Chapter 5
Occlusion Detection via Structured Sparse Learning for Robust Object Tracking

Tianzhu Zhang, Bernard Ghanem, Changsheng Xu and Narendra Ahuja

Abstract Sparse representation based methods have recently drawn much attention in visual tracking due to good performance against illumination variation and occlusion. They assume the errors caused by image variations can be modeled as pixel-wise sparse. However, in many practical scenarios, these errors are not truly pixel-wise sparse but rather sparsely distributed in a structured way. In fact, pixels in error constitute contiguous regions within the object's track. This is the case when significant occlusion occurs. To accommodate for nonsparse occlusion in a given frame, we assume that occlusion detected in previous frames can be propagated to the current one. This propagated information determines which pixels will contribute to the sparse representation of the current track. In other words, pixels that were detected as part of an occlusion in the previous frame will be removed from the target representation process. As such, this paper proposes a novel tracking algorithm that models and detects occlusion through structured sparse learning. We test our tracker on challenging benchmark sequences, such as sports videos, which involve heavy occlusion, drastic illumination changes, and large pose variations. Extensive experimental results show that our proposed tracker consistently outperforms the state-of-the-art trackers.

T. Zhang (✉)
Advanced Digital Sciences Center of Illinois, Singapore, Singapore
e-mail: tzzhang10@gmail.com

B. Ghanem
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
e-mail: bernard.ghanem@kaust.edu.sa

C. Xu
Institute of Automation, Chinese Academy of Sciences, CSIDM, People's Republic of China
e-mail: csxu@nlpr.ia.ac.cn

N. Ahuja
University of Illinois at Urbana-Champaign, Urbana, IL, USA
e-mail: n-ahuja@illinois.edu

T.B. Moeslund et al. (eds.), *Computer Vision in Sports*,
Advances in Computer Vision and Pattern Recognition,
DOI 10.1007/978-3-319-09396-3_5

5.1 Introduction

For sports video analysis, knowing the location of each player on the field at each point of the game is crucial for sports experts (e.g., coaches, trainers, and sports analysts) to better understand complex player formations and trajectory patterns, which ultimately depict the effectiveness of their teams 'strategies as well as their opponents.' Being able to effectively track players can enable the development of reliable activity recognition and higher level processing modules for sports video analysis. Such a tracking building block will have a positive impact on how sports experts analyze game footage, how content providers identify/display particular sports events and highlights accompanied with relevant advertisements, and how end users browse and query large collections of sports video. Moreover, visual tracking is a classical problem in computer vision; it is a core task for many applications [44, 48, 50] e.g., automatic surveillance, robotics, human computer interaction, action recognition, etc. It is also very challenging due to appearance variations such as occlusion, illumination change, significant motion, background clutter, etc. Over the years, a significant amount of effort has been made to overcome these challenges. To survey many of these algorithms, we refer the reader to [32, 41].

A truly robust tracking method must be able to handle occlusion. However, modeling occlusion is not straightforward. There exists a significant amount of work that addresses this issue through statistical analysis [16, 34], robust statistics [1, 7], patch matching [39], the use of multiple cameras [12, 31], context information [40], model analysis [13], and learning occlusion with likelihoods [20]. Recently, sparse representation has been successfully applied to visual tracking [29, 46, 47, 49] under the particle filter framework as an attempt to alleviate the occlusion problem in tracking. In these methods, particles are randomly sampled around the states of the tracked object according to a zero-mean Gaussian distribution. At time t, n particles are sampled. The observation (pixel color values) of each particle in the frame is denoted as: $\mathbf{x} \in \mathbb{R}^d$. In the noiseless case, each particle $\mathbf{x}$ is represented as a linear combination $\mathbf{z}$ of templates that form a dictionary $\mathbf{D} = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_m]$, such that $\mathbf{x} = \mathbf{Dz}$. $\mathbf{D}$ can be constructed from an overcomplete sampling of the target object, based on an initial bounding box at the start of tracking, and dynamically updated to maintain an up-to-date target appearance model.

In many visual tracking scenarios, targets are often partially occluded or corrupted by noise. Occlusion is unpredictable as it may affect any part, or occlude any amount, of the target. The occluded object can be either a connected region or a number of randomly scattered pixels, though the former is more likely in natural images. In addition, only a sparse number of these templates is required to reliably represent each particle, which encourages $\mathbf{z}$ to be sparse. To incorporate these two pieces of information, each particle $\mathbf{x}$ should be represented as a sparse linear combination, while allowing for sparse error $\mathbf{e}$ to encode occlusion: $\mathbf{x} = \mathbf{Dz} + \mathbf{e}$. The sparse coefficients $\mathbf{z}$ and sparse error $\mathbf{e}$ are recovered by solving the following ℓ_1 minimization problem. The current tracking result is usually chosen to be the particle $\mathbf{x}$ with minimum reconstruction error w.r.t. dictionary $\mathbf{D}$.

$$\min \|\mathbf{z}\|_1 + \|\mathbf{e}\|_1 \qquad \text{s.t.} \quad \mathbf{x} = \mathbf{D}\mathbf{z} + \mathbf{e} \tag{5.1}$$

This approach has demonstrated to be robust against partial occlusions, which improves tracking performance. However, it suffers from the following drawbacks: (1) The error (due to occlusion) is not sparse for many tracking scenarios, as exemplified in Fig. 5.1. Because a portion of the target is significantly occluded, we need to discard that portion for the sparsity assumption to still hold. (2) This kind of algorithm does not exploit any prior information about the occlusion, especially the important property that occlusion is spatially contiguous. By modeling error pixels as structured and sparse, the representation is made more accurate to model occlusion and better defined.

To deal with the above drawbacks, we propose a new particle filter tracker that involves tracking by occlusion detection, thus appropriately named the TOD tracker. In each frame, particles are represented in a structured sparse learning framework, which exploits prior information about the location of occlusion and its spatial contiguity. This prior is propagated from previous frames in the form of an occlusion mask. The main goal is to show how this prior information can be effectively incorporated into the sparse representation framework, thus improving its robustness against more types of realistic occlusions. Compared with existing methods, the contributions of this work are twofold: (1) We propose a structured sparse learning method for occlusion detection in object tracking. It exploits structure information to make occlusion both sparse and spatially continuous for more robust performance. To the best of our knowledge, this is the first work to use occlusion prior information through structured sparsity in object tracking. (2) Compared to the popular L_1 tracker [29] that does not model occlusion explicitly, our method is generic. In fact, it yields the L_1 tracker as a special case.

The chapter is organized as follows. In Sect. 5.2, we summarize work most related to ours. The particle filter algorithm is reviewed in Sect. 5.3. The proposed tracking approach and optimization methodology are presented in Sects. 5.4 and 5.5, respectively. In Sect. 5.6, we report and analyze extensive experimental results.

