

Robust Visual Tracking via Structured Multi-Task Sparse Learning

Tianzhu Zhang · Bernard Ghanem · Si Liu · Narendra Ahuja

Received: 2 April 2012 / Accepted: 30 September 2012 / Published online: 9 November 2012
© Springer Science+Business Media New York 2012

Abstract In this paper, we formulate object tracking in a particle filter framework as a structured multi-task sparse learning problem, which we denote as Structured Multi-Task Tracking (S-MTT). Since we model particles as linear combinations of dictionary templates that are updated dynamically, learning the representation of each particle is considered a single task in Multi-Task Tracking (MTT). By employing popular sparsity-inducing $\ell_{p,q}$ mixed norms (specifically $p \in \{2, \infty\}$ and $q = 1$), we regularize the representation problem to enforce joint sparsity and learn the particle representations together. As compared to previous methods that handle particles independently, our results demonstrate that mining the interdependencies between particles improves tracking

performance and overall computational complexity. Interestingly, we show that the popular L_1 tracker (Mei and Ling, IEEE Trans Pattern Anal Mach Intel 33(11):2259–2272, 2011) is a special case of our MTT formulation (denoted as the L_{11} tracker) when $p = q = 1$. Under the MTT framework, some of the tasks (particle representations) are often more closely related and more likely to share common relevant covariates than other tasks. Therefore, we extend the MTT framework to take into account pairwise structural correlations between particles (e.g. spatial smoothness of representation) and denote the novel framework as S-MTT. The problem of learning the regularized sparse representation in MTT and S-MTT can be solved efficiently using an Accelerated Proximal Gradient (APG) method that yields a sequence of closed form updates. As such, S-MTT and MTT are computationally attractive. We test our proposed approach on challenging sequences involving heavy occlusion, drastic illumination changes, and large pose variations. Experimental results show that S-MTT is much better than MTT, and both methods consistently outperform state-of-the-art trackers.

Electronic supplementary material The online version of this article (doi:10.1007/s11263-012-0582-z) contains supplementary material, which is available to authorized users.

T. Zhang
Advanced Digital Sciences Center (ADSC), 1 Fusionopolis Way,
#08-10 Connexis North Tower, Singapore 138632, Singapore
e-mail: tzhang10@gmail.com

B. Ghanem
King Abdullah University of Science and Technology (KAUST),
Al Khwarizmi Building #2224, Thuwal, Kingdom of Saudi Arabia
e-mail: bernard.ghanem@kaust.edu.sa

S. Liu (✉)
Department of Electrical and Computer Engineering,
National University of Singapore, 4 Engineering Drive 3,
Singapore 117576, Singapore
e-mail: dcsluis@nus.edu.sg

N. Ahuja
Department of Electrical and Computer Engineering, Beckman
Institute, and Coordinated Science Laboratory,
University of Illinois at Urbana-Champaign, 2041 Beckman Institute,
405 N. Mathews Ave., Urbana, IL 61801, USA
e-mail: ahuja@vision.ai.uiuc.edu

Keywords Visual tracking · Particle filter · Graph · Structure · Sparse representation · Multi-task learning

1 Introduction

The problem of tracking a target in video arises in many important applications such as automatic surveillance, robotics, human computer interaction, etc. For a visual tracking algorithm to be useful in real-world scenarios, it should be designed to handle and overcome cases where the target's appearance changes from frame-to-frame. Significant and rapid appearance variation due to noise, occlusion, varying

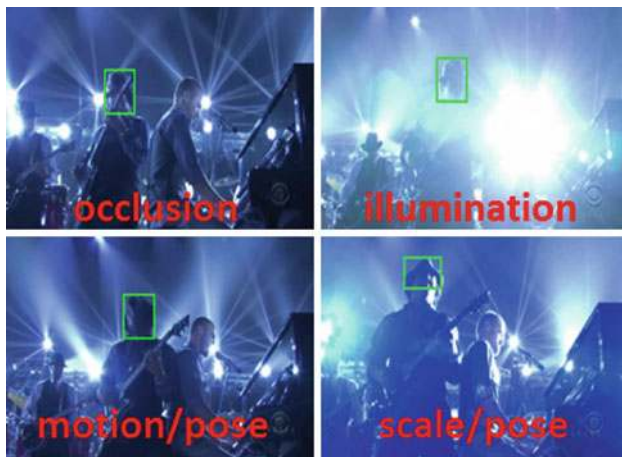


Fig. 1 (Color online) Frames from a *shaking* sequence. The ground truth track of the head is designated in *green*. Due to fast motion, occlusion, cluttered background, and changes in illumination, scale, and pose, visual object tracking is a difficult problem

viewpoints, background clutter, and illumination and scale changes pose major challenges to any tracker as shown in Fig. 1. Over the years, a plethora of tracking algorithms have been proposed to overcome these challenges. For a survey of many of these algorithms, we refer the reader to Yilmaz et al. (2006).

Recently, sparse representation (Candès et al. 2006) has been successfully applied to visual tracking (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010, 2011). In this case, the tracker represents each target candidate as a sparse linear combination of dictionary templates that can be dynamically updated to maintain an up-to-date target appearance model. This representation has been shown to be robust against partial occlusions, which leads to improved tracking performance. However, sparse coding based trackers perform computationally expensive ℓ_1 minimization at each frame. In a particle filter framework, computational cost grows linearly with the number of sampled particles. It is this computational bottleneck that precludes the use of these trackers in real-time scenarios. Consequently, very recent efforts have been made to speedup this tracking paradigm (Mei et al. 2011; Li et al. 2011). More importantly, these methods learn sparse representations of particles separately. Ignoring the relationships that ultimately constrain particle representations tend to make the tracker more prone to drifting away from the target, especially in cases of significant changes in appearance.

In this paper, we propose a computationally efficient multi-task sparse learning approach for visual tracking in a particle filter framework. Here, learning the representation of each particle is viewed as an individual task. Inspired by the above work, the next target state is selected to be the particle that has the highest similarity with a dictionary of target templates. Unlike previous methods, we exploit similarities among particles and, therefore, seek an accurate,

joint representation of these particles w.r.t. the dictionary. In our multi-task approach, particle representations are jointly sparse – only a few (but the same) dictionary templates should be used to represent all the particles at each frame. As opposed to sparse coding based trackers (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010, 2011) that handle particles separately, our use of joint sparsity incorporates the benefits of a sparse particle representation (e.g. partial occlusion handling), while respecting the underlying relationship between particles, which inherently yields a tracker that is more robust against various sources of appearance change. Therefore, we propose a multi-task formulation (denoted as Multi-Task Tracking or MTT) for the robust object tracking problem. We exploit interdependencies among the appearances of different particles to obtain their representations jointly. Joint sparsity is imposed on particle representations through an $\ell_{p,q}$ mixed-norm regularizer, which is optimized using an Accelerated Proximal Gradient (APG) method that guarantees fast convergence. In fact, joint sparsity can be viewed as a global form of structural regularization that influences all particle representations together. Furthermore, to extend the MTT framework to enforce local structure, we observe that some tasks (particle representations) are often more closely related and more likely to share common relevant covariates than other tasks. Therefore, we expand the MTT framework to consider pairwise structural correlations between particles (e.g. spatial smoothness of representation) and denote the novel framework as Structured Multi-Task Tracking abbreviated as S-MTT. A preliminary conference version of this work can be referred to in Zhang et al. (2012b).

Contributions: The contributions of this work are three-fold.

1. We propose a multi-task sparse learning method for object tracking, which is a robust sparse coding method that mines relationships between different tasks to obtain better tracking results than learning each task individually. This is done by exploiting both global and local structure among tasks. To the best of our knowledge, this is the first work to exploit multi-task learning in object tracking.
2. We show that the popular L_1 tracker (Mei and Ling 2011) is a special case of the proposed MTT framework.
3. Since we learn particle representations jointly, we can solve the S-MTT and MTT problems efficiently using an APG method. This makes our tracking method computationally attractive in general and significantly faster than the traditional L_1 tracker in particular.

The rest of the paper is organized as follows. In Sect. 2, we summarize the works most related to ours. The particle filter algorithm is reviewed in Sect. 3. Section 4 gives a detailed description of the proposed tracking approach, with

the optimization details presented in Sect. 4.3. Experimental results are reported and analyzed in Sect. 5. We conclude the paper in Sect. 6.

2 Related Work

Visual tracking is an important topic in computer vision and it has been studied for several decades. There is extensive literature on visual object tracking. In what follows, we only briefly review nominal tracking methods and those that are the most related to our own. We focus specifically on tracking methods that use particle filters and sparse representation, as well as, general multi-task learning methods. For a more thorough survey of tracking methods, we refer the readers to [Yilmaz et al. \(2006\)](#).

2.1 Object Tracking

In general, object tracking methods can be categorized as either generative or discriminative.

2.1.1 Generative Trackers

These methods adopt an appearance model to describe the target observations. Here, the aim of tracking is to search for the target location that has the most similar appearance to the generative model. Examples of generative methods are eigentracker ([Black and Jepson 1998](#)), mean shift tracker ([Comaniciu et al. 2003](#)), appearance model based tracker ([Jepson et al. 2003](#)), context-aware tracker ([Yang et al. 2009](#)), fragment-based tracker (Frag) ([Adam et al. 2006](#)), incremental tracker (IVT) ([Ross et al. 2008](#)), and VTD tracker ([Kwon and Lee 2010](#)). In [Black and Jepson \(1998\)](#), a view-based representation is used for tracking rigid and articulated objects. This approach builds on and extends work on eigenspace representations, robust estimation techniques, and parameterized optical flow estimation. The mean shift tracker ([Comaniciu et al. 2003](#)) is a popular mode-finding method, which successfully copes with camera motion, partial occlusions, clutter, and target scale variations. In [Jepson et al. \(2003\)](#), a robust and adaptive appearance model is learned for motion-based tracking of natural objects. The model adapts to slowly changing object appearance, and it maintains an acceptable measure of stability in the observed image structure during tracking. Moreover, the context-aware tracker ([Yang et al. 2009](#)) focuses on an object's context for robust visual tracking. Specifically, this method integrates into the tracking process a set of auxiliary objects that are automatically discovered in the video via data mining techniques. Furthermore, the tracking method proposed in [Ross et al. \(2008\)](#) incrementally learns a low-dimensional subspace representation, and efficiently adapts to online

changes in target appearance. To adapt to variations in appearance (e.g. due to changes in illumination and pose), the appearance model can be dynamically updated. The Frag tracker ([Adam et al. 2006](#)) aims to solve partial occlusion with a representation based on histograms of local patches. The tracking task is carried out by accumulating votes from matching local patches using a template. However, this template is not updated and, thus, it is not expected to handle changes in object appearance that can be due to scale and shape variations. In the IVT tracker ([Ross et al. 2008](#)), an adaptive appearance model is constructed to account for appearance variation due to rigid or limited deformable motion. Although it has been shown to perform well when target objects undergo lighting and pose variation, IVT is less effective in handling heavy occlusion or non-rigid distortion as a result of the adopted holistic appearance model. Finally, the VTD tracker ([Kwon and Lee 2010](#)) effectively extends the conventional particle filter framework with multiple motion and observation models to account for appearance variation caused by changes in pose, lighting, and scale as well as partial occlusion. Nevertheless, as a result of the adopted generative representation scheme, this tracker is not equipped to distinguish between the target and its context (background).

