still provide the linear minimum mean-square estimation for the state and observation models in (2.1) and (4.1). The object state estimator and corresponding error covariance matrix, obtained by the Kalman filter for object m are denoted here as $\hat{\mathbf{s}}_m(t|t)$ and $\mathbf{P}_m(t|t)$, respectively. The prediction step in the Kalman filter used here, see e.g. [17], involves the following updating recursions for the state estimator and corresponding covariance

$$\hat{\mathbf{s}}_m(t|t-1) = \mathbf{A}\hat{\mathbf{s}}_m(t-1|t-1) \tag{4.2}$$

$$\hat{\mathbf{P}}_m(t|t-1) = \mathbf{A}\hat{\mathbf{P}}_m(t-1|t-1)\mathbf{F}^T + \mathbf{\Sigma}_u. \tag{4.3}$$

The estimated centroid $\hat{\mathbf{c}}_m^t$ will then be used to carry out the correction step of the Kalman filter which involves the following updating recursions:

$$\begin{aligned} \hat{\mathbf{s}}_m(t|t) &= \hat{\mathbf{s}}_m(t|t-1) + \mathbf{G}_m(t) \cdot [\hat{\mathbf{c}}_m^t - \mathbf{H}(t)\hat{\mathbf{s}}_m(t|t-1)] \\ \mathbf{P}_m(t|t) &= (\mathbf{I} - \mathbf{G}_m(t)\mathbf{H}(t))\mathbf{P}_m(t|t-1) \end{aligned} \tag{4.4}$$

for $m = 1, \ldots, M$), while the matrix $\mathbf{G}_m(t)$ which corresponds to the Kalman gain can be evaluated as

$$\mathbf{G}_m(t) = \mathbf{P}_m(t|t-1)\mathbf{H}^T(t)(\sigma_w^2 \cdot \mathbf{I}_{2\times2} + \mathbf{H}(t)\mathbf{P}_m(t|t-1)\mathbf{H}^T(t))^{-1}. \tag{4.5}$$

A separate Kalman filter is implemented for each of the M objects determined using Alg. 1. Each of these filters is using the estimated centroid $\hat{\mathbf{c}}_m^t$ found at every time instant t.

5. Real-time object identification and tracking

5.1. Dealing with large frames

One may notice that the proposed matrix decomposition scheme may involve high complexity computations in the initialization stage when determining $\mathbf{M}_t$, especially when the video resolution is very high leading to a large number of pixels per frame. To deal with this issue, we resort to a divide and conquer strategy. We split the $f_x f_y \times f_x f_y$ kernel covariance matrix into smaller parts corresponding to smaller regions of a frame with size $\varrho \times \varrho$, where $\varrho \ll f_x f_y$.

For each of these smaller regions we obtain $\hat{\mathbf{M}}_t^j$ for $j = 1, \ldots, J$ using Alg. 1 on the kernel covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{\phi}_{\mathbf{x}j},t}$ that corresponds to subframe $\mathbf{x}_j^t$ that subsequently corresponds to a smaller region of the frame $\mathbf{x}_t$ at time instant t, and $J = \frac{f_x f_y}{\varrho^2}$. Then, the smaller sparse factors $\hat{\mathbf{M}}_t^j$ are stacked as follows

$$\hat{\mathbf{M}}_t = [\{\hat{\mathbf{M}}_t^1\}^T, \ldots, \{\hat{\mathbf{M}}_t^J\}^T]^T, \tag{5.1}$$

to construct the sparse factor $\hat{\mathbf{M}}_t$ corresponding to the kernel covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{\phi}_x,t}$ of the entire frame $\mathbf{x}_t$. It is worth noting that $\hat{\mathbf{M}}_t$ is acquired without the need of factorizing the much larger in size $f_x f_y \times f_x f_y$ kernel covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{\phi}_x,t}$. Proceeding as before, the nonzeros entries of $\hat{\mathbf{M}}_t$ can be utilized to estimate the $\mathcal{P}_t$ object pixels. After implementing the K-means clustering method on the $\mathcal{P}_t$ pixels, cluster centroids will serve as the initialization positions of the objects in the filtering stage. Further, the pixel subsets $\mathcal{P}_m^t$ are also used to estimate the width and height of a rectangular subframe, say width w_s^0 and height h_s^0, that surrounds the pixels of each moving object in the frame and gives a rectangular area that estimates the objects' location.