Fig. 5.1 Frames from two different video sequences portraying significant occlusion. The ground truth track of each object is designated in *green*. Clearly, occlusion renders the tracking problem very difficult. However, certain assumptions about the structuredness of occlusion (e.g., spatial contiguity) can be exploited to alleviate its effect on tracking performance

5.2 Related Work

Visual tracking is an important topic in computer vision and it has been studied for several decades. There is extensive literature, and we refer to [36, 37, 41] for a more extensive review. In this section, we review and focus on previous work, from which our proposed tracking method borrows some ideas. In fact, we provide a brief overview of visual tracking and sparse representation for object tracking.

5.2.1 Visual Tracking

In general, visual tracking methods can be categorized into two groups: generative and discriminative.

5.2.1.1 Generative Visual Tracking

Generative tracking methods adopt an appearance model to describe the target observations, and the aim is to search for the target location that has the most similar appearance to this model. Examples of generative methods include eigen-tracker [5], mean shift tracker [9], appearance model based tracker [17], context-aware tracker [38], incremental tracker (IVT) [35], fragment-based tracker (Frag) [1], and VTD tracker [21]. In [5], a view-based representation is used for tracking rigid and articulated objects. The approach builds on and extends work on eigenspace representations, robust estimation techniques, and parametrized optical flow estimation. The mean shift tracker [9] is a traditional and popular method, which successfully copes with camera motion, partial occlusions, clutter, and target scale variations. In [17], a robust and adaptive appearance model is learned for motion-based tracking of natural objects. The model adapts to slowly changing appearance, and it maintains a natural measure of the stability of the observed image structure during tracking. The context-aware tracker [38] considers the context of the tracked object for robust visual tracking. Specifically, this method integrates into the tracking process a set of auxiliary objects that are automatically discovered in the video on the fly by data mining. The IVT tracker [35] seeks an adaptive appearance model that accounts for appearance variation of rigid or limited deformable motion. Although it has been shown to perform well when the target object undergoes lighting and pose variation, this method is less effective in handling heavy occlusion or nonrigid distortion as a result of the adopted holistic appearance model. The Frag tracker [1] aims to solve the partial occlusion problem by using a representation that is based on histograms of local patches. The tracking task is carried out by combining votes of matching local patches using an object template. However, this template is not updated and therefore it is not expected to handle appearance changes due to large variations in scale and/or shape deformation. The VTD tracker [21] effectively extends the

conventional particle filter framework with multiple motion and observation models to account for appearance variation caused by change of pose, lighting, and scale as well as partial occlusion. However, as a result of its adopted generative representation scheme that is not equipped to distinguish between target and background patches, it is prone to drift.

5.2.1.2 Discriminative Visual Tracking

Discriminative tracking methods formulate object tracking as a binary classification problem, which aims to find the target location that can best distinguish the target from the background. Examples of discriminative methods are online boosting (OAB) [14], semi-online boosting [15], ensemble tracking [2], co-training tracking [25], online multiview forests for tracking [22], adaptive metric differential tracking [18], and online multiple instance learning tracking [3]. In the OAB tracker [14], online AdaBoost is adopted to select discriminative features for object tracking. Its performance is affected by background clutter and can easily drift. The ensemble tracker [2] formulates the task as a pixel-based binary classification problem. Although this method is able to differentiate between target and background, the pixel-based representation is rather limited and thereby limits its ability to handle occlusion and clutter. Moreover, the MIL tracker [3] extends multiple instance learning to an online setting for object tracking. Although it is able to address the problem of tracker drift, this method does not handle large nonrigid shape deformations well. In [8], a target confidence map is built by finding the most discriminative RGB color combination in each frame. Furthermore, a hybrid approach that combines a generative model and a discriminative classifier is proposed in [43] to capture appearance changes and allow the reacquisition of an object after total occlusion. Also, global mode seeking can be used to detect and reinitialize the tracked object after total occlusion [42]. Another approach uses image fusion to determine the best appearance model for discrimination and then a generative approach for dynamic target updates [6].

5.2.2 Sparse Representation for Object Tracking

Recently, sparse linear representation based on the particle filter framework has been introduced to object tracking and has been shown to achieve significant tracking performance [4, 23, 26, 29, 30, 45, 47, 51]. In the L_1 tracker [29], a tracking candidate is represented as a sparse linear combination of object templates and trivial templates. Sparse representation is computed by solving a constrained ℓ_1 minimization problem with nonnegativity constraints to solve the inverse intensity pattern problem during tracking. The results show good performance at a high computational expense due to the ℓ_1 minimization. In fact, the computational cost grows proportionally with the number of particle samples. In [30], an efficient L_1 tracker with minimum error

bound and occlusion detection is proposed. The minimum error bound is quickly calculated from a linear least squares equation, and serves as a guide for particle resampling in a particle filter framework. Without loss of precision during resampling, the most insignificant samples are removed before solving the computationally expensive ℓ_1 minimization problem. In [26], dynamic group sparsity is integrated into the tracking problem and high-dimensional image features are used to improve tracking robustness. In [23], dimensionality reduction and a customized orthogonal matching pursuit algorithm are adopted to accelerate the L_1 tracker [29]. However, this method may reduce the tracking performance sometimes [23]. In [4], a very fast numerical solver based on the accelerated proximal gradient approach is developed to solve the ℓ_1 norm minimization problem with guaranteed quadratic convergence. In [45], compressive sensing theory is adopted for real-time tracking. In [51], a sparsity-based discriminative classifier and a sparsity-based generative model are designed for tracking. Zhang et al. [47, 49] propose a multitask learning approach to jointly learn the particle representations for robust object tracking. Our proposed method is inspired by the success of these ℓ_1 minimization based trackers, and we will also adopt the sparsity property for robust tracking.

5.3 Particle Filter

A particle filter [11] is a Bayesian sequential importance sampling technique for estimating the posterior distribution of state variables characterizing a dynamic system. It provides a convenient framework for estimating and propagating the posterior probability density function of state variables regardless of the underlying distribution through a sequence of prediction and update steps. Let $\mathbf{s}_t$ and $\mathbf{y}_t^*$ denote the state variable describing the parameters of an object at time t (e.g., motion features) and its observation, respectively. In the particle filter framework, the posterior $p(\mathbf{s}_t|\mathbf{y}_{1:t}^*)$ is approximated by a finite set of n samples $\left\{\mathbf{s}_t^i\right\}_{i=1}^n$ (called particles) with importance weights w_i. The particle samples $\mathbf{s}_t^i$ are independently drawn from an importance distribution $q(\mathbf{s}_t|\mathbf{s}_{1:t-1}, \mathbf{y}_{1:t}^*)$, which is set to the state transitional probability $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ for simplicity. In this case, the importance weight of particle i is updated by the observation likelihood as: $w_t^i = w_{t-1}^i p(\mathbf{y}_t^*|\mathbf{s}_t^i)$.

Particle filters have been extensively used in object tracking [41], and we also employ particle filters to track the target object. Similar to [29], we assume an affine motion model between consecutive frames. Therefore, the state variable $\mathbf{s}_t$ consists of the six affine transformation parameters (2D linear transformation and translation). By applying an affine transformation using $\mathbf{s}_t$ as parameters, we crop the region of interest $\mathbf{y}_t^*$ from the image and normalize it to the same size as the target templates in our dictionary. The state transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ is modeled to be Gaussian, with the dimensions of $\mathbf{s}_t$ assumed independent. The observation model $p(\mathbf{y}_t^*|\mathbf{s}_t)$ reflects the similarity between a target candidate and target templates in the dictionary. In this work, $p(\mathbf{y}_t^*|\mathbf{s}_t)$ is inversely proportional to the reconstruction error obtained by linearly representing $\mathbf{y}_t^*$ using the template dictionary.