2.1.2 Discriminative Trackers

These methods formulate visual object tracking as a binary classification problem, which seeks the target location that can best separate the target from its background. Examples of discriminative methods are on-line boosting (OAB) ([Grabner et al. 2006](#)), semi-online boosting ([Grabner et al. 2008](#)), ensemble tracking ([Avidan 2005](#)), co-training tracking ([Liu et al. 2009](#)), online multi-view forests for tracking ([Leistner et al. 2010](#)), adaptive metric differential tracking ([Jiang et al. 2011](#)) and online multiple instance learning tracking ([Babenko et al. 2009](#)). In the OAB tracker ([Grabner et al. 2006](#)), online AdaBoost is adopted to select useful features for object tracking. Its performance is affected by background clutter, and the tracker can easily drift. The ensemble tracker ([Avidan 2005](#)) formulates the tracking 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 constrains its ability to handle heavy occlusion and clutter. In the MIL tracker ([Babenko et al. 2009](#)), the multiple instance learning method is extended to an online setting for object tracking. While it is capable of reducing tracker drift, this method is unable to handle large nonrigid shape deformation. In ensemble tracking ([Avidan 2005](#)), a feature vector is constructed for every pixel in the reference image and an adaptive ensemble of classifiers is trained to separate pixels that belong to the object from pixels that belong to the background. In [Collins and Liu \(2003\)](#), a target confidence map

is built by finding the most discriminative RGB color combination in each frame. Moreover, a hybrid approach that combines a generative model and a discriminative classifier is proposed in Yu et al. (2008) to capture appearance changes and allow reacquisition of an object after total occlusion. Global mode seeking can be used to detect and reinitialize the tracked object after total occlusion (Yin and Collins 2008). Yet another approach uses image fusion to determine the most discriminative appearance model and then a generative approach for dynamic target updates (Blasch and Kahler 2005).

2.2 Particle Filters for Object Tracking

Particle filters (also known as condensation or sequential Monte Carlo models) were introduced to visual tracking (Isard and Blake 1998). Since then and over the last decade, it has become a popular tracking framework due primarily to its excellent performance in the presence of nonlinear target motion and to flexibility to different object representations (Wu and Huang 2004). In general, when more particles are sampled and a better target representation is constructed, particle filter based tracking algorithms are more likely to perform reliably in cluttered and noisy environments. However, the computational cost of particle filter trackers tends to increase linearly with the number of particles. Consequently, researchers have proposed various means of speeding up the particle filter framework. In Yang et al. (2005), tracked objects are described using color and edge orientation histogram features, and the observation likelihood is computed in a coarse-to-fine manner, which allows the computation to quickly focus on the more promising regions. In Khan et al. (2004), subspace representations are used in a particle filter for tracking. This tracker is made efficient by applying Rao-Blackwellization to the subspace coefficients in the state vector. In Zhou et al. (2004), the number of particle samples is adjusted according to an adaptive noise component.

2.3 Sparse Representation for Object Tracking

Recently, sparse representation has been introduced to particle filter based object tracking and has yielded noteworthy performance (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010; Li et al. 2011; Bao et al. 2012; Zhang et al. 2012a). In Mei and Ling (2011), a tracking candidate is represented as a sparse linear combination of object templates and trivial templates. For each particle, sparse representation is computed by solving a constrained ℓ_1 minimization problem with non-negativity constraints, thus, solving the inverse intensity pattern problem during tracking. Although this method yields good tracking performance, it comes at the computational expense of multiple ℓ_1 minimization problems that are independently solved. In fact, the computational cost grows linear

with the number of particle samples. In Mei et al. (2011), 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, most of the irrelevant samples are removed before solving the computationally expensive ℓ_1 minimization function. In Liu et al. (2010), dynamic group sparsity is integrated into the tracking problem and high dimensional image features are used to improve tracking robustness. In Li et al. (2011), dimensionality reduction and a customized orthogonal matching pursuit algorithm are adopted to accelerate the L_1 tracker (Mei and Ling 2011). In Bao et al. (2012), APG based solution is used to improve the L_1 tracker (Mei and Ling 2011). In Zhang et al. (2012a), low-rank sparse learning is adopted to consider the correlations among particles for robust tracking. Inspired by these works, we should solve two problems, which are how to consider the correlations among particles and how to make the tracker be fast. Therefore, we propose the S-MTT tracking method.

2.4 Multi-Task Learning

Multi-task learning (MTL, Chen et al. 2009) has recently received much attention in machine learning and computer vision. It capitalizes on shared information between related tasks to improve the performance of each individual task, and it has been successfully applied to popular vision problems such as image classification [(Yuan and Yan 2010) and image annotation (Quattoni et al. 2009)]. The underlying assumption behind many MTL algorithms is that the tasks are related. Thus, a key issue lies in how relationships between tasks are incorporated in the learning framework. Inspired by the above works, we want to improve computational efficiency and capitalize on the interdependence among particle appearances (for additional robustness in tracking). To make this come true, we propose a multi-task sparse representation method for robust object tracking.

3 Particle Filter

The particle filter (Doucet et al. 2001) is a Bayesian sequential importance sampling technique, which recursively approximates the posterior distribution using a finite set of weighted samples 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. location or motion parameters) and its observation respectively. The prediction stage uses the probabilistic system transition model $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ to predict the posterior distribution of \mathbf{s}_t given all available observations $\mathbf{y}_{1:t-1} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{t-1}\}$ up to time $t - 1$ is computed in Eq. (1).

$$p(\mathbf{s}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{s}_t|\mathbf{s}_{t-1})p(\mathbf{s}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{s}_{t-1} \quad (1)$$

At time t , the observation \mathbf{y}_t is available and the state vector is updated using Bayes rule, as in Eq. (2), where $p(\mathbf{y}_t|\mathbf{s}_t)$ denotes the observation likelihood.

$$p(\mathbf{s}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{s}_t)p(\mathbf{s}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})} \quad (2)$$

In the particle filter framework, the posterior $p(\mathbf{s}_t|\mathbf{y}_{1:t})$ is approximated by a finite set of n samples $\{\mathbf{s}_t^i\}_{i=1}^n$ (called particles) with importance weights w_i . The particle samples \mathbf{s}_t^i are drawn from an importance distribution $q(\mathbf{s}_t|\mathbf{s}_{1:t-1}, \mathbf{y}_{1:t})$ and the importance weights are updated according to Eq. (3).

$$w_t^i = w_{t-1}^i \frac{p(\mathbf{y}_t|\mathbf{s}_t^i) p(\mathbf{s}_t^i|\mathbf{s}_{t-1}^i)}{q(\mathbf{s}_t|\mathbf{s}_{1:t-1}, \mathbf{y}_{1:t})} \quad (3)$$

To avoid degeneracy, particles are resampled according to the importance weights so as to generate a set of equally weighted particles. For simplicity, in the case of the bootstrap filter (Doucet et al. 2001), we set $q(\mathbf{s}_t|\mathbf{s}_{1:t-1}, \mathbf{y}_{1:t}) = p(\mathbf{s}_t|\mathbf{s}_{t-1})$, so that the weights are updated by the observation likelihood $p(\mathbf{y}_t|\mathbf{s}_t)$.

Particle filters have been used extensively in object tracking (Yilmaz et al. 2006). In this paper, we also employ particle filters to track the target object. Similar to Mei and Ling (2011), we assume an affine motion model between consecutive frames. Therefore, the state variable \mathbf{s}_t consists of the six parameters of the affine transformation (2D linear transformation and a 2D 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 size of 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 (particle) and dictionary templates. In this paper, $p(\mathbf{y}_t|\mathbf{s}_t)$ is inversely proportional to the reconstruction error obtained by linearly representing \mathbf{y}_t using the dictionary of templates.

4 Structured Multi-Task Tracking (S-MTT)

In this section, we give a detailed description of our particle filter based tracking method that makes use of structured multi-task learning to represent particle samples.

4.1 Structured Multi-Task Representation of Particles

In the MTL framework, tasks that share dependencies in features or learning parameters are jointly solved in order to capitalize on their inherent relationships. Many works in this domain have shown that MTL can be applied to classical problems [(e.g. image annotation (Quattoni et al. 2009) and image classification (Yuan and Yan 2010))] and outperform state-of-the-art methods that resort to independent learning. In this paper, we formulate object tracking as an MTL problem, where learning the representation of a particle is viewed as a single task. Usually, particle representations in tracking are computed separately (e.g. L_1 tracker, Mei and Ling 2011). In this paper, we show that by representing particles jointly in an MTL setting, tracking performance and tracking speed can be significantly improved.

In our particle filter based tracking method, particles are randomly sampled around the current state of the tracked object according to a zero-mean Gaussian distribution. At instance t , we consider n particle samples, whose observations (pixel color values) in the t th frame are denoted in matrix form as: $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$, where each column is a particle in \mathbb{R}^d . In the noiseless case, each particle \mathbf{x}_i is represented as a linear combination \mathbf{z}_i of templates that form a dictionary $\mathbf{D}_t = [\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_m]$, such that $\mathbf{X} = \mathbf{D}_t\mathbf{Z}$. The dictionary columns comprise the templates that will be used to represent each particle. These templates include visual observations of the tracked object (called target templates) possibly under a variety of appearance changes. Since our representation is constructed at the pixel level, misalignment between dictionary templates and particles might lead to degraded performance. To alleviate this problem, one of two strategies can be employed. (i) \mathbf{D}_t can be constructed from a dense sampling of the target object, which can also include transformed versions of these samples. (ii) Columns of \mathbf{X} can be aligned to columns of \mathbf{D}_t as in Peng et al. (2012) to solve the geometric transformation. In this paper, we employ the first strategy, which leads to a larger m but a lower overall computational cost. We denote \mathbf{D}_t with a subscript because the dictionary templates will be progressively updated to incorporate variations in object appearance due to changes in illumination, viewpoint, etc. Our dictionary update scheme is adopted from the work in Mei and Ling (2011), but for completeness, we present its details in Sect. 4.4.