5.2. *Object identification and tracking*

Next, it is outlined how the Kalman filter and the sparse kernel factorization algorithm interact with each other to track multiple moving objects in a given video sequence. During the start-up stage, a number of F_s frames will be utilized to evaluate the kernel covariance matrix. After applying Alg. 1 using the divide and conquer implementation in Sec. (5.1), the nonzero entries in the acquired sparse vector $\hat{\mathbf{M}}_0$ will point to the pixels of the moving objects in the video.

From $\hat{\mathbf{M}}_0$, we can extract the pixels which form the detected moving objects. Here the size of the estimated rectangular area surrounding the object (cf. Sec. 5.1), namely w_s^0 and h_s^0 will be increased to w_s and h_s which satisfies $w_s \geq w_s^0$, and $\text{mod}(w_s, \varrho) = 0$, and $h_s \geq h_s^0$, and $\text{mod}(h_s, \varrho) = 0$. This results a slightly larger rectangular region for each object in which smaller regions of size $\varrho \times \varrho$ can be further extracted. In the next time instance, we will just incorporate the pixels around the predicted centroid position $\hat{\mathbf{s}}_m(t|t-1)$ from Kalman filter Eq (4.2) to form the object kernel covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{\phi}_{\mathbf{x}},t,m}$ following Eq (3.1) in which $\mathbf{x}$ contains the pixels $x^{i,j}$ with x- and y- coordinates within the intervals

$$\begin{aligned}
[\hat{s}_m(t|t-1)]_1 - w_s/2 \leq i \leq [\hat{s}_m(t|t-1)]_1 + w_s/2 \\
[\hat{s}_m(t|t-1)]_2 - h_s/2 \leq j \leq [\hat{s}_m(t|t-1)]_2 + h_s/2.
\end{aligned} \tag{5.2}$$

Similarly to the initialization stage, the divide and conquer implementation is carried out in the kernel covariance matrix $\hat{\mathbf{\Sigma}}_{\boldsymbol{\phi}_{\mathbf{x}},t,m}$ for each object separately to acquire the pixels which corresponds to the moving object m at the current time instant. This approach reduces the computational complexity since each object allocated areas is further split into smaller regions.

To account for newborn or disappearing objects in the video resulting a time-varying number of objects we can periodically generate and factorize the kernel covariance matrix in the entire frame. The complete scheme is outtlined as Alg. 2.

Algorithm 2 Joint Multi-Object Detection and Tracking

```
1: Start-up stage ($t = 0$)/Reconfiguration ($\text{mod}(t, Tc) = 0$): For $F_s$ consecutive frames, the kernel covariance matrix is formed according to Eq (3.5) and Algorithm 1 is applied in the entire frame to determine moving objects in the input frame sequence.
2: for $t = 1, 2, \ldots,$ do
3:     Gather frames within time interval $[(t-1) \cdot F + 1, t \cdot F]$.
4:     Using the $F$ acquired frames form the kernel covariance matrix $\mathbf{\Sigma}_{\boldsymbol{\phi}_x,t,m}$ with pixels in a rectangular region of size $w_s \times h_s$ with centroid $[\hat{s}_m(t|t-1)]_{1:2}$ for $m = 1, \ldots, M$.
5:     Apply Algorithm 1 to $\mathbf{\Sigma}_{\boldsymbol{\phi}_x,t,m}$ and acquire the nonzero entries in $\hat{\mathbf{M}}_t^m$.
6:     Apply Kalman filtering for each of the $M$ objects, i.e., Eq (4.2)-(4.5) to track each of the $M$ objects' centroid pixel.
7: end for
```