5.4 Tracking by Occlusion Detection (TOD)

Occlusion is one of the most important challenges for visual tracking. In this section, we give a detailed description of our particle filter based tracking method, which makes use of occlusion prior information in a structured sparse learning framework to represent particle samples.

5.4.1 Occlusion Detection via Structured Sparsity

Occlusion detection is very important and difficult for tracking. In this section, we show how we incorporate a sparsity-inducing norm that also encodes prior structural information (spatial contiguity) regarding the support of the error incurred when sparse linear representation is used to describe particles. We expect that such structural information renders a more accurate and robust representation model that can handle occlusions in object tracking. In our particle filter based tracking method, particles are randomly sampled around the states of the tracked object according to a zero-mean Gaussian distribution. Similar to [29], we assume an affine motion model between consecutive frames. Therefore, the state of a particle $\mathbf{s}_t$ consists of the six affine transformation parameters (2D linear transformation and translation). By applying an affine transformation based on $\mathbf{s}_t$, we crop the region of interest $\mathbf{y}_t^*$ from the image and normalize it to the same size as the target templates in our dictionary. The state transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ is modeled to be a zero-mean Gaussian, with the dimensions of $\mathbf{s}_t$ independent. The observation model $p(\mathbf{y}_t^*|\mathbf{s}_t)$ reflects the similarity between a particle and target templates in the dictionary. In this chapter, $p(\mathbf{y}_t^*|\mathbf{s}_t)$ is inversely proportional to the reconstruction error obtained by linearly representing $\mathbf{y}_t^*$ using the template dictionary.

We sample n particles at each frame, and the observation (pixel color values) of the ith particle is denoted in vector form as: $\mathbf{x} \in \mathbb{R}^d$ (for simplicity, we ignore the subscript i). The observation $\mathbf{x}$ of a particle is represented as a sparse linear combination $\mathbf{z}$ of m dictionary templates $\mathbf{D} \in \mathbb{R}^{d\times m}$, as shown in Eq. (5.1). $\mathbf{D}$ is updated dynamically to handle frame-to-frame changes in target appearance (The dictionary update issue is addressed later). The popular L1 tracking work [29], which represents each particle by solving an ℓ_1 LASSO problem, can be generalized as shown in Eq. (5.1).

$$\min_{\mathbf{z},\mathbf{e}} \|\mathbf{z}\|_1 + \varphi(\mathbf{e}) \quad \text{s.t.} \quad \mathbf{x} = \mathbf{D}\mathbf{z} + \mathbf{e}, \tag{5.2}$$

In the L1 tracker, the regularizer $\varphi(\bullet)$ on $\mathbf{e}$ is chosen to be $\|\mathbf{e}\|_1$. This regularization scheme encourages the error (e.g., occlusion) to be pixel-wise sparse. This assumption fails in many tracking scenarios as exemplified in Fig. 5.1. It also does not incorporate the structural information inherent to occlusion, namely spatial contiguity. Basically, the ℓ_1-norm regularization treats each entry (pixel) in $\mathbf{e}$ independently.

It does not take into account any specific structures or possible relations among subsets of the entries. To encode this structured prior information, we assume that the spatial support of the error is contiguous. This can be enforced by modeling the error as spatially smooth. Also, this error can be assumed to be sparse, if any significant occlusion is detected and removed beforehand. Note that we assume that some pixels in a particle are occluded and those are determined by an occlusion mask that is propagated from frame-to-frame. At every frame, this mask is used to determine the pixels, from which the particle representation $\mathbf{z}$ is computed. This representation is used to estimate the error at *each* pixel. By thresholding this error with a predefined threshold, the occlusion mask is updated and propagated to the next frame.

To incorporate pairwise relationships between pixels in the particle, we adopt a graph-guided fused LASSO framework that explicitly takes into account the complex dependency structure represented as a graph, whose nodes are pixels in the particle. We assume that the d pixels in each particle are organized in a graph G with a set of nodes V and edges E. In this chapter, we adopt a simple strategy for constructing such a graph, whereby an edge exists between any pair of neighboring pixels and its weight is proportional to the correlation of their intensity values and inversely proportional to the Euclidean distance between them. More sophisticated methods can be employed, but they are not the focus of this chapter. Let $\mathbf{w}_{ml}$ denote the weight of an edge $(m, l) \in E$ that represents the strength of correlation between pixels m and l. Therefore, to encourage spatial contiguity between particle pixels, we employ a graph-guided fusion penalty, which extends the standard LASSO by fusing the $\mathbf{e}_m$ and $\mathbf{e}_l$ if $(m, l) \in E$. With the above notation, we formulate the representation problem as a structured sparse ℓ_1 problem as follows.

$$\begin{aligned} &\min_{\mathbf{z},\mathbf{e}} \|\mathbf{z}\|_1 + \lambda\|\mathbf{e}\|_1 + \gamma \sum_{(m,l)\in E} \mathbf{w}_{ml}\|\mathbf{e}_m - \mathbf{e}_l\|_1 \\ &\text{s.t.} \quad \mathbf{x} = \mathbf{D}\mathbf{z} + \mathbf{e}, \end{aligned} \tag{5.3}$$

where λ and γ are trade-off parameters that control the complexity of the model. A larger value for γ leads to a greater fusion effect. The $\mathbf{w}_{ml}$ weighs the fusion penalty for each edge such that $\mathbf{e}_m$ and $\mathbf{e}_l$ for highly correlated pixels have a large $\mathbf{w}_{ml}$. Details of solving this problem are provided in Sect. 5.5.

Discussion: In this chapter, we propose a generic formulation for robust object tracking using structured sparse learning as shown in Eq. (5.3). By defining γ differently, different object trackers are obtained. When $\gamma = 0$, TOD becomes the popular L_1 tracker [29]. In this way, the popular L_1 tracker [29] is a special case of our formulation. To the best of our knowledge, introducing the structured information in occlusion detection for tracking has not been proposed in any of the previous works. In Fig. 5.2, we present an example of how our TOD tracker works as compared to the L1 tracker. In the top row, we show a result of representing particle $\mathbf{x}$ using structured sparse learning instead of traditional sparse learning (used in L1 tracking), whose result is shown in the bottom row. Clearly, the error generated by TOD leads to a high response at the actual location of the occlusion, while it is missed by traditional

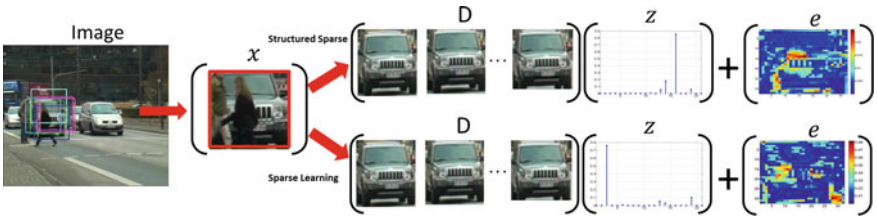

Fig. 5.2 Schematic example of TOD. The representation **z** of particle **x** w.r.t. dictionary **D** is learned by solving Eq. (5.3). Notice that **z** is sparse in general, i.e., a few dictionary templates are used to represent **x**. The *first row* is our TOD, and the *second row* is the popular L_1 tracker. Compared with the L_1 tracker, our methods can obtain much more continuous occlusion detection result

sparse learning. It is evident that by enforcing spatial contiguity on the error values, the occlusion can be better localized. This error is thresholded to produce an occlusion mask that is propagated to the next frame.