In many visual tracking scenarios, target objects are often corrupted by noise or partially occluded. As in Mei and Ling (2011), this noise can be modeled as sparse additive noise that can take on large values anywhere in its support. Therefore, in the presence of noise, we can still represent the particle observations \mathbf{X} as a linear combination of templates, where the dictionary is augmented with trivial (or occlusion) templates \mathbf{I}_d (identity matrix of $\mathbb{R}^{d \times d}$), as shown in Eq. (4). The representation error \mathbf{e}_i of particle i using dictionary \mathbf{D}_t is the

i th column in \mathbf{E} . The nonzero entries of \mathbf{e}_i indicate the pixels in \mathbf{x}_i that are corrupted or occluded. The nonzero support of \mathbf{e}_i can be different among particles and is unknown a priori.

$$\mathbf{X} = [\mathbf{D}_t \ \mathbf{I}_d] \begin{bmatrix} \mathbf{Z} \\ \mathbf{E} \end{bmatrix} \Rightarrow \boxed{\mathbf{X} = \mathbf{BC}} \quad (4)$$

4.1.1 Imposing Joint Sparsity via $\ell_{p,q}$ Mixed-Norm

Because most particles are densely sampled around the current target state, their representations with respect to \mathbf{D}_t will be sparse (few templates are required to represent them) and similar to each other (the support of particle representations is similar) in general. These two properties culminate in individual particle representations (single tasks) being *jointly sparse*. In other words, joint sparsity will encourage all particle representations to be individually sparse *and* share the same (few) dictionary templates that reliably represent them. This yields a more robust representation for the ensemble of particles. In fact, joint sparsity has been recently employed to address MTL problems (Quattoni et al. 2009; Yuan and Yan 2010). A common technique to explicitly enforce joint sparsity in MTL is the use of sparsity-inducing norms to regularize the parameters shared among the individual tasks. In this paper, we investigate the use of convex $\ell_{p,q}$ mixed norms (i.e. $p \geq 1$ and $q \geq 1$) to address the problem of MTL in particle filter based tracking (denoted as MTT). Therefore, we need to solve the convex optimization problem in Eq. (5), where λ is a tradeoff parameter between reliable reconstruction and joint sparsity regularization.

$$\min_{\mathbf{C}} \frac{1}{2} \|\mathbf{X} - \mathbf{BC}\|_F^2 + \lambda \|\mathbf{C}\|_{p,q} \quad (5)$$

Note that we define $\|\mathbf{C}\|_{p,q}$ as in Eq. (6), where $\|\mathbf{C}_i\|_p$ is the ℓ_p norm of \mathbf{C}_i and \mathbf{C}_i is the i th row of matrix \mathbf{C} .

$$\|\mathbf{C}\|_{p,q} = \left(\sum_{i=1}^{m+d} (\|\mathbf{C}_i\|_p)^q \right)^{1/q} \quad (6)$$

As in Eq. (5), given a dictionary \mathbf{B} , for the n tasks $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$ (each column is a particle), we aim to discover, across these n tasks, a few common templates that are the most useful for particle representation. In this setting, the constraint of joint sparsity across different tasks is valuable since different tasks may favor different sparse reconstruction coefficients, yet the joint sparsity enforces the robustness in coefficient estimation. Moreover, joint sparsity exploits correlations among different tasks to obtain better generalization performance as compared to learning each task individually. The goal of joint sparsity is achieved by imposing an $\ell_{p,q}$ mixed-norm penalty on the reconstruction coefficients. In fact, joint sparsity can be viewed as a global form of structural regularization that influences all particle

representations together. In the next section, we extend the MTT framework to enforce local structure as well.

4.1.2 Imposing Structure via Graph Regularization

Enforcing joint sparsity using the $\ell_{p,q}$ mixed-norm exploits the global structure inherent to particle representations in any given frame. However, in particle based MTT, some of the tasks are often more closely related and more likely to share common relevant covariates than other tasks. This induces another layer of structure, which affects particle representations *locally*. Therefore, we expand the MTT framework to consider pairwise structural correlations between particles (e.g. spatial smoothness of representation) and denote the novel framework as Structured Multi-Task Tracking abbreviated as S-MTT. The S-MTT formulation can be viewed as a generalization of MTT, since local structural information endows MTT with another layer of robustness in tracking. In fact, our experiments show that incorporating such local structural information significantly improves the performance of MTT.

We assume that the learned representations \mathbf{C} can be related through pairwise interactions, which are considered local structural priors to \mathbf{C} . In this paper, we use these structural priors to enforce spatial smoothness among particle representations. In other words, particles that are spatially located near each other in the same frame should have similar representations. In general, higher order relationships can be added to the S-MTT framework; however, such relationships significantly increase the complexity of learning the optimal \mathbf{C} . Therefore, we restrict ourselves to pairwise relationships that are defined as edges in a graph, whose nodes constitute the particle representations (i.e. columns of \mathbf{C}). As such, we incorporate these local pairwise relationships into Eq. (5) by adding a suitable graph regularization term.

To do this, we investigate the use of the well-known and widely used, normalized graph smoothness regularizer, which is a weighted sum of pairwise distances between the normalized representations in \mathbf{C} . The weight of each distance term reflects how strongly the corresponding pairwise relationship should be enforced. Since the normalized version of this regularizer has been shown to produce better and more stable results in many learning problems, we prefer it over its unnormalized counterpart (Zhu 2008).

In Eq. (7), we formalize the graph regularizer, denoted as $G(\mathbf{C})$. Here, we define a symmetric weight matrix $\mathbf{W} \in \mathbb{R}_+^{n \times n}$ that describes the similarity measure between every pair of particle representations. In fact, \mathbf{W} represents the weights of all edges in the graph. Therefore, \mathbf{W}_{ij} is the similarity measure between the i th particle \mathbf{c}_i and the j th particle \mathbf{c}_j . Here, we denote $\hat{d}_i = \sum_{j=1}^n \mathbf{W}_{ij}$, the sum of the elements of the i th row of \mathbf{W} , and $\hat{\mathbf{D}} = \text{diag}(\hat{d}_1, \hat{d}_2, \dots, \hat{d}_n)$. In graph

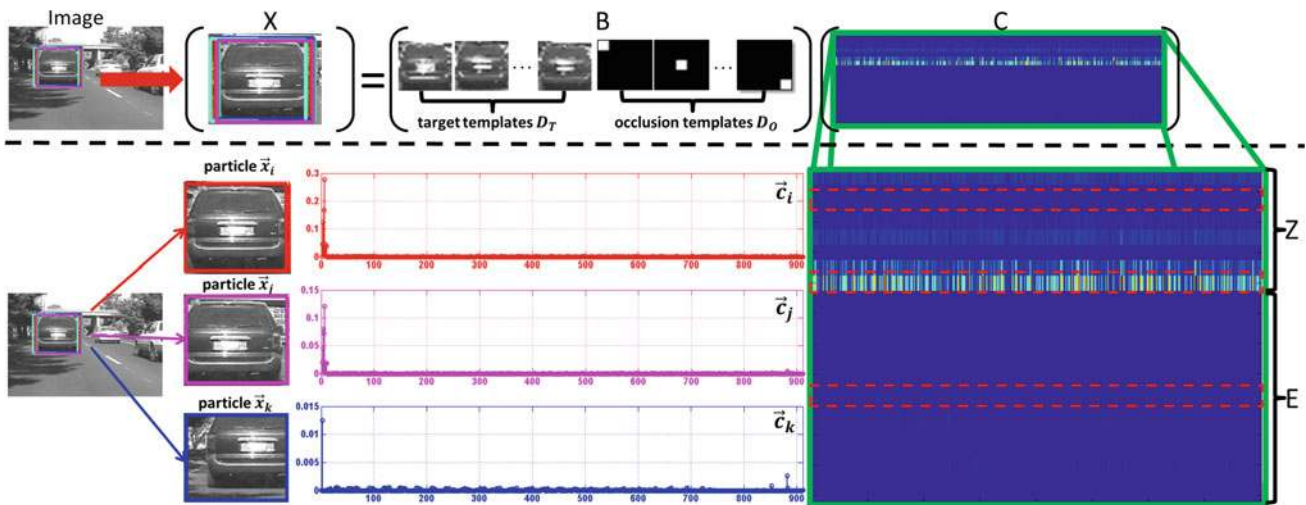


Fig. 2 (Color online) Schematic example of the L_{21} tracker. The representation \mathbf{C} of all particles \mathbf{X} w.r.t. dictionary \mathbf{B} (set of target and occlusion templates) is learned by solving Eq. (9) with $p = 2$ and $q = 1$. Notice that the columns of \mathbf{C} are jointly sparse, i.e. a few (but

the same) dictionary templates are used to represent all the particles together. The particle \mathbf{x}_i is selected among all other particles as the tracking result, since \mathbf{x}_i is represented the best by object templates only

theory, $\hat{\mathbf{D}}$ is called the degree of the graph. In the last step, we denote $\mathbf{L} = \hat{\mathbf{D}} - \mathbf{W}$ as the Laplacian of the graph and $Tr(\mathbf{A})$ as the trace of matrix \mathbf{A} .

$$G(\mathbf{C}) = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \mathbf{W}_{ij} \left\| \frac{\mathbf{c}_i}{\sqrt{\hat{d}_i}} - \frac{\mathbf{c}_j}{\sqrt{\hat{d}_j}} \right\|_2^2 = Tr(\mathbf{C}\hat{\mathbf{L}}\mathbf{C}^T) \text{ where } \hat{\mathbf{L}} = \hat{\mathbf{D}}^{-\frac{1}{2}}\mathbf{L}\hat{\mathbf{D}}^{-\frac{1}{2}} \quad (7)$$

We define \mathbf{W}_{ij} to decrease exponentially with the distance between the spatial locations of the i th and j th particles, as in Eq. (8). Here, we denote \mathbf{l}_i as the 2D location of the center of the i th particle in the current frame and δ as a smoothing factor. The δ is the average value of all distances between \mathbf{l}_i and \mathbf{l}_j . Note that other similarity measures can be used to describe \mathbf{W}_{ij} including the PASCAL overlap score.¹

$$\mathbf{W}_{ij} = \exp\left(-\frac{\|\mathbf{l}_i - \mathbf{l}_j\|_2^2}{2\delta^2}\right) \quad (8)$$

Therefore, the particle representations \mathbf{C} can be computed by solving Eq. (9), which simply adds the graph regularizer $G(\mathbf{C})$ to Eq. (5). Here, $\hat{\mathbf{L}}$ is the normalized Laplacian matrix, and λ_1 and λ_2 are two parameters that quantify the tradeoff between local and global structural regularization.

$$\min_{\mathbf{C}} \frac{1}{2} \|\mathbf{X} - \mathbf{BC}\|_F^2 + \frac{\lambda_1}{2} Tr(\mathbf{C}\hat{\mathbf{L}}\mathbf{C}^T) + \lambda_2 \|\mathbf{C}\|_{p,q} \quad (9)$$

4.2 Discussion

To encourage a sparse number of dictionary templates to be selected for all particles, we restrict our choice of $\ell_{p,q}$ mixed norms to the case of $q=1$, thus, $\|\mathbf{C}\|_{p,1} = \sum_{i=1}^{m+d} \|\mathbf{C}_i\|_p$. Among its convex options, we select three popular and widely studied $\ell_{p,1}$ norms: $p \in \{1, 2, \infty\}$. The S-MTT objective in Eq. (9) is composed of a convex² quadratic term and a non-smooth regularizer, and thus we conventionally adopt the APG method (Tseng 2008) for optimization. The solution to Eq. (9) for these choices of p and q is described in Sect. 4.3. Note that each choice of p yields a different S-MTT tracker, which we will denote as the L_{p1}^* tracker. To discriminate between S-MTT and MTT trackers, we denote the MTT tracker (i.e. when $\lambda_1 = 0$ in Eq. (9)) using the $\ell_{p,1}$ mixed norm as the L_{p1} tracker. In Sect. 5.6, we show that L_{p1}^* trackers can lead to significant improvement in tracking performance over L_{p1} trackers, in general. In Fig. 2, we present an example of how the L_{21} tracker works. Given all particles \mathbf{X} (sampled around the tracked car) and based on a dictionary \mathbf{B} , we learn the representation matrix \mathbf{C} by solving Eq. (9). Note that smaller values are darker in color. Clearly, columns of \mathbf{C} are jointly sparse, i.e. a few (but the same) dictionary templates are used to represent all the particles together. Particle \mathbf{x}_i is chosen as the current tracking result \mathbf{y}_t because it has the smallest reconstruction error w.r.t. to the target templates \mathbf{D}_t . Since particles \mathbf{x}_j and \mathbf{x}_k are misaligned

¹ The score is the ratio of the intersection to the union of two bounding boxes. In our case, it would be the ratio of the intersection of the ground truth and the predicted tracks to their union in each frame.