6. Synthetic numerical tests

The performance of the proposed scheme is first tested on a synthetic frame sequence that contains three rectangular objects that move independently from each other. Video background contains randomly generated Gaussian zero-mean noise with variance 20, while the objects consist of pixels with intensity varying between 240 to 255. The synthetic video contains 60 frames, during which object 1 moves from the left to the right, object 2 moves from the top to the bottom of the frame and object 3 moves from the right to the left. The frame size is 100 by 100, while all the objects are of size 10 by 10. The true coordinates of each object's centroid pixel are recorded to evaluate our proposed tracking scheme. To select a proper kernel variance, firstly we empirically choose a variance range [0.001, 1], by checking the kernel covariance matrix formed with different variances in the variance range, we select the variance value which results the desired block diagonal structure in the kernel covariance matrix. The value is set as $\sigma^2 = 0.01$. Three sample images are given in Figure 2, 3, 4, with frame indices 20, 40, and 60. The objects are marked out by a dark-colored circle to show how our method captures the momentary location of each of the rectangular moving objects. The tracking root mean-squared error (RMSE) which quantifies the number of pixels by which the estimated centroid pixel coordinates is 'missing' the true centroid of each object is depicted in Figure 5. It is clear that both the proposed tracking scheme and the scheme in [34] localizes the moving objects within 2 pixels of accuracy on average. Though, the approach in [34] requires prior knowledge of the objects' initial coordinates, and a proper search window size which our scheme can generate in an unsupervised manner and without the need of any prior information.

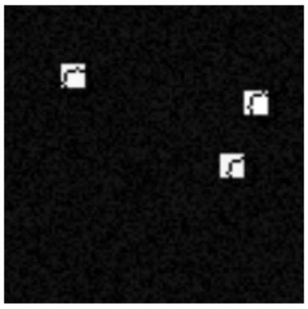

Figure 2. Synthetic frame sequence; Frame 20. The white squares correspond to the ground truth objects moving, while the dark curve inside the squares gives the estimated object outline by applying tracking using a Gaussian kernel with variance in the range [0.001, 1].

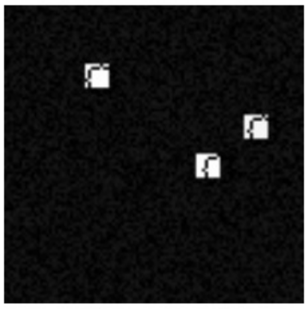

Figure 3. Synthetic frame sequence; Frame 40.The white squares correspond to the ground truth objects moving, while the dark curve inside the squares gives the estimated object outline by applying tracking using a Gaussian kernel with variance in the range [0.001, 1].

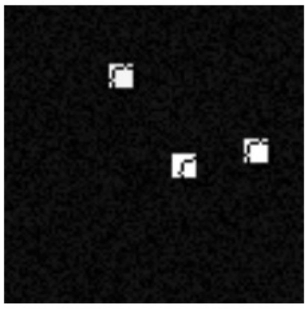

Figure 4. Synthetic frame sequence; Frame 60.The white squares correspond to the ground truth objects moving, while the dark curve inside the squares gives the estimated object outline by applying tracking using a Gaussian kernel with variance in the range [0.001, 1].

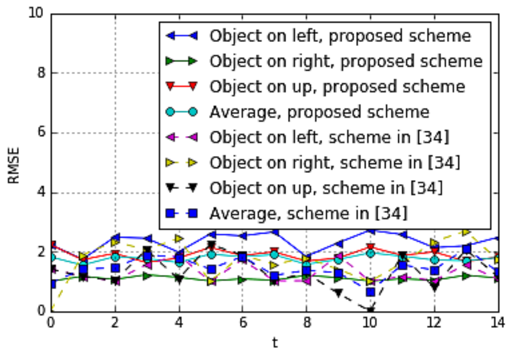

Figure 5. Tracking root mean-squared error (RMSE) for the synthetic video sequence. The proposed novel method is compared with the approach in [34].

7. Multi-object detection and tracking in thermal video

Next, the proposed tracking scheme is tested on video sequences extracted from the datasets available on the OTCBVS website [21], where a Raytheon L-3 Thermal-Eye 2000AS infrared sensor is utilized to acquire 8-bit grayscale images (with a resolution of 320 × 240 pixels per frame). The first image sequence is extracted from the OTCBVS dataset 05, i.e, terravic motion infrared database. In this video sequence, a man with a weapon moves from the left to the right with deformation. Six sample frames are presented in Figure 6, 7, 8. Even though the shape of the target varies with the non-rigid movement of the limbs, our novel algorithm is able to detect the person and tracks the target accurately (the white box surrounding the objects denotes the estimated area where the algorithm projects there is an object). Frames 243 and 282 are zoomed in to show the accuracy of the bounding box our proposed tracking scheme generated; see Figure 9, 10.

