Int J Comput Vis
DOI 10.1007/s11263-014-0738-0

Robust Visual Tracking Via Consistent Low-Rank Sparse Learning

Tianzhu Zhang · Si Liu · Narendra Ahuja · Ming-Hsuan Yang · Bernard Ghanem

Received: 15 November 2013 / Accepted: 3 June 2014

Abstract Object tracking is the process of determining the states of a target in consecutive video frames based on properties of motion and appearance consistency. In this paper, we propose a consistent low-rank sparse tracker (CLRST) that builds upon the particle filter framework for tracking. By exploiting temporal consistency, the proposed CLRST algorithm adaptively prunes and selects candidate particles. By using linear sparse combinations of dictionary templates, the proposed method learns the sparse representations of image regions corresponding to candidate particles jointly by exploiting the underlying low-rank constraints. In addition, the proposed CLRST algorithm is computationally attractive since temporal consistency property helps prune particles and the low-rank minimization problem for learning joint sparse representations can be efficiently solved by a sequence of closed form update operations. We evaluate the proposed CLRST algorithm against 14 state-of-the-art tracking methods on a set of 25 challenging image sequences. Experimental results show that the CLRST algorithm performs favorably against state-of-the-art tracking methods in terms of accuracy and execution time.

Keywords Visual tracking · Temporal consistency · Sparse representation · Low-rank representation

Communicated by M. Hebert.

T. Zhang
Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
e-mail: tzzhang10@gmail.com

T. Zhang · B. Ghanem
Advanced Digital Sciences Center (ADSC) of the University of Illinois, 1 Fusionopolis Way, #08-10 Connexis North Tower, Singapore 138632, Singapore

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

N. Ahuja
Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
e-mail: ahuja@vision.ai.uiuc.edu

M.-H. Yang
School of Engineering, University of California at Merced, Merced, CA 95344, USA
e-mail: mhyang@ucmerced.edu

B. Ghanem
Department of Electrical Engineering, King Abdullah University of Science and Technology (KAUST), Al Khawarizmi Building, Thuwal 23955-6900, Saudi Arabia
e-mail: bernard.ghanem@kaust.edu.sa

1 Introduction

Visual tracking is a well-known problem in computer vision with numerous applications including surveillance, robotics, human-computer interaction, and motion analysis, to name a few. Despite demonstrated success (Yilmaz et al. 2006; Salti et al. 2012), it remains challenging to design a robust visual tracking algorithm due to factors such as occlusion, background clutter, varying viewpoints, and illumination and scale changes.

Recently, numerous algorithms based on ℓ_1 minimization (Tsaig and Donoho 2006) have been proposed for visual tracking (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010; Bao et al. 2012; Li et al. 2011; Zhang et al. 2013c) where an image observation is sparsely represented by a dictionary of templates with online update. These methods have demonstrated that the use of sparse representation facilitates robustness to image corruptions caused by partial occlusions