² Since the degree matrix $\hat{\mathbf{D}}$ is diagonal and non-negative and since the Laplacian \mathbf{L} of any graph is positive semi-definite, the normalized Laplacian $\hat{\mathbf{L}}$ is positive semi-definite. Thus, $G(\mathbf{C})$ is convex in \mathbf{C} .

versions of the car, they are not represented well by \mathbf{D}_t (i.e. \mathbf{z}_j and \mathbf{z}_k have small values). This precludes the tracker from drifting into the background.

As for Eq. (6), when $p = q = 1$, we note that

$$\|\mathbf{C}\|_{1,1} = \sum_{i=1}^{m+d} \|\mathbf{C}_i\|_1 = \sum_{i=1}^n \|\mathbf{c}_i\|_1, \quad (10)$$

where \mathbf{c}_i and \mathbf{C}_i represent the i th column and i th row in \mathbf{C} respectively. This equivalence property between rows and columns (i.e. the sum of the ℓ_p norms of rows and that of columns are the same) only occurs when $p = 1$. In this case, MTT (Eq. (5)) and S-MTT (Eq. (9) when $\lambda_1 = 0$) become equivalent to Eq. (10). This optimization problem is no longer an MTL problem, since the n representation tasks are unrelated and are solved separately. Interestingly, Eq. (10) is the same formulation used in the popular L_1 tracker (Mei and Ling 2011), which can be viewed as a special case of our proposed family of S-MTT algorithms, namely the L_{11} tracker. In fact, using the optimization technique in Sect. 4.3, we observe that our L_{11} implementation is two orders of magnitude faster than the traditional L_1 tracker.

$$\min_{\mathbf{c}_1, \dots, \mathbf{c}_n} \sum_{j=1}^n \left(\frac{1}{2} \|\mathbf{x}_j - \mathbf{B}\mathbf{c}_j\|_2^2 + \lambda_2 \|\mathbf{c}_j\|_1 \right) \quad (11)$$

A detailed overview of the proposed S-MTT tracking method is shown in Algorithm 1. Based on the previous state \mathbf{s}_{t-1} , we can use the importance sampling approach (Isard and Blake 1998) to obtain new particles and crop the corresponding image patches to obtain their observations \mathbf{X} . Then, we learn their representations \mathbf{C} by solving Eq. (9), to be shown in Sect. 4.3. The particle that has the smallest

Algorithm 1: Structured Multi-Task Tracking Algorithm.

Input:

- Current frame at t
- Dictionary template \mathbf{D}_{t-1}
- All n particles \mathbf{s}_{t-1}

Output:

- Tracked target \mathbf{y}_t .
 - Current states \mathbf{s}_t .
 - Updated target templates \mathbf{D}_t .
- 1: Generate n particles \mathbf{s}_t with the importance sampling approach [16]
 - 2: Obtain mapped observations corresponding to all particles \mathbf{s}_t to obtain \mathbf{X}
 - 3: Compute structured multi-task sparse representation $\mathbf{C} = \begin{bmatrix} \mathbf{Z} \\ \mathbf{E} \end{bmatrix}$ by solving Eq (9)
 - 4: Calculate the reconstruction error $\Delta r_i = \|\mathbf{x}_i - \mathbf{D}_{t-1}\mathbf{z}_i\|_2$, $i = 1, 2, \dots, n$
 - 5: Calculate $p(\mathbf{y}_i | \mathbf{s}_i) = \exp(-\Delta r_i^2)$ for each particle
 - 6: Select the particle with the highest value of $p(\mathbf{y}_i | \mathbf{s}_i)$ as the current tracking result \mathbf{y}_t
 - 7: Update dictionary template \mathbf{D}_t as shown in Section 4.4
-

reconstruction error is selected to be the current tracking result. Finally, the dictionary templates in \mathbf{D}_t are updated adaptively, to be shown in Sect. 4.4.

4.3 Solving Eq. (9)

In S-MTT, we need to solve Eq. (9) when $q=1$ and $p \in \{1, 2, \infty\}$. Clearly, the overall objective is non-smooth due to the non-smoothness of the $\ell_{p,1}$ mixed norm. If straightforward first-order subgradient methods were used to solve the S-MTT problem, only sublinear convergence (i.e. convergence to an ϵ -accurate solution in $O(\frac{1}{\epsilon^2})$ iterations) is guaranteed. To obtain a better convergence rate, we exploit recent developments in non-smooth convex optimization. The unpublished manuscript by Nesterov (2007) considers the problem of minimizing a convex objective composed of a smooth convex term and a “simple” non-smooth convex term. Here, “simple” means that the proximal mapping³ of the non-smooth term can be computed efficiently. In this case, an APG method can be devised to solve the non-smooth convex program with guaranteed quadratic convergence. Because of its attractive convergence property, this APG method has been extensively used to efficiently solve smooth convex optimization problems with non-smooth norm regularizers (e.g. MTL problems, Chen et al. 2009). It should be noted that Beck and Teboulle (2009) independently proposed the “ISTA” algorithm for solving linear inverse problem with the same convergence rate. This work was further extended to convex-concave optimization in Tseng (2008).

In general, APG iterates between updating the current representation matrix $\mathbf{C}^{(k)}$ and an aggregation matrix $\mathbf{V}^{(k)}$. Each APG iteration consists of two steps: (1) a generalized gradient mapping step that updates $\mathbf{C}^{(k)}$ keeping $\mathbf{V}^{(k)}$ fixed, and (2) an aggregation step that updates $\mathbf{V}^{(k)}$ by linearly combining $\mathbf{C}^{(k+1)}$ and $\mathbf{C}^{(k)}$. To initialize $\mathbf{V}^{(0)}$ and $\mathbf{C}^{(0)}$, we set them to $\mathbf{0}$.

4.3.1 Gradient Mapping Step

Given the current estimate $\mathbf{V}^{(k)}$, we obtain $\mathbf{C}^{(k+1)}$ by solving Eq. (12), which is nothing but the proximal mapping of $\tilde{\lambda} \|\mathbf{Y}\|_{p,1}$. Here, η is a small step parameter, and $\tilde{\lambda} = \eta\lambda_2$.

$$\mathbf{C}^{(k+1)} = \arg \min_{\mathbf{Y}} \frac{1}{2} \|\mathbf{Y} - \mathbf{H}\|_2^2 + \tilde{\lambda} \|\mathbf{Y}\|_{p,q}, \quad (12)$$

The temporal parameter \mathbf{H} is an η step from the current estimate $\mathbf{V}^{(k)}$ along the negative gradient of the smooth term in Eq. (9) and is calculated in Eq. (13).

³ The proximal mapping of a non-smooth convex function $h(\cdot)$ is defined as: $\mathbf{prox}_h(\mathbf{x}) = \arg \min_{\mathbf{u}} (h(\mathbf{u}) + \frac{1}{2} \|\mathbf{u} - \mathbf{x}\|_2^2)$.

$$\begin{aligned} \mathbf{H} &= \mathbf{V}^{(k)} - \eta \nabla_s^{(k)} \\ &= \mathbf{V}^{(k)} - \eta \left[\mathbf{B}^T \mathbf{B} \mathbf{V}^{(k)} + \lambda_1 \mathbf{V}^{(k)} \hat{\mathbf{L}} - \mathbf{B}^T \mathbf{X} \right]. \end{aligned} \tag{13}$$

Taking joint sparsity into consideration and since $q = 1$, Eq. (12) decouples into $(m + d)$ disjoint subproblems (one for each row vector \mathbf{C}_i), as shown in Eq. (14), where \mathbf{Y}_i and \mathbf{H}_i denote the i th row of the matrix \mathbf{Y} and \mathbf{H} , respectively.

$$\mathbf{C}_i^{(k+1)} = \arg \min_{\mathbf{Y}_i} \frac{1}{2} \|\mathbf{Y}_i - \mathbf{H}_i\|_2^2 + \tilde{\lambda} \|\mathbf{Y}_i\|_p, \tag{14}$$

Each subproblem is the proximal mapping of the ℓ_p vector norm, which is a variant of the vector projection problem onto the ℓ_p ball. The solution to each subproblem and its time complexity depends on p . In Sect. 4.3.3, we provide the solution of this subproblem for popular ℓ_p norms, namely $p \in \{1, 2, \infty\}$.

4.3.2 Aggregation Step

In this step, we construct a linear combination of $\mathbf{C}^{(k)}$ and $\mathbf{C}^{(k+1)}$ to update $\mathbf{V}^{(k+1)}$ as follows:

$$\mathbf{V}^{(k+1)} = \mathbf{C}^{(k+1)} + \frac{\alpha_{k+1} (1 - \alpha_k)}{\alpha_k} (\mathbf{C}^{(k+1)} - \mathbf{C}^{(k)}), \tag{15}$$

where α_k is conventionally set to $\frac{2}{k+3}$. Our overall APG algorithm is summarized in Algorithm 2. Note that convergence is achieved when the relative change in solution or objective function falls below a predefined tolerance.

Algorithm 2: Structured Multi-Task Jointly Sparse Representation

Input : $\mathbf{X}, \mathbf{B}, p, q = 1, \lambda_1, \tilde{\lambda}$, and η

Output: \mathbf{C}

- 1 Initialize $k \leftarrow 0, \mathbf{V}^{(k)} = \mathbf{0}, \mathbf{C} = \mathbf{0}, \alpha_k = 1$
 - 2 **while** not converged **do**
 - 3 Compute
 $\mathbf{H} = \mathbf{V}^{(k)} - \eta \left[\mathbf{B}^T \mathbf{B} \mathbf{V}^{(k)} + \lambda_1 \mathbf{V}^{(k)} \hat{\mathbf{L}} - \mathbf{B}^T \mathbf{X} \right]$
 - 4 Solve Eq (14) $\forall i = 1, \dots, (m + d)$ to obtain $\mathbf{C}^{(k+1)}$
 - 5 $\alpha_{k+1} = \frac{2}{k+3}$
 - 6 $\mathbf{V}^{(k+1)} = \mathbf{C}^{(k+1)} + \frac{\alpha_{k+1}(1-\alpha_k)}{\alpha_k} (\mathbf{C}^{(k+1)} - \mathbf{C}^{(k)})$
 - 7 $k \leftarrow k + 1$
 - 8 **end**
-

4.3.3 Solving Eq. (14) for $p \in \{1, 2, \infty\}$

The solution to Eq. (14) depends on the value of p . For $p \in \{1, 2, \infty\}$, we show that this solution has a closed form. Note that these solutions can be extended to ℓ_p norms beyond the three that we consider here.