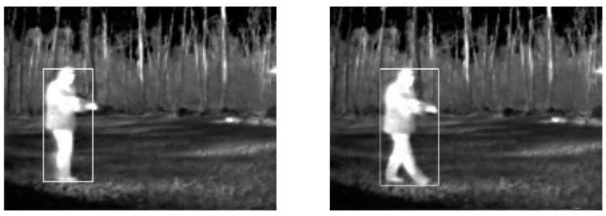

Figure 6. Tracking results for sequence 1; Frames 231, and 240. The white frame around the moving person indicates the estimated object area by the novel approach.

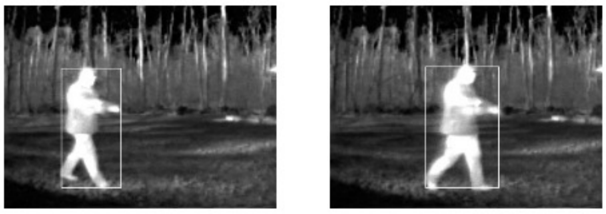

Figure 7. Tracking results for sequence 1; Frames 243, and 267.The white frame around the moving person indicates the estimated object area by the novel approach.

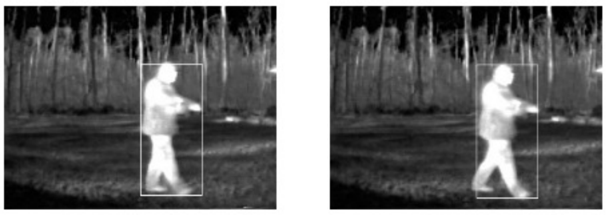

Figure 8. Tracking results for sequence 1; Frames 282, and 288.

Another experiment is conducted on another video sequence extracted from the OTCBVS database. In this sequence, there are two pedestrians one of which moves from the left to the right, while the other pedestrian moves from the right to the left. Six sample frames are displayed in Figure 11, 12, 13. For a better view of how our proposed tracking scheme manages to localize both pedestrians, frames 251 and 360 are zoomed in and displayed in Figure 14, 15, 16, and 17. The tracking RMSE for our proposed method and the tracking scheme in [34] is compared in Figure 18. It can be seen that our tracking scheme outperforms the scheme in [34] for both pedestrians, and it is worth noting that for our scheme, the average tracking error for most of the tracking time is below 4 pixels.

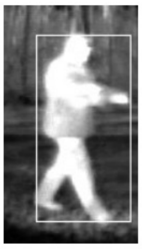

Figure 9. Tracking results for sequence 1; Frames 243 zoomed in, demonstrating the accurate tracking of the area in which the moving object resides.

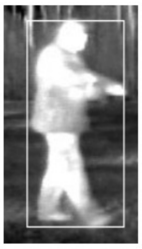

Figure 10. Tracking results for sequence 1; Frames 282 zoomed in, demonstrating the accurate tracking of the area in which the moving object resides.

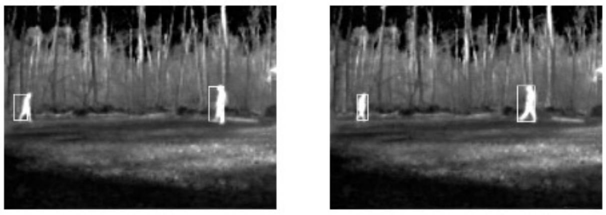

Figure 11. Tracking results for sequence 2; Frames 240 and 264.

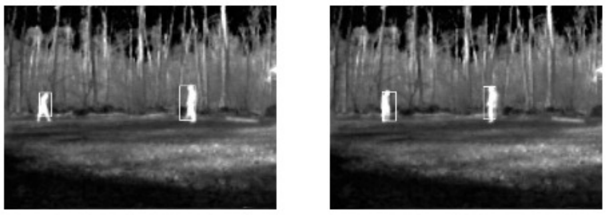

Figure 12. Tracking results for sequence 2; Frames 282 and 318.

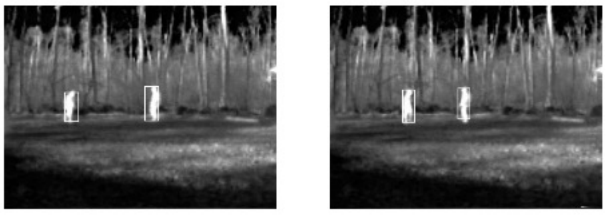

Figure 13. Tracking results for sequence 2; Frames 336 and 360.