Published online: 19 June 2014

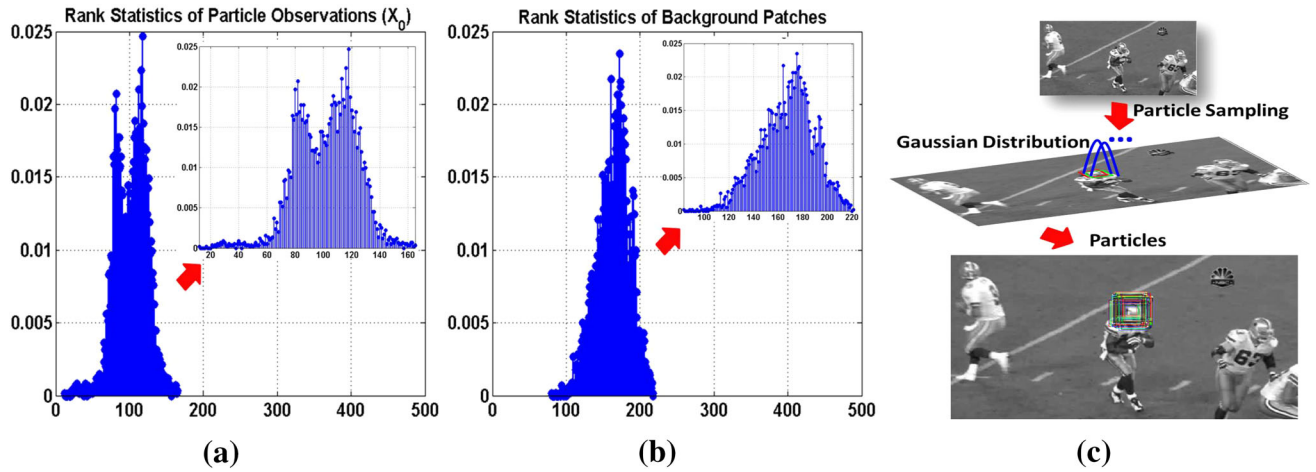

Fig. 1 (**a**) Rank statistics of an image observation matrix $\mathbf{X_0}$ corresponding to sampled particles per frame. The rank statistics are computed from 15 videos with about 6,433 frames. The target objects are initialized with the same size (i.e., 30×30 or $d = 900$ pixels) and the number of particles at each frame n_0 is set to 500, i.e., $\mathbf{X_0} \in \mathbf{R}^{900\times500}$. For each $\mathbf{X_0}$ of each frame, we compute the SVD SVD of $\mathbf{X_0}$ and determine the numerical rank as the number of non-zero singular values whose sum is more than 0.7 times the sum of all singular values. The rank of $\mathbf{X_0}$ in most frames is about 100, which is much lower than d and n_0. Thus, the image observations corresponding to particles at each frame tend to be low-rank. After pruning, the candidate particle observations $\mathbf{X}$ is expected to have lower rank. (**b**) Rank statistics of image observations corresponding to sampled particles from the background. The rank in most frames is about 180, which is larger than the rank of $\mathbf{X_0}$. However, it is smaller than n_o because the background particles are sampled around, but at a sufficient distance from a target object. If the background patches are sampled from cluttered images, the rank is likely to be higher. (**c**) Particles are sampled at and around target with a Gaussian distribution. As a result, an image observation matrix $\mathbf{X_0}$ is likely to have low-rank property

or lighting variations. Nevertheless these formulations entail solving ℓ_1 minimization problems, which is known to be time-consuming. Furthermore, since the target states are usually estimated in a particle filter framework, the computational cost grows linearly with the number of sampled particles. More importantly, these methods learn sparse representations corresponding to drawn particles independently, and do not consider the underlying relationship that constrains them.

To address these problems, we exploit the temporal consistency property as well as the underlying relationship of image observations. By exploiting temporal consistency, irrelevant particles can be pruned by using the previous tracking results, thereby reducing the overall computational cost. After pruning, the linear representations of the candidate particles based on the current dictionary are constrained to be low-rank and sparse. In this work, the low-rank[1] property captures the underlying structure of the image observations corresponding to candidate particles. This structure arises because image observations of candidate particles tend to lie in a low-rank subspace as motivated by the empirical results shown in Fig. 1, where some of them also have very similar visual appearances. Therefore, we exploit sparsity, low-rank constraint, and temporal consistency to learn robust linear representations corresponding to candidate particles for efficient and effective object tracking.

[1] Generally, the matrix of particle representations is not full-rank. It tends to have a low rank that is usually larger than one.

The proposed visual tracking algorithm is developed based on the following observations:

- The optimal particle is the one at which the image observation has the lowest reconstruction error based on the current dictionary consisting of target object templates.
- Temporal consistency should be exploited to constrain the candidate particles and prune irrelevant ones, thereby making the tracking algorithm more efficient. In addition, this property facilitates more stable state predictions and tracking results.
- After pruning, the image observations corresponding to the remaining candidate particles should be highly correlated and thus the matrix of corresponding image observations should have low rank. The relationship between these image observations corresponding to particles should be exploited which has not been used in existing tracking methods based on sparse representation (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010; Li et al. 2011). Figure 1a shows one example where the image observations $\mathbf{X_0}$ of drawn particles at each frame have low-rank. The rank of candidate image observations $\mathbf{X}$ should be much lower after pruning.
- As occlusion and noise significantly affect tracking performance, the error term of an image representation can be sparsely modeled.
- During tracking, an image observation of a candidate particle can be better represented using a dictionary of