5.4.2 Dictionary Template Update

In the literature, a large body of work has been proposed to use object templates for visual tracking [27]. Target appearance remains the same only for a certain period of time, but eventually the object templates are no longer an accurate representation of its appearance. A fixed appearance template is not sufficient to handle changes in appearance due to occlusion or changes in illumination and pose. Also, if the templates are updated too often, small errors are introduced each time a template is updated, errors accumulate, and the tracker may drift from the target. Many approaches have been proposed over the years to address the drift problem [19, 28]. In this chapter, we do so by dynamically updating templates in **D**.

In order to initialize the object and background dictionaries, we sample equal-sized patches at and around the initial position of the object. In our experiments, we shift the initial bounding box by 1–3 pixels in each direction, thus resulting in $m = 20$ object templates as in [29]. Note that m is a user-defined parameter. All templates are normalized. To each object template, we allocate a weight ω_i that is indicative of how representative the template is. In fact, the more a template is used to represent tracking results, the higher is its weight. Next, we describe how we use these weights to update **D**. As mentioned earlier, the tracking result at instance t is the particle $\mathbf{z}_i$ that is best represented by **D** such that $i = \arg\min_{k=1,\ldots,n} (\|\mathbf{x}_k - \mathbf{D}\mathbf{z}_k\|_2)$. The weight of an object template in **D** is updated depending on how much that template is used in representing $\mathbf{z}_i$. If $\mathbf{z}_i$ is sufficiently represented (up to a predefined threshold) by the dictionary, then there is no need to update it. Otherwise, the current tracking result replaces the object template that has the smallest weight. The weight of this new template is set to the median of the current normalized weight vector $\boldsymbol{\omega}$. This template update scheme is summarized in Algorithm 1. We have two criteria: (1) The

Algorithm 1: *Dictionary Template Update*

Predefined threshold ε_1 and ε_2
$\mathbf{y}^*$ is the newly chosen tracking target and $\mathbf{z}_i$ its representation. Set $\Delta d_i = \|\mathbf{x}_i - \mathbf{D}\mathbf{z}_i\|_2$ and $sim_i = sim(\mathbf{D}, \mathbf{y}^*)$, where sim is the maximum similarity between $\mathbf{y}$ and all elements in $\mathbf{D}$.
$\boldsymbol{\omega}$ is the current weight vector of templates in $\mathbf{D}$
Update weights according to the coefficients of the target templates: $\omega_k \leftarrow \omega_k \exp(\mathbf{z}_i(k))$
if $(sim_i < \varepsilon_1 \ \& \ \Delta d_i > \varepsilon_2)$ **then**
  $r \leftarrow \arg\min_{k=1,\ldots,m_O} \omega_k$
  $\mathbf{D}(:, r) \leftarrow \mathbf{y}^*$, /*replace template with $\mathbf{y}^*$ */
  $\omega_r \leftarrow$ median($\boldsymbol{\omega}$), /*replace weight*/
end if
Normalize $\boldsymbol{\omega}$ such that $\|\boldsymbol{\omega}\|_1 = 1$

similarity sim_i between the current tracking result and template should be smaller than ε_1, which avoids updating templates frequently and thus avoids tracker drift; Once the current tracking result leads to a big variance, we add it to the dictionary by replacing it with the 'least' used dictionary template; (2) The error Δd_i should be smaller than ε_2, which means we update the dictionary template if only if there is no occlusion; In our experiments, ε_1 and ε_2 are set to be 0.6, and 0.7, respectively.

5.5 Optimization

In this section, we provide a detailed description of how Eq. (5.3) is solved efficiently. First, we rewrite the graph-guided fusion LASSO problem in Eq. (5.3), using a vertex-edge incident matrix $\mathbf{W} \in \mathbb{R}^{|E| \times d}$, as follows:

$$\sum_{(m,l) \in E} \mathbf{w}_{ml} \|\mathbf{e_m} - \mathbf{e_l}\|_1 = \|\mathbf{We}\|_1$$

where each row in $\mathbf{W}$ corresponds to an edge in the graph. If we label each edge with a linear index, we can define $\mathbf{W}$ formally as below:

$$\mathbf{W}_{j,k} = \begin{cases} \mathbf{w}_{ml} & if \quad j = (m,l) \quad \text{and} \quad k = m \\ -\mathbf{w}_{ml} & if \quad j = (m,l) \quad \text{and} \quad k = l \\ 0 & \text{otherwise} \end{cases}$$

Therefore, the overall penalty in Eq. (5.3) including both LASSO and graph-guided fusion penalty functions can be written as $\|\mathbf{Be}\|_1$, where $\mathbf{B} = [\lambda\mathbf{W}; \gamma\mathbf{I}]$ and $\mathbf{I} \in \mathbb{R}^{d \times d}$ denotes an identity matrix. Then, the structured sparsity problem in Eq. (5.3) is converted into the following problem:

$$\min_{\mathbf{z},\mathbf{e}} \|\mathbf{z}\|_1 + \|\mathbf{Be}\|_1 \quad s.t. \quad \mathbf{x} = \mathbf{Dz} + \mathbf{e} \tag{5.4}$$

To solve Eq. 5.4, we introduce two slack variables and add two equality constraints, thus converting it into Eq. (5.5).

$$\min_{z,e} \ \|\mathbf{z}_1\|_1 + \|\mathbf{f}\|_1 \tag{5.5}$$
$$\text{such that: } \mathbf{x} = \mathbf{D}\mathbf{z}_2 + \mathbf{e};\ \mathbf{z}_2 = \mathbf{z}_1;\ \mathbf{f} = \mathbf{B}\mathbf{e}$$

This transformed problem can be minimized using the conventional Inexact Augmented Lagrange Multiplier (IALM) method that has attractive quadratic convergence properties and is extensively used in matrix rank minimization problems [33]. IALM is an iterative method that augments the traditional Lagrangian function with quadratic penalty terms. This allows closed form updates for each of the unknown variables. By introducing augmented lagrange multipliers (ALM) to incorporate the equality constraints into the cost function, we obtain the Lagrangian function in Eq. (5.6) that we show, in what follows, can be optimized through a sequence of simple closed form update operations (refer to Eq. (5.7)).