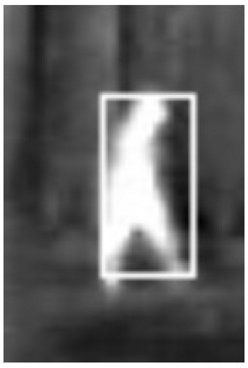

Figure 14. Tracking results for sequence 2; Frame 251 zoomed in for pedestrian on the left.

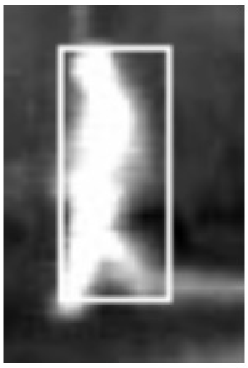

Figure 15. Tracking results for sequence 2; Frame 251 zoomed in for pedestrian on the right.

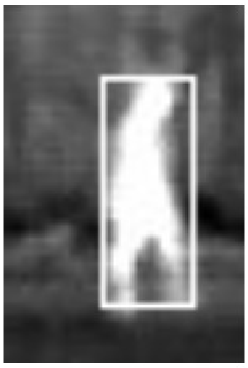

Figure 16. Tracking results for sequence 2; Frame 360 zoomed in for pedestrian on the left.

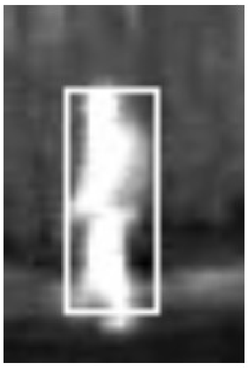

Figure 17. Tracking results for sequence 2; Frame 360 zoomed in for pedestrian on the right.

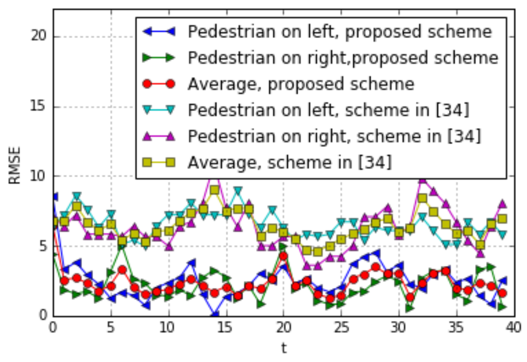

Figure 18. OTCBVS thermal video tracking RMSE. The novel approach is compared with the method in [34].

Further, we compare our novel object detection scheme with a simple approach where objects are detected by thresholding the pixel values. The result is displayed in Table. 1. One can observe that the thresholding approach is very sensitive to the selection of the threshold and it may detect the wrong number of objects. Thresholding may miss objects whose pixels have intensity less than the chosen threshold or declare as objects background artifacts whose pixel intensity is larger than the threshold. However, our scheme is not affected by background noise and is capable of finding the correct number of objects.

Table 1. Number of detected objects.

	Novel object detection	Pixel thresholding
Synthetic video sequence	3	3
Thermal video sequence 1	1	2
Thermal video sequence 2	2	1

8. Tracking with missing pixels

Oftentimes, videos may be corrupted due to camera or storage issues, resulting missing pixels in the video frames. Here, our tracking scheme is tested in the scenario where a portion of the frame pixels are missing (their intensity is set to 0); see Figure 19 and 20. Here, two sample frames 224, 252 with a 5% random pixel loss are provided to display the tracking result. Despite the missing pixels our approach is able to track the objects. For the pedestrian on the right, despite the loss of several pixels, our tracking scheme enables precise localization. Notice that the tracking performance of the pedestrian on the left is not as good as the pedestrian on the right, since the left side pedestrian occupies a smaller number of pixels which results a larger portion of the object to disappear in the presence of missing pixels.

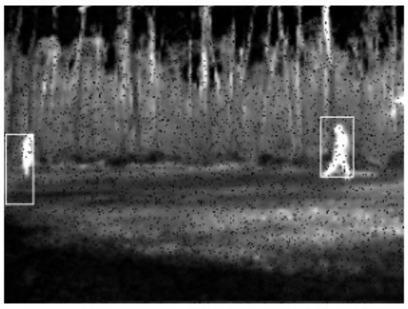

Figure 19. Tracking in the presence of 5% missing pixels; Frame 224.