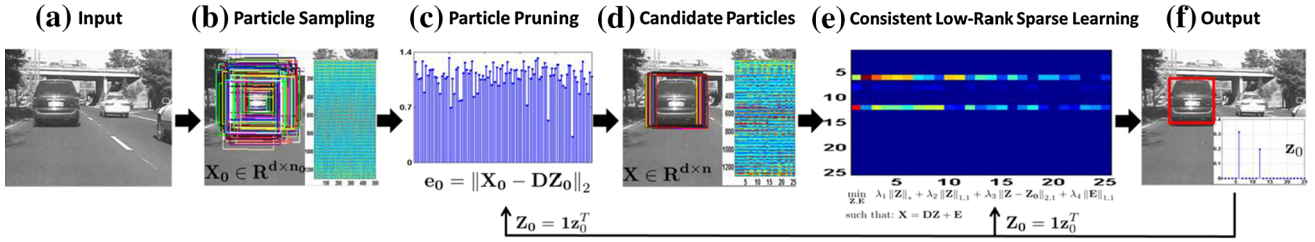

Fig. 2 Enforcing the sparsity, low-rank, and temporal consistency properties in the proposed CLRST algorithm. (**a**) The frame at time t. (**b**) All particles sampled based on previous particles, and their observations $\mathbf{X_0} \in \mathbf{R}^{\mathbf{d}\times\mathbf{n_0}}$. Here, the number of particle is $n_0 = 500$, and an observation is defined to be the grayscale values of each normalized particle in the image. (**c**) Particles are pruned using the reconstruction error $\mathbf{e}_0$, where the k-th element is the ℓ_2 error incurred by linearly representing the k-th particle (i.e., k-th column of $\mathbf{X}_0$) using the previous representation $\mathbf{z}_0$ and a dictionary $\mathbf{D}$ of object and background templates (25 dictionary elements). Here, we define $\mathbf{e}_0 \triangleq \|\mathbf{X_0} - \mathbf{DZ_0}\|_2$, where $\mathbf{Z_0} = \mathbf{1}\mathbf{z}_0^\top$ is a rank one matrix, whose columns are equal to $\mathbf{z}_0$. (**d**) Resulting candidate particles $\mathbf{X} \in \mathbf{R}^{\mathbf{d}\times\mathbf{n}}$ after the particles with the large reconstruction error are pruned. Here, 25 candidate particles are obtained after pruning. Since some of them are similar, they possess a low-rank property, which constrains their representations. (**e**) The representations of n candidate particles using our CLRST algorithm, which enforces sparse, low-rank, and consistent properties. Here, the rank of $\mathbf{X}$ (observations of candidate particles) is 2. (**f**) The tracking result in frame t and its representation $\mathbf{z}_0$

templates from both object and background templates with online update. This emphasizes the importance of representing what a target is and what it is not. Generally, an image observation of a "good" target candidate is effectively represented by the object templates and not the background templates, thereby leading to a sparse representation. Likewise, an image observation of a "bad" target candidate can be more sparsely represented by a dictionary of background templates.

In the proposed algorithm, after pruning, the matrix of candidate image observation $\mathbf{X}$ (where each image observation corresponding to a particle is stored in a column) in the current frame is represented as a linear combination $\mathbf{Z}$ of object and background templates that define a dictionary of templates $\mathbf{D}$. We require that $\mathbf{Z}$ be (1). sparse because only a few templates are required to represent an image observation well, (2). low-rank because the matrix of image observations $\mathbf{X}$ have a low rank structure[2] as observed from the example in Figs. 1a, and 3. temporally consistent because particles in the current frame tend to have representations similar to the representation of the previous tracking result. In addition, we use sparse error $\mathbf{E}$ to account for occlusion and noise in one representation for an image observation. Therefore, the representations of the current particles are computed by solving a low-rank, sparse representation problem. We show that the solution to this problem is obtained by performing a sequence of closed form optimization steps by the Inexact Augmented Lagrange Multiplier method. To account for appearance variations and alleviate tracking drift, we update $\mathbf{D}$ progressively via a sequence of template replacement and re-weighting steps. The object templates of $\mathbf{D}$ are updated only when the object undergoes significant changes, while the background templates are updated at every frame. The particle at which the image region is most similar to the current target is selected as the tracking result. Figure 2 shows the flowchart and how these three properties are enforced in the proposed tracking algorithm.

The contributions of this work are three-fold. (1) We formulate object tracking as a consistent, sparse, as well as low-rank representation problem from a new perspective. This is carried out by exploiting the relationship between the observations of particle samples and jointly representing them using a dictionary of templates with online update. To the best of our knowledge, this is the first work exploiting the low-rank nature underlying the image observations corresponding to sampled particles. (2) We take temporal consistency between the representations of particles into account. It is used to prune particles and constrain their low-rank representations. As a result, representations of candidate particles can be computed jointly and efficiently. The resulting low-rank, sparse, and temporally consistent representation of candidate particles facilitates robust performance for visual tracking. (3) The proposed CLRST algorithm is a generic formulation that encompasses three special cases: the low-rank sparse tracker (LRST), low-rank tracker (LRT), and the sparse tracker (ST). We show how these algorithms are related and the importance of each property for visual tracking.

2 Related Work

The recent years have witnessed much progress in tracking with numerous applications (Yilmaz et al. 2006; Salti et al. 2012; Wu et al. 2013; Zhang et al. 2012b; Kristan and Cehovin 2013; Pang and Ling 2013). In this section, we discuss the most relevant methods to our work.