$$\begin{aligned} &L(\mathbf{z}_{1-2}, \mathbf{y}_{1-3}, u_{1-3}) \\ &= \|\mathbf{z}_1\|_* + \|\mathbf{f}\|_1 \\ &\quad + tr\left[\mathbf{y}_1^T (\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e})\right] + \frac{u_1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e}\|_F^2 \\ &\quad + tr\left[\mathbf{y}_2^T (\mathbf{z}_2 - \mathbf{z}_1)\right] + \frac{u_2}{2}\|\mathbf{z}_2 - \mathbf{z}_1\|_F^2 \\ &\quad + tr\left[\mathbf{y}_3^T (\mathbf{f} - \mathbf{B}\mathbf{e})\right] + \frac{u_3}{2}\|\mathbf{f} - \mathbf{B}\mathbf{e}\|_F^2 \qquad (5.6) \\ &\Rightarrow \min_{\mathbf{z}_{1-2}, \mathbf{y}_{1-3}, u_{1-3}} L(\mathbf{z}_{1-2}, \mathbf{y}_{1-3}, u_{1-3}) \qquad (5.7) \end{aligned}$$

$\mathbf{y}_1$, $\mathbf{y}_2$, and $\mathbf{y}_3$ are lagrange multipliers, and $u_1 > 0$, $u_2 > 0$, and $u_3 > 0$ are three penalty parameters. The above problem can be solved by either exact or inexact ALM algorithms [24]. For efficiency, we choose the inexact ALM, whose details we outline in Algorithm (2). Its convergence properties can be proven similar to those in [24]. In fact, both IALM is an iterative algorithm that solves for each variable in a coordinate descent manner. In other words, each iteration of IALM involves the updating of each variable, with the other variables fixed to their most recent values. Consequently, we obtain five update steps corresponding to the five sets of variables we need to optimize for. Note that Steps 1–5 all have closed form solutions.

Step 1: [Update $\mathbf{z}_1$] Updating $\mathbf{z}_1$ requires the solution to the optimization problem in Eq. (5.8). This solution can be computed in closed form in Eq. (5.9), where $\mathcal{S}_\lambda\left(\mathbf{z}_{ij}\right) = sign(\mathbf{z}_{ij}) \max\left(0, |\mathbf{z}_{ij}| - \lambda\right)$ is the soft-thresholding operator, and $\mathbf{z}_{ij}$ is the jth element of vector $\mathbf{z}$.

$$\mathbf{z}_1^* = \arg\min_{\mathbf{z}_1} \frac{1}{u_1}\|\mathbf{z}_1\|_* + \frac{1}{2}\left\|\mathbf{z}_1 - \left(\mathbf{z}_2 + \frac{1}{u_2}\mathbf{y}_2\right)\right\|_F^2 \tag{5.8}$$

$$\Rightarrow \boxed{\mathbf{z}_1^* = \mathscr{S}_{\frac{1}{u_1}}\left(\mathbf{z}_2 + \frac{1}{u_2}\mathbf{y}_2\right)} \tag{5.9}$$

Step 2: [Update f] $\mathbf{f}$ is updated by solving the optimization problem in Eq. (5.10) with the closed form solution shown in Eq. (5.11).

$$\mathbf{f}^* = \arg\min_{\mathbf{f}} \frac{1}{u_3}\|\mathbf{f}\|_1 + \frac{1}{2}\left\|\mathbf{f} - \left(\mathbf{B}\mathbf{e} + \frac{1}{u_3}\mathbf{y}_3\right)\right\|_F^2 \tag{5.10}$$

$$\Rightarrow \boxed{\mathbf{f}^* = \mathscr{S}_{\frac{1}{u_3}}\left(\mathbf{B}\mathbf{e} + \frac{1}{u_3}\mathbf{y}_3\right)} \tag{5.11}$$

Step 3: [Update e] $\mathbf{e}$ is updated by solving the optimization problem in Eq. (5.12) with the closed form solution shown in Eq. (5.13).

$$\begin{aligned}\mathbf{e}^* = \arg\min_{\mathbf{e}} \; & tr[\mathbf{y}_1^t(\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e})] + \frac{u_1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e}\|_F^2 \\ & + tr[\mathbf{y}_3^t(\mathbf{B}\mathbf{e} - \mathbf{f})] + \frac{u_3}{2}\|\mathbf{B}\mathbf{e} - \mathbf{f}\|_F^2\end{aligned} \tag{5.12}$$

$$\Rightarrow \boxed{\mathbf{e}^* = \left(\mathbf{B}^T\mathbf{B} + \mathbf{I}\right)^{-1}\mathbf{G}}, \tag{5.13}$$

where $\mathbf{G} = \mathbf{x} - \mathbf{D}\mathbf{z}_2 + \frac{1}{u_1}\mathbf{y}_1 - \mathbf{B}^T\left(\frac{1}{u_3}\mathbf{y}_3 - \mathbf{f}\right)$.

Step 4: [Update $\mathbf{z}_2$**]** $\mathbf{z}_2$ is updated by solving the optimization problem in Eq. (5.14) with the closed form solution shown in Eq. (5.15).

$$\begin{aligned}\mathbf{z}_2{}^* = \arg\min_{\mathbf{z}_2} \; & tr[\mathbf{y}_1^t(\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e})] + \frac{u_1}{2}\|\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e}\|_F^2 \\ & + tr[\mathbf{y}_2^t(\mathbf{z}_2 - \mathbf{z}_1)] + \frac{u_2}{2}\|\mathbf{z}_2 - \mathbf{z}_1\|_F^2\end{aligned} \tag{5.14}$$

$$\Rightarrow \boxed{\mathbf{z}_2{}^* = \left(\mathbf{D}^T\mathbf{D} + \mathbf{I}\right)^{-1}\mathbf{G}}, \tag{5.15}$$

where $\mathbf{G} = \mathbf{D}^T(\mathbf{x} - \mathbf{e} + \frac{1}{u_1}\mathbf{y}_1) + \mathbf{z}_1 - \frac{1}{u_2}\mathbf{y}_2$.

Step 5: Update Multipliers $\mathbf{y}_1, \mathbf{y}_2$**:** We update the Lagrange multipliers in Eq. (5.16), where $\rho > 1$.

$$\begin{cases}\mathbf{y}_1 = \mathbf{y}_1 + u_1(\mathbf{x} - \mathbf{D}\mathbf{z}_2 - \mathbf{e}) \\ \mathbf{y}_2 = \mathbf{y}_2 + u_2(\mathbf{z}_2 - \mathbf{z}_1) \\ \mathbf{y}_3 = \mathbf{y}_3 + u_3(\mathbf{f} - \mathbf{B}\mathbf{e}) \\ u_1 = \rho u_1;\; u_2 = \rho u_2;\; u_3 = \rho u_3\end{cases} \tag{5.16}$$

Algorithm 2: *Structured sparse learning for occlusion detection (Solving Eq (5))*

Input : data $\mathbf{x}$, parameters λ, γ, and ρ
Output: $\mathbf{z}$

Initialize $\mathbf{z}_2 = \mathbf{0}$, $\mathbf{y}_1 = 0$, $\mathbf{y}_2 = 0$, $\mathbf{y}_3 = 0$
while *not converged* **do**
  fix other variables and update $\mathbf{z}_1$ [Eq (9)]
  fix other variables and update $\mathbf{f}$ [Eq (11)]
  fix other variables and update $\mathbf{e}$ [Eq (13)]
  fix other variables and update $\mathbf{z}_2$ [Eq (15)]
  update multipliers and parameters [Eq (16)]
  Update final solution $\mathbf{z} \leftarrow \mathbf{z}_2$
end

The IALM algorithm that solves Eq. (5.5) is shown in Algorithm (2), where convergence is reached when the change in objective function or solution $\mathbf{z}$ is below a user-defined threshold $\varepsilon = 10^{-3}$. Empirically, we find that our IALM algorithm is insensitive to a large range of ε values. In our implementation, $u_1 = u_2 = u_3$.