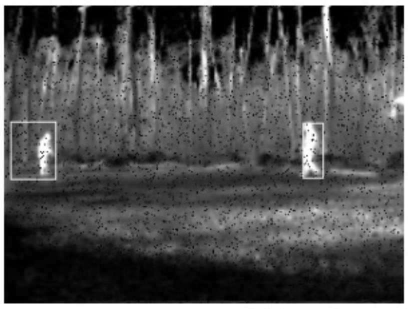

Figure 20. Tracking in the presence of missing pixels; Frame 252.

9. Concluding remarks

A novel multi-object detection and tracking algorithm was put forth in video sequences. The task of identifying objects in a sequence of frames was transformed in a sparse kernel covariance factorization problem, where the support of the estimated sparse factors point to the pixels of each object present in a frame. To this end, a sparsity-aware kernel covariance matrix factorization scheme, based on norm-1 regularization was proposed and minimized utilizing a coordinate descent approach. After objects are successfully determined, Kalman filtering is implemented cooperatively with the sparse kernel covariance factorization scheme to allow accurate tracking of each object's centroid pixels. Numerical tests on different video datasets validate the effectiveness of our proposed video tracking mechanism in the presence of multiple objects, and corroborate the improved tracking performance over existing alternatives.

Acknowledgments

Work in this paper was supported by the AFOSR Grant FA9550-15-1-0103 and the NSF grant 1509780.

References

1. Bar-Shalom Y, (2001) *Estimation With Applications to Tracking and Navigation*. New York: Wiley.

2. Bertsekas DP, (2003) *Nonlinear Programming*, Second Edition, Athena Scientific.
3. Black MJ, Jepson AD (1998) Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *Int J Comput Vision* 26: 63–84.
4. Chen IK, Hsu SL, Chi CY, et al. (2014) Automatic video segmentation and object tracking with real-time RGB-D data *Proc of the IEEE Intl Conf on Consum Electr*: 486–487.
5. Cielniak G, Duckett T, Lilienthal AJ (2007) Improved data association and occlusion handling for vision-based people tracking by mobile robots. *Proc of IEEE Intl Conf on Intell Robot Syst*.
6. Davis JW, Sharma V (2007) Background-subtraction using contour-based fusion of thermal and visible imagery. *Comput Vis Image Und* 106: 162–182.
7. Ennulat RD, Pommerrenig D (1988) Uncooled high resolution infrared imaging plane.
8. Hanif M, Ali U (2006) Optimized visual and thermal image fusion for efficient face recognition. *Proc of IEEE Intl Conf on Inform Fusion*.
9. Harchaoui Z, Bach F (2007) Image classification with segmentation graph kernels. *Proc of IEEE Conf on Comput Vis and Pattern Recogn*.
10. Hartigan JA, Wong MA (1979) Algorithm AS 136: A K-means clustering algorithm. *J R Stat Soc C-Appl* 28: 100–108.
11. Heo J, Kong SG, Abidi BR, et al. (2004) Fusion of visual and thermal signatures with eyeglass removal for robust face recognition. *Proc of IEEE Conf on Comput Vis and Pattern Recogn Workshop*, 122–122.
12. Hofmann T, Schölkopf B, Smola AJ (2008) Kernel methods in machine learning. *Ann Stat*, 1171–1220.
13. Horn RA, (1985) Matrix Analysis. Cambridge, U.K.: Cambridge Univ. Press.
14. Isola P, Zoran D, Krishnan D, et al. (2014) Crisp boundary detection using pointwise mutual information. *European Conference on Computer Vision*, 799–814.
15. Jiang M, Pan Z, Tang Z (2017) Visual object tracking based on Cross-modality Gaussian-Bernoulli deep Boltzmann machines with RGB-D sensors. *Sensors* 121.
16. Kang K, Maroulas V, Schizas I, et al. (2018) Improved distributed particle filters for tracking in wireless sensor network. *Comput Stat Data An* 117: 90–108.
17. Kay SM, (1993) *Fundamental of Statistical Signal Processing: Estimation Theory*, Prentice Hall.
18. Kwak N (2012) Kernel discriminant analysis for regression problems. *Pattern Recogn* 45: 2019–2031.
19. Luber M, Spinello L, Arras KO (2011) People tracking in RGB-D data with on-line boosted target models. *Proc of IEEE Intl Conf on Intell Robot Syst*.
20. Maroulas V, Stinis P (2012) Improved particle filters for multi-target tracking. *J Comput Phys* 231: 602–611.
21. IEEE OTCBVS WS Series Bench; Roland Miezianko, Terravic Research Infrared Database. Available: http://vcipl-okstate.org/pbvs/bench/

22. Padole CN, Alexandre LA (2010) Motion based particle filter for human tracking with thermal imaging. *Proc of IEEE Intl Conf on Emerging Trends in Engineering and Technology (ICETET)*: 158–162.