For $p=1$: The solution of Eq. (14) is equivalent to the L_1 tracker solution, as shown in Eq. (11). Here, the update $\mathbf{C}_i^{(k+1)}$ is computed in closed form in Eq. (16), where $\mathcal{S}_{\tilde{\lambda}}$ is the soft-thresholding operator defined in scalar form as

$\mathcal{S}_{\tilde{\lambda}}(a) = \text{sign}(a) \max(0, |a| - \tilde{\lambda})$. This operator is applied elementwise on \mathbf{H}_i .

$$\begin{aligned} \mathbf{C}_i^{(k+1)} &= \arg \min_{\mathbf{Y}_i} \frac{1}{2} \|\mathbf{Y}_i - \mathbf{H}_i\|_2^2 + \tilde{\lambda} \|\mathbf{Y}_i\|_1 \\ &= \mathcal{S}_{\tilde{\lambda}}(\mathbf{H}_i) \end{aligned} \tag{16}$$

For $p=2$: Following Chen et al. (2009), the update $\mathbf{C}_i^{(k+1)}$ is computed in closed form in Eq. (17).

$$\begin{aligned} \mathbf{C}_i^{(k+1)} &= \arg \min_{\mathbf{Y}_i} \frac{1}{2} \|\mathbf{Y}_i - \mathbf{H}_i\|_2^2 + \tilde{\lambda} \|\mathbf{Y}_i\|_2 \\ &= \max \left(0, 1 - \frac{\tilde{\lambda}}{\|\mathbf{H}_i\|_2} \right) \mathbf{H}_i \end{aligned} \tag{17}$$

For $p=\infty$ The update $\mathbf{C}_i^{(k+1)}$ is computed via a projection onto the ℓ_∞ ball that can be done by a simple sorting procedure (Chen et al. 2009). In this case, the solution is given in Eq. (18).

$$\mathbf{C}_i^{(k+1)} = \max \left(0, 1 - \frac{\tilde{\lambda}}{\|\mathbf{H}_i\|_1} \right) \mathbf{a}, \tag{18}$$

where the j th element of vector \mathbf{a} is defined as

$$\begin{aligned} a_j &= \text{sign}(\mathbf{H}_{ij}) \min \left(|\mathbf{H}_{ij}|, \frac{1}{j} \left(\sum_{r=1}^j u_r - \tilde{\lambda} \right) \right) \\ \forall j &= 1, \dots, n. \end{aligned}$$

The temporary parameters u_r and \hat{j} are obtained as follows. We set $u_j = |\mathbf{C}_{ij}| \forall j$ and sort these values in decreasing order: $u_1 \geq u_2 \geq \dots \geq u_n$. Then, we set $\hat{j} = \max\{j : \sum_{r=1}^j (u_r - u_j) < \tilde{\lambda}\}$. Here, \mathbf{C}_{ij} and \mathbf{H}_{ij} denote the element in the i th row and j th column of matrices \mathbf{C} and \mathbf{H} , respectively.

4.3.4 Computational Complexity of Algorithm 2

The computational complexity of each iteration in Algorithm 2 is dominated by the gradient computation in Step 3 and the update of $\mathbf{C}_i^{(k+1)}$ in Step 4. Exploiting the structure of \mathbf{B} , the complexity of Step 3 is $O(mnd)$, while that of Step 4 depends on p . The latter complexity is $O(n(m+d))$ for $p \in \{1, 2\}$ and $O(n(m+d)(1 + \log n))$ for $p = \infty$. Since $d \gg m$, the per-frame complexity of the proposed S-MTT and MTT trackers is $O(mnd\epsilon^{-\frac{1}{2}})$, where the number of iterations is $O(\epsilon^{-\frac{1}{2}})$ for an ϵ -accurate solution. In comparison, the time complexity of the L_1 tracker (that is equivalent to our L_{11} tracker) is at least $O(nd^2)$. In our experiments, we observe that the L_{11} tracker (that uses APG) is two orders of magnitude faster than the L_1 tracker (that solves n Lasso problems independently) in general. For example, when $m = 11, n = 200$, and $d = 32 \times 32$, the average per-frame run-time for L_{11} and L_1 are 1.1 and

179 s respectively. This is on par with the accelerated “real-time” implementation of the L_1 tracker in Li et al. (2011).

4.4 Dictionary Update

Target appearance remains the same only for a certain period of time, but eventually the object templates in \mathbf{D}_t are no longer an accurate representation of the target’s appearance. A fixed appearance template is prone to the tracking drift problem, since it is incapable of handling appearance changes over time. In this paper, our dictionary update scheme is adopted from the work in Mei and Ling (2011). Each target template in \mathbf{D}_t is assigned a weight that is indicative of how representative the template is. In fact, the more a template is used to represent tracking results, the higher its weight is. When \mathbf{D}_t cannot represent some particles well (up to a predefined threshold), the target template with the smallest weight is replaced by the current tracking result. To initialize the m target templates, we sample equal-sized patches at and around the initial position of the target. In our experiments, we shift the initial bounding box by 1–3 pixels in each direction, thus, resulting in $m = 11$ object templates as in Mei and Ling (2011). All dictionary templates are normalized.

5 Experimental Results

In this section, we present extensive experimental results that validate the effectiveness and efficiency of our proposed S-MTT and MTT methods. We also conduct a thorough comparison between our proposed trackers and state-of-the-art tracking methods where applicable.

The experimental results are organized as follows. In Sect. 5.1, we give an overview of the video dataset that we test our S-MTT/MTT trackers. Section 5.2 enumerates the six state-of-the-art trackers that we compare against. The implementation details of our proposed trackers are highlighted in Sect. 5.3. In Sect. 5.4, we report the average runtime of the S-MTT trackers with varying parameter settings, as well as, compare it to the runtime of the L_1 tracker. Qualitative and quantitative comparisons between S-MTT/MTT and the state-of-the-art trackers are made in Sects. 5.5 and 5.6 respectively. The comparative results demonstrate that our method provides more robust and accurate tracking results than the state-of-the-art. Several videos for the tracking results can be found in the supplementary material. The videos and codes will be made available on our project website.⁴

⁴ <https://sites.google.com/site/videoadsc/>.

5.1 Datasets

To evaluate our proposed trackers, we compile a set of 15 challenging tracking sequences (denoted as *car4*, *car11*, *david indoor*, *sylv*, *trellis70*, *girl*, *coke11*, *faceocc2*, *shaking*, *football*, *singer1*, *singer1(low frame rate)*, *skating1*, *skating1(low frame rate)*, *soccer*). The video sequences *car4*, *car11*, *david indoor*, *sylv* and *trellis70* can be downloaded from an online source.⁵ The video sequences *girl*, *coke11* and *faceocc2* can be downloaded from an online source.⁶ The other video sequences *shaking*, *football*, *singer1*, *singer1(low frame rate)*, *skating1*, *skating1(low frame rate)* and *soccer* can be downloaded from an online source.⁷ These videos are recorded in indoor and outdoor environments and include most challenging factors in visual tracking: complex background, moving camera, fast movement, large variation in pose and scale, occlusion, as well as shape deformation and distortion (see Figs. 3, 4).

5.2 Baselines

We compared the proposed algorithms (MTT and S-MTT) with six state-of-the-art visual trackers: VTD tracker (Kwon and Lee 2010), L_1 tracker (Mei and Ling 2011), IVT tracker (Ross et al. 2008), MIL tracker (Babenko et al. 2009), Fragments-based tracker (Frag) (Adam et al. 2006), and OAB tracker (Grabner et al. 2006). We use the publicly available source codes or binaries provided by the authors themselves with the same initialization and parameter settings to generate the comparative results. In our experiments, our proposed tracking methods use the same parameters for all the test sequences.

5.3 Implementation Details

All our experiments are done using MATLAB R2008b on a 2.66 GHZ Intel Core2 Duo PC with 6 GB RAM. The template size d is set to half the size of the target initialization in the first frame. Usually, d is in the order of several hundreds of pixels. For all experiments, we model $p(\mathbf{s}_t | \mathbf{s}_{t-1}) \sim \mathcal{N}(\mathbf{0}, \text{diag}(\boldsymbol{\sigma}))$, where $\boldsymbol{\sigma} = [0.005, 0.0005, 0.0005, 0.005, 4, 4]^T$. We set the number of particles $n = 400$, the total number of target templates $m = 11$ [The templates are obtained with 1–3 pixels shift around the target position as the same as the L_1 tracker (Mei and Ling 2011)], and the number of occlusion templates to d . In Algorithm 2, we set $\eta = 0.01$, $\lambda_1 = 1$, $\tilde{\lambda}$ (by cross-validation) to $\{0.01, 0.005, 0.2\}$ for L_{21} , L_{11} and $L_{\infty 1}$ respectively, and $\tilde{\lambda}$ to $\{0.005, 0.001, 0.2\}$ for L_{21}^* , L_{11}^* and

⁵ <http://www.cs.toronto.edu/~dross/ivt/>.

⁶ http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml.

⁷ <http://cv.snu.ac.kr/research/~vtd/>.

Table 1 Average per-frame runtime (in seconds) of 4 trackers (L_{21}^* , $L_{\infty 1}^*$, L_{11}^* , and L_1) with varying template sizes d and number of particles n

n	d															
	16×16				32×32				48×48				64×64			
	L_1	L_{21}^*	$L_{\infty 1}^*$	L_{11}^*	L_1	L_{21}^*	$L_{\infty 1}^*$	L_{11}^*	L_1	L_{21}^*	$L_{\infty 1}^*$	L_{11}^*	L_1	L_{21}^*	$L_{\infty 1}^*$	L_{11}^*
100	2.07	0.32	0.15	0.11	84.5	0.41	0.78	0.72	601.3	2.62	2.01	2.36	2230.7	8.88	5.97	9.50
200	3.91	0.33	0.53	0.21	178.9	1.46	1.98	1.55	1238.3	5.56	3.22	3.85	4536.0	18.18	9.37	13.99
300	5.79	0.73	1.10	0.44	256.4	2.71	2.90	2.23	1969.9	5.04	5.95	6.98	6717.2	23.96	37.87	28.99
400	7.90	1.21	1.47	0.55	340.5	4.69	5.39	2.74	2665.2	11.45	15.32	4.99	8881.8	23.26	39.10	31.28
500	11.39	1.61	1.85	0.62	422.5	4.85	7.26	3.19	3158.6	19.38	20.30	8.83	1100.3	32.17	46.53	37.07
600	12.05	2.24	2.71	0.76	497.1	5.97	8.34	4.57	3915.9	21.04	25.32	8.57	1314.7	40.08	53.79	41.71
700	13.58	2.96	3.62	0.99	572.3	6.17	7.45	5.03	4429.9	22.36	35.19	10.96	1533.2	48.86	63.51	49.19
800	15.89	3.58	4.35	1.08	659.4	7.62	10.37	5.77	5080.4	23.82	27.61	13.16	1889.9	55.99	72.11	54.88

$L_{\infty 1}^*$ respectively. Each tracker uses the same parameters for all video sequences. In all cases, the initial position of the target is selected manually.