Computational Complexity: For the proposed TOD, it just uses the soft-thresholding operator, and is also very fast. This complexity is on par with that of other fast particle-based tracking algorithms. In comparison, the computational complexity of the L_1 tracker [29], which uses a sparse linear representation similar to our proposed tracker, is at least $\mathcal{O}\left(nd^2\right)$, since the number of dictionary templates (object and trivial) is $(m + 2d)$ and n Lasso problems are solved independently. Clearly, our method is more computationally attractive than L_1 tracker. When $m = 21$, $n = 400$, and $d = 32 \times 32$, the average per-frame run-time for TOD and L_1 trackers are about 5 s and 6 min, respectively.

5.6 Experimental Results

In this section, we do experimental results that validate the effectiveness and efficiency of our TOD method. We also make a thorough comparison between TOD and state-of-the-art tracking methods where applicable. We compile a set of 10 challenging tracking sequences to evaluate TOD. The sequences are sports videos and general videos include challenging appearance variations due to changes in pose, illumination, scale, and occlusion. Most of them involve various types of partial occlusions or multiple occlusions. We compare our TOD method to 6 recent and state-of-the-art trackers denoted as: L_1 [29], RCT [45], MIL [3], IVT [35], Frag [1], and OAB [14]. We implemented them using publicly available source codes or binaries provided by the authors. They were initialized using their default parameters. In our implementation, the initial position of the target is selected manually, and we shift the initial bounding box by 1–3 pixels in each dimension,

thus resulting in $m = 21$ target templates $\mathbf{D}$ (similar to L_1 tracker [29]). All our experiments are done using MATLAB on a 2.66 GHZ Intel Core2 Duo PC with 18 GB RAM. For all experiments, we model $p(\mathbf{s}_t|\mathbf{s}_{t-1}) \sim \mathcal{N}(\mathbf{0}, \text{diag}(\boldsymbol{\sigma}))$, where $\boldsymbol{\sigma} = [0.005, 0.0005, 0.0005, 0.005, 3, 3]^T$. We set the number of particles $n = 400$. In Algorithm 2, we set $\lambda = 1$ and $\gamma = 5$. Next, we give a qualitative and quantitative analysis of TOD, and compare it against the 6 baseline methods. Our experiments show that TOD produces more robust and accurate tracks.

5.6.1 Qualitative Comparison

The qualitative comparison results are shown in Figs. 5.3 and 5.4, which show tracking results of the 7 trackers on a subset of the videos. The details are introduced as follows. In the *AF1* sequence, a player is tracked with appearance changes due to camera motion. Tracking results for frames {10, 162, 300, 400} are presented in Fig. 5.3a. IVT and MIL start to drift around frame 162. Due to changes in appearance, OAB and L_1 start to undergo target drift from frame 300. Frag starts to fail after frame 400. RCT can track the target through the whole sequence; however, this tracker is not as robust or accurate as the TOD tracker. For the *AF2* sequence, the player is subject to changes in illumination and pose. Based on the results in Fig. 5.3b, OAB, RCT, and L_1 start to drift from the target at frame 200, while MIL and Frag drift at frame 277 and finally lose the target. IVT tracks the target quite well with a little drift. However, the target is successfully tracked throughout the entire sequence by TOD. In *So1* shown in Fig. 5.3c, a player with white color is tracked. The results at 4 frames are shown in Fig. 5.3c. Because there is only minor occlusion by other players, most of the methods can track the face accurately except Frag, which drifts around frame 170. The *So2* sequence contains abrupt object and camera motion with significant scale changes, which cause most of the trackers to drift as shown in Fig. 5.3d. TOD, L_1, and RCT handle these changes well. Compared with L_1, TOD obtains much better performance, which shows that harnessing local structure between pixels is useful for object tracking. In the *So3* sequence, tracking results for frames {1, 27, 92, 230} are presented in Fig. 5.3e. Frag and IVT start to drift around frame 27 and 92, respectively. Due to changes in lighting and camera motion, most of the trackers drift including L_1 and OAB. TOD, MTT, and RCT can track the target through the whole sequence; however, the proposed TOD tracker shows the best performance.

On the *faceocc2* sequence, the results are shown in Fig. 5.4a. Most trackers start drifting from the man's face when it is almost fully occluded by the book. Because the L_1 and TOD methods explicitly handle partial occlusions, and update the object dictionary progressively, they handle the appearance changes in this sequence very well. Fig. 5.4b shows tracking results for the *girl* sequence. Performance on this sequence exemplifies the robustness of TOD to occlusion (complete occlusion of the girl's face as she swivels in the chair) and large pose change (the face undergoes significant 3D rotation). TOD and L_1 are capable of tracking the target during the

Fig. 5.3 Tracking results of 7 methods on 5 sports video sequences. Frame numbers are denoted in *red* and the 7 tracking results (bounding boxes) are color-coded in each frame **a** AF1 **b** AF2 **c** So1 **d** So2 **e** So3