23. Peng J, Zhou Y, Chen CLP (2015) Region-kernel-based support vector machines for hyperspectral image classification *IEEE T Geosci Remote* 53: 4810–4824.

24. Ren G, Maroulas V, Schizas ID (2015) Distributed Sensors-Targets Spatiotemporal Association and Tracking. *IEEE Taes* 51: 2570–2589.

25. Ren G, Maroulas V, Schizas ID (2016) Decentralized Sparsity-Based Multi-Source Association and State Tracking. *Signal Process* 120: 627–643.

26. Ren G, Maroulas V, Schizas ID (2016) Exploiting sensor mobility and covariance sparsity for distributed tracking of multiple targets. *J Adv Sig Pr* 2016: 53.

27. Rosipal R, Girolami M, Trejo LJ, et al. (2001) Kernel PCA for feature extraction and de-noising in nonlinear regression. *Neural Comput Appl* 10: 231–243.

28. Schizas ID (2013) Distributed informative-sensor identification via sparsity-aware matrix factorization. *IEEE Trans on Sig Proc* 61: 4610–4624.

29. Teichman A, Lussier JT, Thrun S (2013) Learning to segment and track in RGBD. *IEEE T Autom Sci Eng* 10: 841–852.

30. Tibshirani R (1996) Regression shrinkage and selection via the Lasso. *J R Stat Soc B* 58: 267–288.

31. Treptow A, Cielniak G, Duckett T (2005) Active people recognition using thermal and grey images on a mobile security robot. *Proc of IEEE Intl Conf on Intell Robot Syst.*

32. Tseng P (2001) Convergence of a block coordinate descent method for nondifferentiable minimization. *J Opt Theory App* 109: 475–494.

33. Vert JP, Tsuda K, Schölkopf B (2004) A primer on kernel methods. *Kernel Methods in Comput Biol*: 35–70.

34. Zhang K, Zhang L, Yang and M (2012) Real-time compressive tracking. *European Conference on Computer Vision*: 864–877.

35. Xu F, Liu X, Fujimura K (2005) Pedestrian detection and tracking with night vision. *IEEE T Intell Transp*: 63–71.

36. Yasuno M, Yasuda N, Aoki M (2005) Pedestrian detection and tracking in far infrared images. *IEEE Conf on Intell Transport S*: 131–136.

37. Zou H, Hastie T, Tibshirani R (2006) Sparse principal component analysis. *J Comput Graph Stat*: 15.

Supplementary

10. Proof of Eq (3.10), (3.11)

Let $\mathbf{M}_t(j) = h$, while setting the rest of the minimization variables in (3.7) to their most up-to-date values at the end of cycle $k - 1$. It follows that $\hat{\mathbf{M}}_t^k(j)$ is the minimizer of

$$\arg\min_h h^4 + c_1 \cdot h^2 + c_2 \cdot h + \lambda t, \quad \text{s. to } |h| \leq t, \tag{10.1}$$

where

$$c_1 = 2\sum_{i,i\neq j}^{p}[\hat{\mathbf{M}}_t^{k-1}(i)]^2 - 2\delta^k(j,j) + \phi, \text{ and} \tag{10.2}$$

$$c_2 = -4\sum_{i,i\neq j}^{p}\delta^k(j,i)\hat{\mathbf{M}}_t^{k-1}(i). \tag{10.3}$$

After evaluating the derivatives of the cost in (10.1) wrt h and t and applying the Karush-Kuhn-Tucker optimality conditions [2] it follows that $h^* := \hat{\mathbf{M}}_t^k(j)$ should satisfy $4(h^*)^3 + 2c_1h^* + c_2 + \mu_1^* - \mu_2^* = 0$ and $-\mu_1^* - \mu_2^* + \lambda = 0$, where μ_1^* and μ_2^* are the optimal multipliers corresponding to the inequality constraints of (10.1). Note that $\mu_1^* \geq 0, \mu_2^* \geq 0$, while the complementary slackness conditions impose that $\mu_1^*(h^* - t^*) = \mu_2^*(-t^* - h^*) = 0$. If $h^* > 0$ the slackness conditions imply that $\mu_2^* = 0$ from which it follows that $\mu_1^* = \lambda$. Substituting the latter values in $4(h^*)^3 + 2c_1h^* + c_2 + \mu_1^* - \mu_2^* = 0$ gives (3.10). Similarly, the negative candidate minimizers of (10.1) can be obtained by the roots of (3.11).