5.4 Computational Cost

The popular L_1 tracker that uses a similar sparse representation model for particle appearance has shown to achieve better tracking performance than state-of-the-art trackers (Mei and Ling 2011). However, its computational cost grows proportionally as the number of particle samples n and template size d increase. Due to the inherent similarity between the L_1 tracker and the proposed trackers (MTT and S-MTT), we compare their average runtimes in Table 1. S-MTT and MTT has very similar computational costs, so for simplicity, we just report the runtime results of S-MTT (L_{21}^* , $L_{\infty 1}^*$, L_{11}^*) and L_1 in Table 1. Based on the results, it is clear that our trackers are much more efficient than the L_1 tracker. For example, when $m = 11$, $n = 400$, and $d = 32 \times 32$, the average per-frame run-time for L_{21}^* , $L_{\infty 1}^*$, L_{11}^* , and L_1 trackers are 4.69, 5.39, 2.74, and 340.5 s, respectively. Interestingly, our L_{11}^* tracker, which is similar to L_1 tracking but solved using the APG method, is about 120 times faster than the L_1 tracker. As we know, increasing n and d will improve tracking performance. For L_1 tracking, the runtime cost increases dramatically with both n and d ; however, this increase is much more reasonable with our trackers. Note that the computational complexity of S-MTT is derived in Sect. 4.3.4.

5.5 Qualitative Comparison

The *car4* sequence is captured in an open road scenario. Tracking results at frames {20, 186, 235, 305, 466, 641} for all 12 methods are shown in Fig. 3a. The different tracking methods are color-coded. OAB, Frag, and VTD start to drift from the target at frame 186, while MIL starts to show

some target drifting at frame 200 and finally loses the target at frame 300. IVT and L_1 track the target quite well. The target is successfully tracked throughout the entire sequence by our L_{11} , L_{21} , $L_{\infty 1}$, L_{11}^* , L_{21}^* , and $L_{\infty 1}^*$ methods.

In the *car11* sequence, a car is driven into a very dark environment, while being videotaped from another moving car. Tracking results for frames {10, 110, 200, 250, 309, 393} are presented in Fig. 3b. Frag starts to drift around frame 60. Due to changes in lighting, MIL starts to undergo target drift from frame 120. OAB and L_1 methods start to fail in frame 284. IVT and VTD can track the target through the whole video sequence; however, these tracks are not as robust or accurate as the proposed L_{21} , $L_{\infty 1}$, L_{21}^* and $L_{\infty 1}^*$ trackers.

The *coke11* sequence contains frequent occlusions and fast motion, which cause motion blur. The S-MTT, MTT, L_1 , OAB, and MIL trackers can track the target accurately almost throughout the entire sequence. The other trackers fails due to pose change and occlusion as shown in Fig. 3c.

In the *david* sequence, a moving face is tracked. The tracking results at frames {354, 423, 465, 502, 588, 760} are shown in Fig. 3d. Frag and VTD fail around frames 423 and 465 respectively. OAB starts to drift at frame 550. MIL and L_1 adequately track the face, but experience target drift, especially at frames 690 and 500, respectively. The IVT, S-MTT and MTT methods track the moving face accurately.

In the *faceocc* sequence, a moving face is tracked, which can evaluate the robustness to occlusions of different methods. The tracking results at frames {100, 231, 314, 474, 571, 865} are shown in Fig. 3e. Because there is only occlusion by a book and no changes in illumination and motion, most of the methods can track the face accurately except OAB and MIL, which encounter minor drift.

Results on the *faceocc2* sequence are shown in Fig. 3f. Most trackers start drifting from the man's face when it is almost fully occluded by the book. Because the L_1 , MTT and S-MTT methods explicitly handle partial occlusions, and update the object dictionary progressively, they handle the

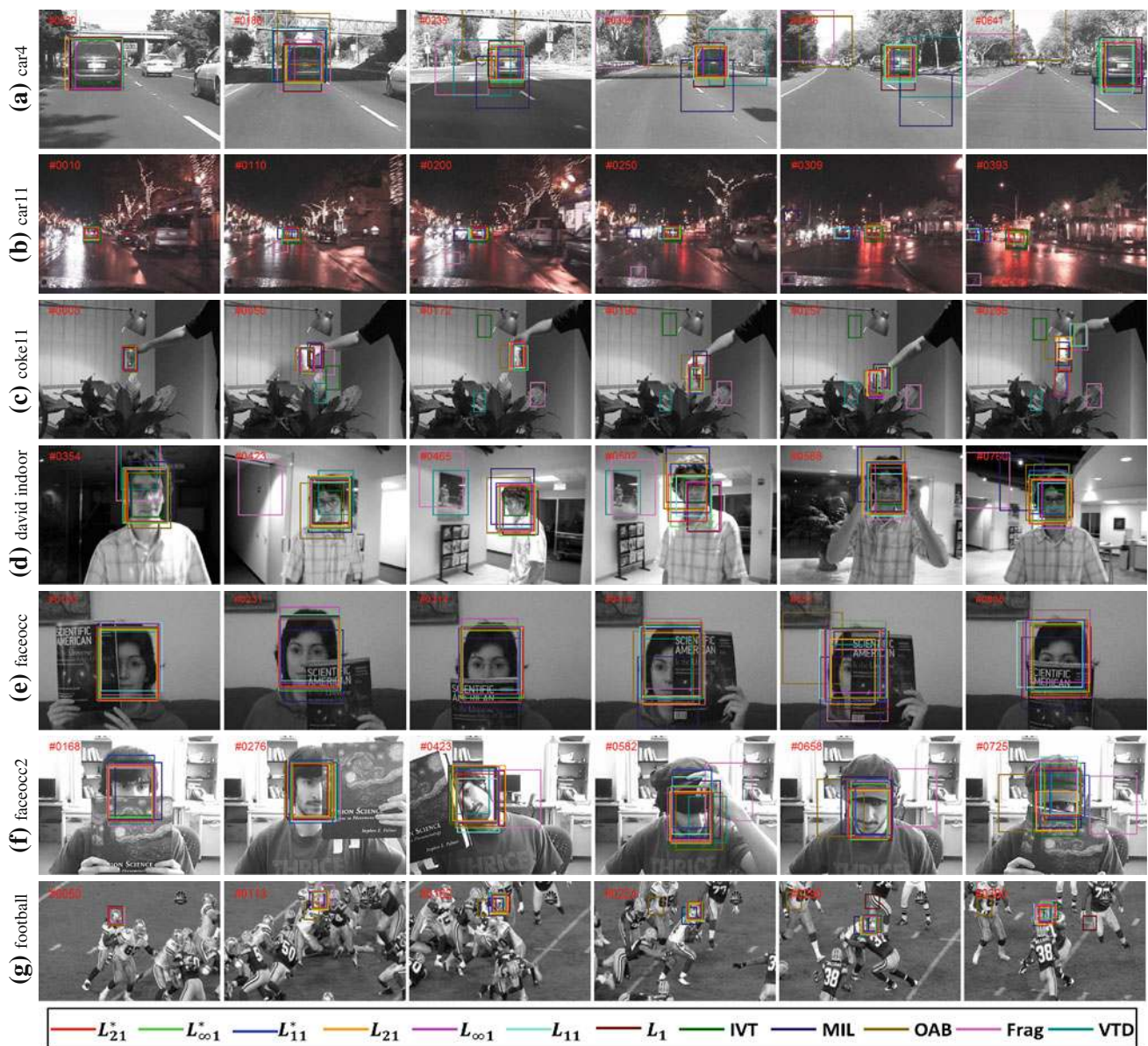


Fig. 3 (Color online) Tracking results of 12 trackers on 7 video sequences denoted with *different colors*. Frame numbers are overlaid in *red*. See text for details

appearance changes in this sequence very well and continue tracking the target during and after the occlusion.

The *football* sequence includes severe background clutter, which is similar in appearance to the tracked target. For the other methods, tracking drifts from the intended object (helmet) to other similar looking objects in the vicinity. This is especially the case when the two football players collide at frame 362 (refer to Fig. 3g). The proposed trackers (such as L_{21} , $L_{\infty 1}$ and L_{21}^*) overcome this problem and successfully track the target because they exploit structural relationships between particle representations.

Figure 4a shows tracking results for the *girl* sequence. Performance on this sequence exemplifies the robustness of

MTT 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). S-MTT, MTT and L_1 are capable of tracking the target during the entire sequence. Other trackers experience drift at different instances: Frag at frame 248, OAB and IVT at frame 436, and VTD at frame 477.

In the *onelsr* sequence, the background color is similar to the color of the woman's trousers, and the man's shirt and pants have a similar color to the woman's coat. In addition, the woman undergoes partial occlusion when the man in the scene walks behind her. Some results are shown in Fig. 4b. While tracking the woman, IVT, MIL, Frag, OAB, and VTD start tracking the man when the woman is partially occluded

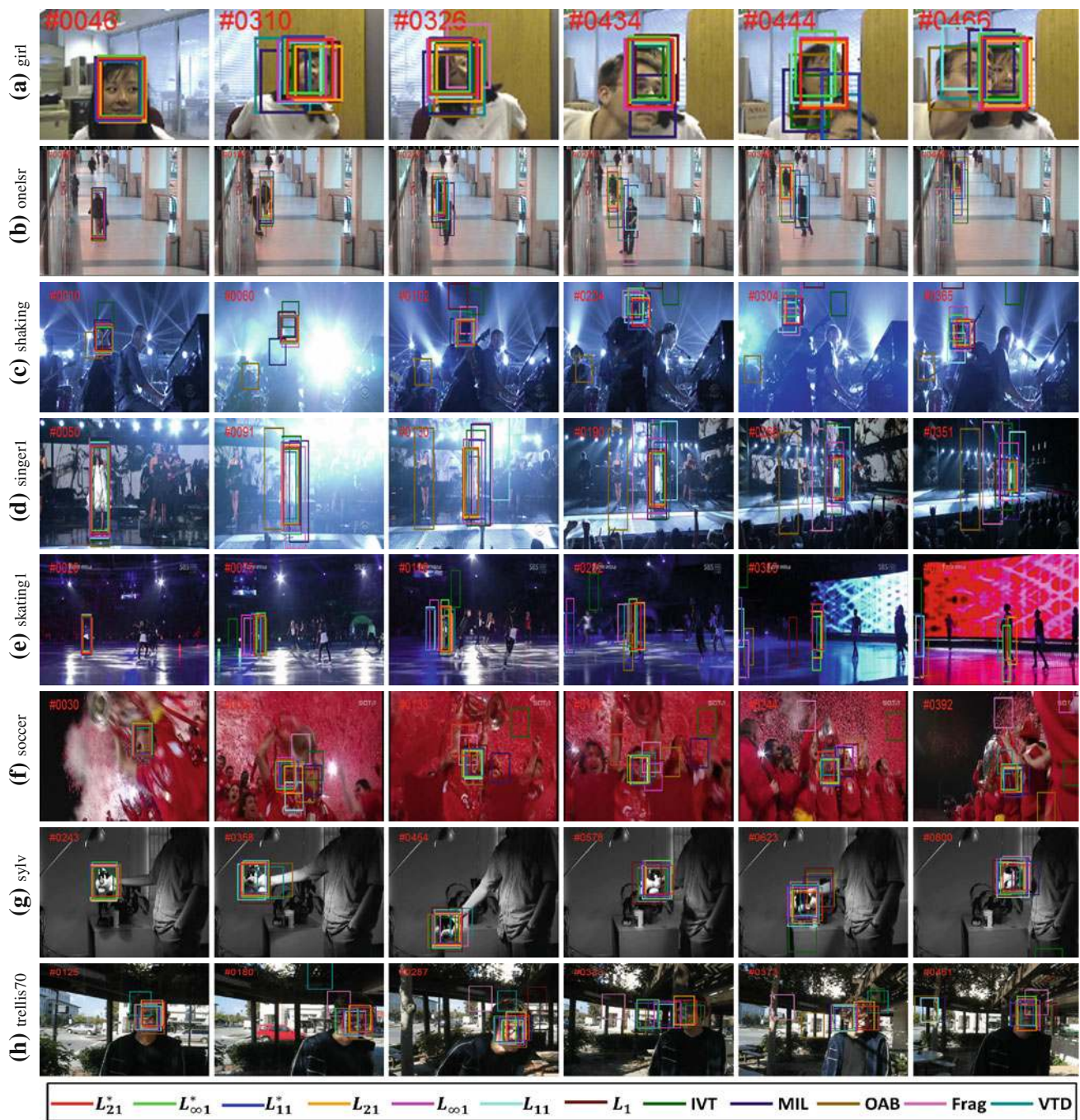


Fig. 4 (Color online) Tracking results of 12 trackers on 8 video sequences delineated by *different colors*. Frame numbers are overlaid in *red*. See text for details