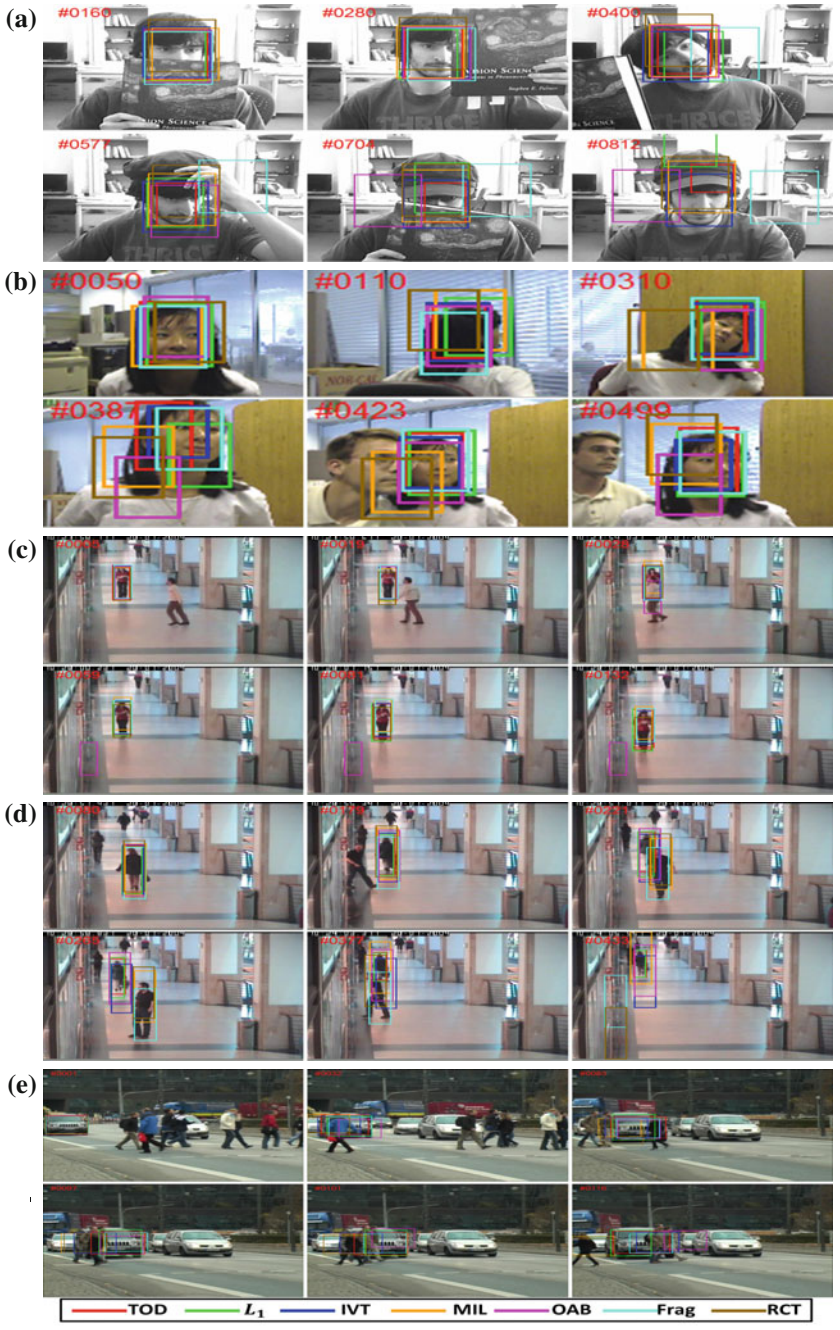

Fig. 5.4 Tracking results of 7 methods on 5 general video sequences. Frame numbers are denoted in *red* and the 7 tracking results (bounding boxes) are color-coded in each frame **a** faceocc2 **b** girl **c** onelsr1 **d** onelsr1 **e** tud_crossing

entire sequence. Other trackers experience drift at different time instances. Fig. 5.4c shows tracking results for the *onelsr1* sequence. In this sequence, partial occlusion happens, and it is much more easier. Therefore, many trackers (except OAB) can track the target through the whole video sequence. In the *onelsr2* sequence (refer to Fig. 5.4d), the walking woman is partially occluded by a walking man. IVT, MIL, Frag, OAB, and RCT lose the target woman, start tracking the man when partial occlusion occurs around frame 200, and are unable to recover from this failure. TOD and L_1 track the woman quite well. In the *tud_crossing* sequence, the target is severely occluded by multiple humans as shown in Fig. 5.4e. RCT and MIL start to drift around frame 32. Due to multiple occlusions, IVT starts to undergo target drift from frame 83. Other trackers, TOD, L_1, and Frag can track the target through the whole video; however, among all of the trackers, the TOD shows the best performance.

5.6.2 Quantitative Comparison

To give a fair quantitative comparison among the 7 trackers, we obtain manually labeled ground truth tracks for all the sequences. Most of the ground truth can be downloaded with the sequences. Tracking performance is evaluated according to the average per-frame distance (in pixels) between the center of the tracking result and that of ground truth as used in [3, 10, 29]. Clearly, this distance should be small. In Fig. 5.5, the average center distance for each tracker over the 10 sequences is plotted. TOD consistently outperform the other trackers in all sequences except for *AF2* and *onelsr1*, where they obtain very similar results to IVT. OAB is effected by background clutter and easily drifts from the target. MIL performs well except under severe illumination changes. RCT is not stable on several video sequences, especially those that contain occlusion and illumination variations. Frag and L_1 handle partial occlusion well, but tend to fail under severe illumination and pose changes. IVT is hardly affected by parameter settings and obtains good results in the absence of severe illumination changes. TOD can consistently produce a smaller distance than other trackers. This implies that TOD can accurately track the target despite severe occlusions and pose variations.

To demonstrate the effectiveness of our TOD, we compare it with the L_1 and RCT trackers, which are the most related trackers to ours based on sparse learning and have shown state-of-the-art performance [29, 45]. Based on the results in Fig. 5.5, TOD outperform the L_1 tracker and RCT. This is primarily due to the use of structure information for occlusion modeling, which makes TOD robust to occlusion problem. In addition, about the computational cost, TOD is much more efficient than L_1 as discussed in Sect. 5.5.

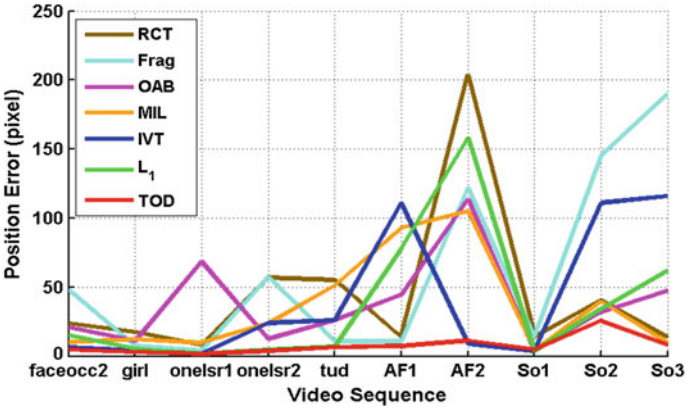

Fig. 5.5 Average distance of 7 trackers applied to 10 sequences

5.7 Conclusion

In this chapter, we propose a novel tracking method that allows for occlusion modeling and detection via structured sparse learning. By considering the structural information inherent to occlusion (e.g., spatial contiguity), the proposed TOD is much more robust for tracking under occlusion. The structured sparse learning problem is solved using an efficient IALM method. We show that the popular L_1 tracker [29] is a special case of our formulation. Also, we extensively analyze the performance of our tracker on challenging real-world video sequences and show that it outperforms 6 state-of-the-art trackers. In the future, we will do research on how to embed the temporal information to model occlusion in our framework.

Acknowledgments This study is supported by the research grant for the Human Sixth Sense Programme at the Advanced Digital Sciences Center from Singapore's Agency for Science, Technology, and Research (A*STAR).