11. Convergence of Alg. 1:

Let $\ell(\{\mathbf{M}_t(j)\}_{j=1}^p)$ denote the cost in (3.7) which is defined over $\mathbb{R}^{p\times 1}$, and let us define

$$\ell_0(\{\mathbf{M}_t(j)\}_{j=1}^p,) := \sum_{j=1}^p\sum_{j'=1}^p[\hat{\boldsymbol{\Sigma}}_{\phi_x,t}(j,j') \\ - \mathbf{M}_t(j)\mathbf{M}_t(j')]^2.$$

Further, consider the following level set:

$$\mathcal{L}_t^0 := \{\{\mathbf{M}_t(j)\}_j^p : \ell(\{\mathbf{M}_t(j)\}_{j=1}^p) \leq \ell(\hat{\mathbf{M}}_t^0)\}, \tag{11.1}$$

where $\hat{\mathbf{M}}_t^0$ is the $p \times 1$ matrix used to initialize Alg. 1 and selected such that $\|\hat{\mathbf{M}}_t^0\|_1 < \infty$, from which it follows that $\ell(\hat{\mathbf{M}}_t^0) < \infty$. Then, from (11.1) and the form of $\ell(\cdot)$ it follows that the member matrices $\mathbf{M}_t$ of $\mathcal{H}_t^0$ satisfy

$$\sum_{j=1}^p \lambda_\ell|\mathbf{M}_t(j)| \leq \ell(\hat{\mathbf{M}}_t^0) < \infty.$$

Thus, the level set $\mathcal{L}^0$ is closed and bounded (compact). Also, $\ell(\cdot)$ is continuous on $\mathcal{L}^0$.
Recall from [cf. (10.1)] that the cost involved in updating $\hat{\mathbf{M}}_t^k(j)$ can be written as $J_t^k(j) := h^4 + c_1h^2 + c_2h + \lambda|h|$. If $c_2 \neq 0$ then after determining the monotonicity of $J_t^k(j)$, it has a unique minimizer. And if $c_2 = 0$, then $J^k(j)$ is symmetric around zero. In that case if $c_1 > 0$ then the unique minimizer of $J_t^k(j)$ is 0. Though, if $c_1 < 0$ then $J_t^k(j)$ has two minimizers with the same magnitude but different sign. In that case we can consistently select the positive (or negative) minimizer ensuring a unique minimizer per iteration. Function $\ell(\cdot)$ satisfies the regularization conditions outlined in [32, (A1)]. In detail, the domain of $\ell_0(\cdot)$ is formed by matrices whose entries satisfy $\mathbf{M}_t(j) \in (-\infty, +\infty)$. Then, domain$(\ell_0) = (-\infty,\infty)^{p\times 1}$ is an open set. Further, $\ell_0(\cdot)$ is Gâteaux differentiable over domain(ℓ_0). The Gâteaux derivative is

$$\ell_0'(\mathbf{M};\boldsymbol{\Delta}_M) := \lim_{\epsilon\to 0}[\ell_0(\mathbf{M} + \epsilon\boldsymbol{\Delta}_M) - \ell_0(\mathbf{M})]/\epsilon. \tag{11.2}$$

After carrying out the necessary algebraic operations it follows readily that $\ell_0'(\mathbf{M};\boldsymbol{\Delta}_M)$ exists for all $\boldsymbol{\Delta}_M \in$ domain(ℓ_0), and it equals

$$-2\text{tr}[(\hat{\boldsymbol{\Sigma}}_{x,t} - \mathbf{M}_t\mathbf{M}_t^T)(\mathbf{M}_t\boldsymbol{\Delta}_M^T + \boldsymbol{\Delta}_M\mathbf{M}_t^T)] + \mathbf{1}^T(\mathbf{M}_t \odot \boldsymbol{\Delta}_M)\mathbf{1}.$$

The aforementioned properties ensure that Alg. 1 iterates converge to a stationary point of $\ell(\cdot)$ [32, Thm. 4.1 (c)].