around frame 200, and are unable to recover from this failure after that. The L_1 tracker tracks the woman quite well. Compared with other trackers, our L_{21}^* , $L_{\infty 1}^*$, L_{21} and $L_{\infty 1}$ trackers are more robust to the occlusion. In addition, our L_{21}^* , $L_{\infty 1}^*$, L_{21} and $L_{\infty 1}$ trackers are much better than L_1 , L_{11} , and L_{11}^* , which demonstrate that imposing joint sparsity between particle representations is helpful for robust object tracking.

In the *shaking* sequence, the tracked object is subject to changes in illumination and pose. While the stage lighting condition is drastically changed, and the pose of the object is severely varied due to head shaking, our method successfully tracks the object (refer to Fig. 4c). Compared with L_{11} , L_{11}^* and L_1 , L_{21} , $L_{\infty 1}$, L_{21}^* and $L_{\infty 1}^*$ perform better because their joint sparse particle representation is more robust to rapid changes. Other methods (OAB, IVT, L_1 , and Frag) fail

to track the object when these changes occur. VTD and MIL methods can track the object quite well except for some errors around frame 60.

The *singer1(l)* sequence contains abrupt object motion with significant illumination and scale changes, which cause most of the trackers to drift as shown in Fig. 4d. S-MTT, MTT and VTD handle these changes well. Compared with MTT (L_{21} , $L_{\infty 1}$ and L_{11}), S-MTT (L_{21}^* , $L_{\infty 1}^*$ and L_{11}^*) obtains much better performance, which shows that harnessing local structure between particle representations is useful for object tracking.

In the *skatingl* sequence, there are abrupt object motion, severe illumination and scale changes, viewpoint changes and occlusions, which lead most of the trackers to fail. Our proposed trackers (MTT and S-MTT) and VTD handle these changes well as shown in Fig. 4e. Note that, in the 353th frame in the Fig. 4e, our proposed trackers are slightly better than the VTD method, which is the most recent state-of-the-art tracking method that can cope with abrupt motion and appearance changes.

Results on the *soccer* sequence are shown in Fig. 4f. They demonstrate how our proposed method outperforms most of the state-of-the-art trackers when the target is severely occluded by other objects. The L_{21}^* , $L_{\infty 1}^*$, L_{11}^* , L_{21} and L_{11} methods accurately track the player's face despite scale and pose changes as well as occlusion/noise from the confetti raining around him. Other methods (IVT, L_1 , $L_{\infty 1}$, OAB, MIL, and Frag) fail to track the object reliably. The VTD tracker can track the target in this sequence quite well.

Results on the *sylv* sequence are shown in Fig. 4g. In this sequence, a stuffed animal is being moved around, thus, leading to challenging pose, lighting, and scale changes. IVT fails around frame 623 as a result of a combination of pose and illumination change. The rest of the trackers are able to track the target throughout the sequence, although the Frag, MIL, VTD, OAB, L_{11} and L_1 encounter minor drift from the target.

The *trellis70* sequence is captured in an outdoor environment where lighting conditions change drastically. The video is acquired when a person walks underneath a trellis covered in vines. As shown in Fig. 4h, the cast shadow changes the appearance of the target face significantly. Furthermore, the combined effects of pose and lighting variations along with a low frame rate make visual tracking extremely difficult. Nevertheless, the L_{21} , $L_{\infty 1}$, L_{21}^* and $L_{\infty 1}^*$ trackers can follow the target accurately and robustly, while the other tracking methods perform below par in this case. VTD and Frag fail around frame 185. L_1 starts drifting at frame 287, while MIL and OAB fail at frame 323. IVT starts drifting at frame 330.

5.6 Quantitative Comparison

To give a quantitative comparison between the 12 trackers, we obtain ground truth for all 15 sequences. Here, we note

that ground truth for some of the video sequences is readily available. We manually label the other sequences. Tracker 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 Babenko et al. (2009) and Mei and Ling (2011). Clearly, this distance⁸ should be small.

In Fig. 5, we plot the distance of each tracker over time on 2 sequences for simplicity. From this figure, we see that MTT and S-MTT trackers consistently produce a smaller distance than other trackers in general. This implies that MTT and S-MTT can accurately track the target despite severe occlusions, pose variations, illumination changes, and abrupt motions.

In Table 2, we show the average center distance for each tracker over the 15 sequences. It is clear that the S-MTT and MTT methods are consistently better than the other trackers in most sequences. Among the MTT methods, L_{21} outperforms L_{11} and $L_{\infty 1}$ in general. For S-MTT method, L_{21}^* are much better than L_{11}^* and $L_{\infty 1}^*$. In fact, except for the *faceocc*, *football*, *shaking* and *skatingl* sequences, in which we obtain similar results as IVT and VTD, the L_{21} and L_{21}^* trackers do outperform the other methods. Frag and L_1 perform well under partial occlusion but tend to fail under severe illumination and pose changes. The IVT tracker is hardly affected by changes in appearance except those due to illumination. OAB is effected by background clutter and easily drifts from the target. MIL performs well except when severe illumination changes force the tracker to drift into the background. VTD tends to be robust against illumination change, but it cannot handle severe occlusions and viewpoint changes adequately.

Now, we compare the performance of S-MTT and MTT methods. Based on the results in Table 2, L_{21} and $L_{\infty 1}$ outperform L_{11} . This is due to the fact that the L_{11} tracker learns particle representations separately, while L_{21} and $L_{\infty 1}$ capitalize on dependencies between different particles to obtain more robust jointly sparse representations. These results demonstrate that it is useful for visual tracking to impose joint sparsity among particle representations. For S-MTT, the performance of the corresponding three trackers (L_{21}^* , $L_{\infty 1}^*$ and L_{11}^*) has similar tendencies as MTT. However, S-MTT is reasonably better than MTT. This validates the impact of using local graph structure to regularize particle representations and yield more robust object tracking.

In addition, we compare MTT and S-MTT with the L_1 tracker, which is the most related tracker to ours and has shown state-of-the-art performance (Mei and Ling 2011). Based on the results in Table 2, MTT (L_{21} and $L_{\infty 1}$) and S-MTT (L_{21}^* and $L_{\infty 1}^*$) outperform the L_1 tracker. That is because L_1 tracking represents particles separately, while the

⁸ This dissimilarity measure is used often to compare tracking performance. Other measures can be used, including the PASCAL overlap score.

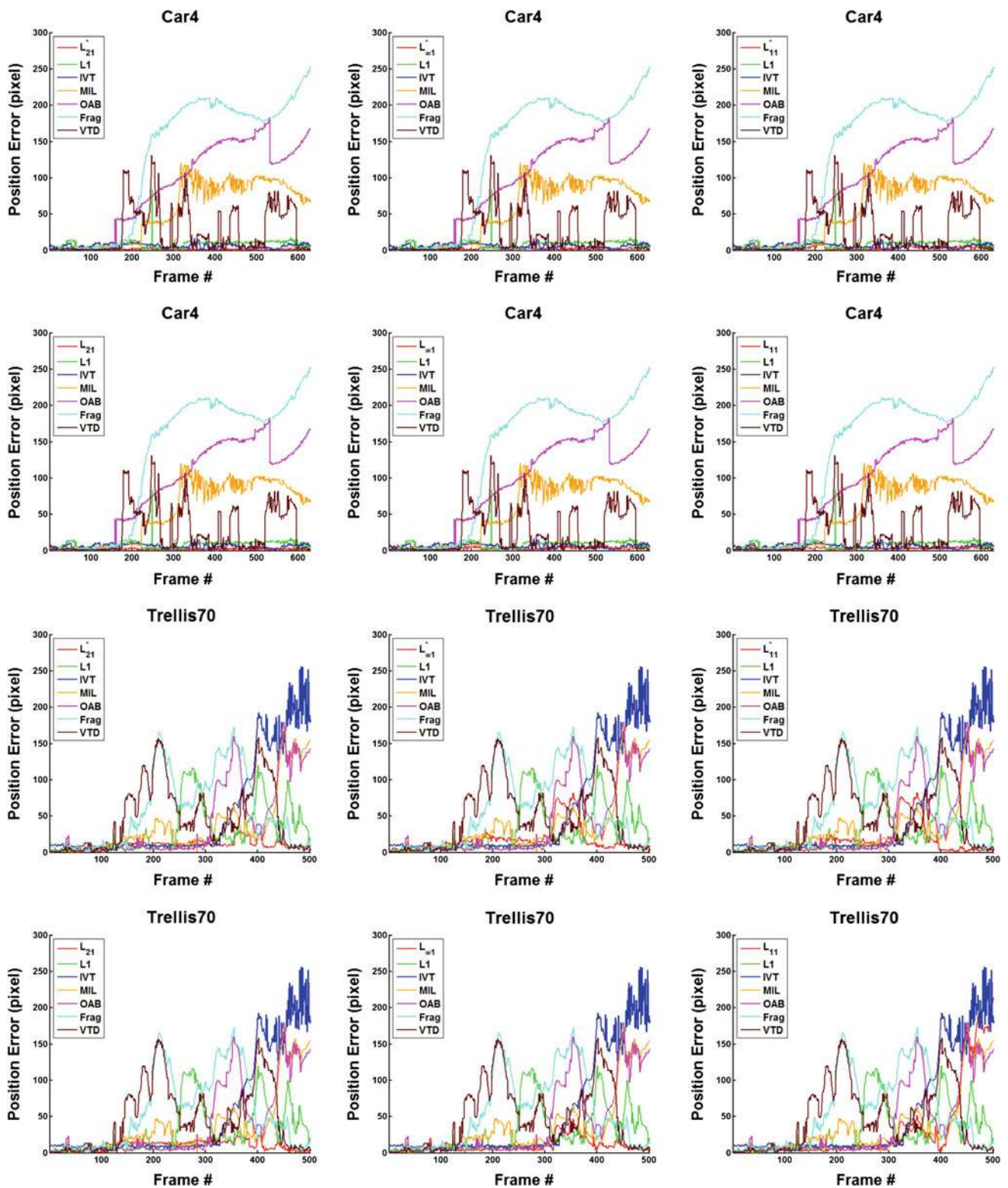


Fig. 5 (Color online) Center distance (in pixels) between tracking result and ground truth over time for 12 trackers applied to 15 video sequences

proposed trackers capitalize on dependencies between different particle representations to obtain a more robust jointly sparse representation. Our results demonstrate that it is useful

for visual tracking to mine particle relationships. L_{11}^* outperforms the L_{11} and L_1 trackers, since the L_{11}^* tracker makes use of the local graph structure. Moreover, in theory, the L_1