References

1. Adam A, Rivlin E, Shimshoni I (2006) Robust fragments-based tracking using the integral histogram. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 798–805
2. Avidan, S (2005) Ensemble tracking. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 494–501
3. Babenko B, Yang M-H, Belongie S (2009) Visual tracking with online multiple instance learning. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 983–990
4. Bao C, Wu Y, Ling H, Ji H (2012) Real time robust l_1 tracker using accelerated proximal gradient approach. In: Proceedings of IEEE conference on computer vision and pattern recognition

5. Black MJ, Jepson AD (1998) Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. Int J Comput Vis 26(1):63–84
6. Blasch E, Kahler B (2005) Multiresolution eoir target tracking and identification. In: International conference on information fusion, vol 8, pp 1–8
7. Chockalingam P, Pradeep N, Birchfield S (2009) Adaptive fragmentsbased tracking of non-rigid objects using level sets. In: ICCV
8. Collins RT, Liu Y (2003) On-line selection of discriminative tracking features. In: Proceedings of the IEEE international conference on computer vision, pp 346–352
9. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. IEEE Trans Pattern Anal Mach Intell 25(5):564–575
10. Dinh TB, Vo N, Medioni G (2011) Context tracker: exploring supporters and distracters in unconstrained environments. In: Conference on computer vision and pattern recognition
11. Doucet A, De Freitas N, Gordon N (eds) (2001) Sequential Monte Carlo methods in practice. Springer, New York
12. Fleuret F, Berclaz J, Lengagne R, Fua P (2008) Multicamera people tracking with a probabilistic occupancy map. IEEE Trans PAMI 30(2):267–282
13. Gay-Bellile V, Bartoli A, Sayd P (2010) Direct estimation of nonrigid registrations with image-based self-occlusion reasoning. IEEE Trans PAMI 32(1):87–104
14. Grabner H, Grabner M, Bischof H (2006) Real-time tracking via on-line boosting. In: Proceedings of British machine vision conference, pp 1–10
15. Grabner H, Leistner C, Bischof H (2008) Semi-supervised on-line boosting for Robust tracking. In: Proceedings of European conference on computer vision, pp 234–247
16. Han B, Davis L (2005) On-line density-based appearance modeling for object tracking. In: ICCV
17. Jepson A, Fleet D, El-Maraghi T (2003) Robust on-line appearance models for visual tracking. IEEE Trans Pattern Anal Mach Intell 25(10):1296–1311
18. Jiang N, Liu W, Wu Y (2011) Adaptive and discriminative metric differential tracking. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1161–1168
19. Kaneko T, Hori O (2003) Feature selection for reliable tracking using template matching. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 796–802
20. Kwak S, Nam W, Han B, Han JH (2011) Learning occlusion with likelihoods for visual tracking. In: ICCV
21. Kwon J, Lee KM (2010) Visual tracking decomposition. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1269–1276
22. Leistner C, Godec M, Saffari A, Bischof H (2010) Online multi-view forests for tracking. In: DAGM, pp 493–502
23. Li H, Shen C, Shi Q (2011) Real-time visual tracking with compressed sensing. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1305–1312
24. Lin Z, Ganesh A, Wright J, Wu L, Chen M, Ma Y (2009) Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix. In: Technical Report UILU-ENG-09-2214, UIUC
25. Liu R, Cheng J, Lu H (2009) A Robust boosting tracker with minimum error bound in a co-training framework. In: Proceedings of the IEEE international conference on computer vision, pp 1459–1466
26. Liu B, Yang L, Huang J, Meer P, Gong L, Kulikowski C (2010) Robust and fast collaborative tracking with two stage sparse optimization. In: Proceedings of European conference on computer vision, pp 1–14
27. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision (DARPA). In: DARPA image understanding workshop, pp 121–130
28. Matthews I, Ishikawa T, Baker S (2004) The template update problem. IEEE Trans Pattern Anal Mach Intell 26:810–815
29. Mei X, Ling H (2011) Robust visual tracking and vehicle classification via sparse representation. IEEE Trans Pattern Anal Mach Intell 33(11):2259–2272

30. Mei X, Ling H, Wu Y, Blasch E, Bai L (2011) Minimum error bounded efficient l_1 tracker with occlusion detection. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1257–1264
31. Mittal A, Davis LS (2003) M2tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. Int J Comput Vis 51(3):189–203
32. Moeslund TB, Hilton A, Kruger V, Sigal L (2011) Visual analysis of humans
33. Peng Y, Ganesh A, Wright J, Xu W, Ma Y (2011) RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. IEEE Trans Pattern Anal Mach Intell 34(11):2233–2246
34. Ross D, Lim J, Yang M (2004) Adaptive probabilistic visual tracking with incremental subspace update. In: European conference on computer vision
35. Ross D, Lim J, Lin R-S, Yang M-H (2008) Incremental learning for Robust visual tracking. Int J Comput Vis 77(1):125–141
36. Salti S, Cavallaro A, Stefano LD (2012) Adaptive appearance modeling for video tracking: survey and evaluation. IEEE Trans Image Process 21(10):4334–4348
37. Wu Y, Lim J, M-H Yang (2013) Online object tracking: a benchmark. In: Proceedings of IEEE conference on computer vision and pattern recognition
38. Yang M, Wu Y, Hua G (2009) Context-aware visual tracking. IEEE Trans Pattern Anal Mach Intell 31(7):1195–1209
39. Yang M, Yuan J, Wu Y (2007) Spatial selection for attentional visual tracking. In: Conference on computer vision and pattern recognition
40. Yang M, Wu Y, Hua G (2009) Context-aware visual tracking. PAMI 31(1):1195–1209
41. Yilmaz A, Javed O, Shah M (2006) Object tracking: a survey. ACM Comput Surv 38(4):13+
42. Yin Z, Collins R (2008) Object tracking and detection after occlusion via numerical hybrid local and global mode-seeking. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1–8
43. Yu Q, Dinh TB, Medioni G (2008) Online tracking and reacquistion using co-trained generative and discriminative trackers. In: Proceedings of European conference on computer vision, pp 78–691 (2008)
44. Zhang T, Lu H, Li SZ (2009) Learning semantic scene models by object classification and trajectory clustering. In: CVPR
45. Zhang K, Zhang L, M-H Yang (2012) Real-time compressive tracking. In: Proceedings of European conference on computer vision
46. Zhang T, Ghanem B, Liu S, Ahuja N (2012) Low-rank sparse learning for robust visual tracking. In: European conference on computer vision
47. Zhang T, Ghanem B, Liu S, Ahuja N (2012) Robust visual tracking via multi-task sparse learning. In: Proceedings of IEEE conference on computer vision and pattern recognition
48. Zhang T, Liu J, Liu S, Xu C, Lu H (2011) Boosted exemplar learning for action recognition and annotation. IEEE Trans Circuits Syst Video Technol 21(7):853–866
49. Zhang T, Ghanem B, Liu S, Ahuja N (2013) Robust visual tracking via structured multi-task sparse learning. Int J Comput Vis 101(2):367–383
50. Zhang T, Liu S, Xu C, Lu H (2013) Mining semantic context information for intelligent video surveillance of traffic scenes. IEEE Trans Ind Inform 9(1):149–160
51. Zhong W, Lu H, M-H Y (2012) Robust object tracking via sparsity-based collaborative model. In: Proceedings of IEEE conference on computer vision and pattern recognition