Table 2 (Color online) The mean distances of 12 different trackers on 15 different video sequences

VID	L_{21}^*	$L_{\infty 1}^*$	L_{11}^*	L_{21}	$L_{\infty 1}$	L_{11}	IVT	L_1	MIL	OAB	Frag	VTD
<i>car4</i>	1.79	2.40	3.60	2.25	3.83	3.33	8.50	6.37	53.76	88.12	127.29	27.01
<i>car11</i>	2.01	2.02	16.73	1.93	2.01	23.69	19.18	5.42	53.75	5.69	72.71	3.74
<i>coke11</i>	7.34	9.46	7.38	3.15	7.39	6.53	12.09	58.53	13.70	11.33	70.98	62.70
<i>david</i>	7.55	8.38	6.67	7.88	8.37	9.17	9.87	7.16	25.95	21.15	66.91	58.99
<i>faceocc</i>	12.52	18.59	10.92	7.75	23.45	19.15	7.00	9.15	34.35	17.23	7.90	8.73
<i>faceocc2</i>	6.45	7.04	10.07	8.09	8.06	10.11	15.25	6.55	10.23	20.84	48.22	11.85
<i>football</i>	4.07	5.19	4.51	4.71	4.87	5.25	15.44	5.25	7.98	53.31	6.35	3.33
<i>girl</i>	3.88	4.61	5.43	4.49	4.33	9.85	4.98	4.16	12.38	10.99	7.43	11.44
<i>onelsr</i>	3.39	2.28	20.26	3.26	3.63	23.61	4.69	24.03	23.82	12.46	57.57	44.29
<i>shaking</i>	9.35	8.62	10.80	8.36	9.54	11.37	37.78	52.23	7.91	100.27	15.27	3.96
<i>singer1</i>	1.43	1.14	4.53	1.82	4.74	18.88	5.25	9.79	11.09	62.99	26.92	1.45
<i>skating1</i>	5.61	9.18	75.51	7.41	82.48	77.89	20.08	74.94	49.18	39.25	63.26	5.02
<i>soccer</i>	10.45	11.06	17.16	14.26	46.99	16.06	58.49	97.78	46.34	65.31	41.42	10.50
<i>sylv</i>	4.77	3.95	4.01	4.78	4.71	7.56	14.53	39.37	15.29	10.42	6.78	7.37
<i>trellis70</i>	10.30	17.76	15.36	10.32	9.96	33.14	31.08	54.02	37.32	41.46	55.69	47.76
average	6.06	7.44	14.20	6.03	14.96	18.37	17.61	30.32	26.87	37.39	44.98	20.54

On average, the proposed trackers (L_{21}^* , $L_{\infty 1}^*$, L_{11}^* , L_{21} , $L_{\infty 1}$, and L_{11}) outperform the other 6 state-of-the-art trackers. For each sequence, the smallest and second smallest distances are denoted in red and blue respectively

tracker is a special case of our MTT framework (refer to Eq. (11)), and it should produce the same results as L_{11} . However, this is not reflected in our empirical results due to three reasons. (a) The L_1 tracker is forced to adopt a smaller template size ($d = 12 \times 15$) due to its high computational cost $\mathcal{O}(nd^2)$. A larger d leads to a richer representation and improved tracking performance. As mentioned earlier, MTT methods set d to half the size of the initial bounding box, which is generally more than 600 pixels. (b) In the public MATLAB implementation of L_1 , the dictionary weights are used not only to update the target templates but also to multiply the templates themselves, which leads to an artificially sparser representation. For L_{11} , the weights are only used to update the target templates. In addition, MTT uses a more efficient solver (refer to Sect. 4.3.3) to learn particle representations, so L_{11} can reach a better solution than L_1 for the same stopping criterion at every frame. (c) Since the L_1 and L_{11} trackers both adopt the particle filter framework, their tracking results for the same sequence can be different because the particles that are randomly sampled at each frame tend to be different.

6 Conclusion

In this paper, we formulate particle filter based tracking as a structured multi-task sparse learning problem, where particle representations, regularized by a sparsity-inducing $\ell_{p,1}$ mixed norm and a local graph term, are learned jointly using an efficient APG method. We show that the popular L_1 tracker (Mei and Ling 2011) is a special case of our proposed formulation. Also, we extensively analyze the performance of our tracking paradigm on challenging real-world video sequences and show it outperforming six state-of-the-art tracking methods.

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

- Adam, A., Rivlin, E., & Shimshoni, I. (2006). Robust fragments-based tracking using the integral histogram. In *IEEE conference on computer vision and pattern recognition* (pp. 798–805).
- Avidan, S. (2005). Ensemble tracking. In *IEEE conference on computer vision and pattern recognition* (pp. 494–501).
- Babenko, B., Yang, M. H., & Belongie, S. (2009). Visual tracking with online multiple instance learning. In *IEEE conference on computer vision and pattern recognition* (pp. 983–990).
- Bao, C., Wu, Y., Ling, H., & Ji, H. (2012). Real time robust l1 tracker using accelerated proximal gradient approach. In *IEEE conference on computer vision and pattern recognition* (pp. 1–8).
- Beck, A., & Teboulle, M. (2009). A fast iterative shrinkagethresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Science*, 2(1), 183–202.
- Black, M. J., & Jepson, A. D. (1998). Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26(1), 63–84.
- Blasch, E., & Kahler, B. (2005). Multiresolution EO/IR target tracking and identification. In *International conference on information fusion* (Vol. 8, pp. 1–8).
- Candès, E. J., Romberg, J. K., & Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8), 1207–1223.
- Chen, X., Pan, W., Kwok, J., & Carbonell, J. (2009). Accelerated gradient method for multi-task sparse learning problem. In *IEEE international conference on data mining* (pp. 746–751).
- Collins, R. T., & Liu, Y. (2003). On-line selection of discriminative tracking features. In *International conference on computer vision* (pp. 346–352).
- Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5), 564–575.
- Doucet, A., De Freitas, N., & Gordon, N. (2001). *Sequential Monte Carlo methods in practice* (1st ed.). Springer.
- Grabner, H., Grabner, M., & Bischof, H. (2006). Real-time tracking via on-line boosting. In *British machine vision conference* (pp. 1–10).
- Grabner, H., Leistner, C., & Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. In *European conference on computer vision* (pp. 234–247).

- Isard, M., & Blake, A. (1998). Condensation—Conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1), 5–28.
- Jepson, A., Fleet, D., & El-Maraghi, T. (2003). Robust on-line appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10), 1296–1311.
- Jiang, N., Liu, W., & Wu, Y. (2011). Adaptive and discriminative metric differential tracking. In *IEEE conference on computer vision and pattern recognition* (pp. 1161–1168).
- Khan, Z., Balch, T., & Dellaert, F. (2004). A rao-blackwellized particle filter for eigentracking. In *IEEE conference on computer vision and pattern recognition* (pp. 980–986).
- Kwon, J., & Lee, K. M. (2010). Visual tracking decomposition. In *IEEE conference on computer vision and pattern recognition* (pp. 1269–1276).
- Leistner, C., Godec, M., Saffari, A., & Bischof, H. (2010). Online multi-view forests for tracking. In *DAGM* (pp. 493–502).
- Li, H., Shen, C., & Shi, Q. (2011). Real-time visual tracking with compressed sensing. In *IEEE conference on computer vision and pattern recognition* (pp. 1305–1312).
- Liu, B., Huang, J., Yang, L., & Kulikowski, C. (2011). Robust visual tracking with local sparse appearance model and k-selection. In *IEEE conference on computer vision and pattern recognition* (pp. 1–8).
- Liu, B., Yang, L., Huang, J., Meer, P., Gong, L., & Kulikowski, C. (2010). Robust and fast collaborative tracking with two stage sparse optimization. In *European conference on computer vision* (pp. 1–14).
- Liu, R., Cheng, J., & Lu, H. (2009). A robust boosting tracker with minimum error bound in a co-training framework. In *International conference on computer vision* (pp. 1459–1466).
- Mei, X., & Ling, H. (2011). Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11), 2259–2272.
- Mei, X., Ling, H., Wu, Y., Blasch, E., & Bai, L. (2011). Minimum error bounded efficient l1 tracker with occlusion detection. In *IEEE conference on computer vision and pattern recognition* (pp. 1257–1264).
- Nesterov, Y. (2007). Gradient methods for minimizing composite objective function. In CORE discussion paper.
- Peng, Y., Ganesh, A., Wright, J., Xu, W., & Ma, Y. (2012). RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34, 2233–2246.
- Quattoni, A., Carreras, X., Collins, M., & Darrell, T. (2009). An efficient projection for l1, infinity regularization. In *International conference on machine learning* (pp. 857–864).
- Ross, D., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1), 125–141.
- Tseng, P. (2008). On accelerated proximal gradient methods for convex–concave optimization. Technical report. <http://pages.cs.wisc.edu/~brecht/cs726docs/Tseng.APG.pdf>.
- Wu, Y., & Huang, T. S. (2004). Robust visual tracking by integrating multiple cues based on co-inference learning. *International Journal of Computer Vision*, 58, 55–71.
- Yang, C., Duraiswami, R., & Davis, L. (2005). Fast multiple object tracking via a hierarchical particle filter. In *International conference on computer vision* (pp. 212–219).
- Yang, M., Wu, Y., & Hua, G. (2009). Context-aware visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7), 1195–1209.
- Yilmaz, A., Javed, O., & Shah, M. (2006). Object tracking: A survey. *ACM Computing Surveys*, 38(4), 13.
- Yin, Z., & Collins, R. (2008). Object tracking and detection after occlusion via numerical hybrid local and global mode-seeking. In *IEEE conference on computer vision and pattern recognition* (pp. 1–8).
- Yu, Q., Dinh, T. B., & Medioni, G. (2008). Online tracking and reacquisition using co-trained generative and discriminative trackers. In *European conference on computer vision* (pp. 678–691).
- Yuan, X., & Yan, S. (2010). Visual classification with multi-task joint sparse representation. In *IEEE conference on computer vision and pattern recognition* (pp. 3493–3500).
- Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2012a). Low-rank sparse learning for robust visual tracking. In *European conference on computer vision* (pp. 1–8).
- Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2012b). Robust visual tracking via multi-task sparse learning. In *IEEE conference on computer vision and pattern recognition* (pp. 1–8).
- Zhou, S. K., Chellappa, R., & Moghaddam, B. (2004). Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, 11(1), 1491–1506.
- Zhu, X. (2008). Semi-supervised learning literature survey. Computer sciences technical report 1530, University of Madison.