[2] This follows from the linear representation assumption. Since $\mathbf{X} = \mathbf{DZ}$ and $\mathbf{D}$ can be designed to be an overcomplete full row or column rank matrix, then $\text{rank}(\mathbf{X}) = \text{rank}(\mathbf{Z})$. So, if $\mathbf{X}$ is low-rank, it follows that $\mathbf{Z}$ is also low-rank.

Visual tracking methods can be categorized as generative and discriminative. Generative tracking methods adopt appearance models to represent target objects and search for the most similar image regions (Black and Jepson 1998; Comaniciu et al. 2003; Jepson et al. 2003; Yang et al. 2009; Ross et al. 2008; Adam et al. 2006; Kwon and Lee 2010). In Black and Jepson (1998), an algorithm builds on view-based eigenspace representations, robust estimation techniques, and parameterized optical flow estimation, is proposed for object tracking. The mean shift tracking algorithm (Comaniciu et al. 2003) models a target with nonparametric distributions of features (e.g., color pixels) and locates the object with mode shifts. In Jepson et al. (2003), an adaptive appearance model based on mixture of Gaussians is proposed to model objects with stable components. An adaptive appearance model that accounts for target appearance variation is proposed in the incremental visual tracking (IVT) method (Ross et al. 2008). 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 non-rigid motion as a result of the adopted holistic appearance model. The Frag tracker (Adam et al. 2006) addresses the partial occlusion problem by modeling object appearance with histograms of local patches. The tracking task is carried out by combining votes of matching local patches based on histograms. As the model is not updated, this method is less effective for handling large appearance changes. The visual tracking by decomposition (VTD) method (Kwon and Lee 2010) extends the conventional particle filter framework with multiple motion and observation models to account for large appearance variation caused by change of pose, lighting and scale as well as partial occlusion. As the adopted generative representation scheme is not designed to distinguish between target and background patches, it is prone to drift in complex scenes.

Discriminative methods formulate object tracking as a binary classification with local search which aims to find the target image region that best distinguishes from the background (Avidan 2005; Grabner et al. 2006; Jiang et al. 2011; Babenko et al. 2009). In Collins and Liu (2003), a target confidence map is constructed by finding the most discriminative features based on features of color pixels. The ensemble tracking algorithm (Avidan 2005) formulates the task as a pixel based binary classification problem with local search. Although this method is able to differentiate the target and background, the pixel-based representation is less effective for handling occlusion and clutters. In Grabner et al. (2006), a method based on online adaptive boosting (OAB) is proposed to select discriminative features for object tracking. As each tracking result and model update is based on the object detection of each frame, tracking errors are likely accumulated and thereby causing drifts. To account for ambiguities in selecting the best target location, a boosting approach that extends the multiple instance learning (MIL) framework for online object tracking is developed (Babenko et al. 2009). While it is able to reduce tracking drifts, this method does not handle large nonrigid shape deformation or scale well. A hybrid approach that combines a generative model and a discriminative classifier is proposed to handle appearance changes (Yu et al. 2008).

Sparse linear representation has recently been introduced to object tracking with demonstrated success (Mei and Ling 2011; Mei et al. 2011; Liu et al. 2010; Li et al. 2011; Zhang et al. 2012a,d, 2014; Zhong et al. 2012; Bao et al. 2012). In the ℓ_1 tracking method (Mei and Ling 2011), a candidate region is represented by a sparse linear combination of target and trivial templates where the coefficients are computed by solving a constrained ℓ_1 minimization problem with non-negativity constraints. As this method entails solving one ℓ_1 minimization problem for each particle, the computational complexity is significant. An efficient ℓ_1 tracker with minimum error bound as well as occlusion detection is subsequently developed (Mei et al. 2011. In addition, methods based on dimensionality reduction as well as orthogonal matching pursuit (Li et al. 2011) and efficient numerical solver using an accelerated proximal gradient scheme (Bao et al. 2012) have been developed to make the ℓ_1 tracking method more efficient. In Liu et al. (2010), dynamic group sparsity is incorporated in the tracking problem and high dimensional image features are used to improve tracking performance. Most recently, an algorithm that learns the sparse representations of all particles jointly (Zhang et al. 2012d, 2013a) is proposed for object tracking.

Considerable progress has been made in recent years for solving low rank matrix minimization and completion problems. Since matrix rank is not a convex function, its convex surrogate (i.e., the matrix nuclear norm) is used for approximation and efficiently solved (Cai et al. 2010; Ma et al. 2011; Recht et al. 2010; Peng et al. 2011) with numerous applications including face recognition (Peng et al. 2011), image retrieval (Liu et al. 2012), subspace clustering (Liu et al. 2010), image classification (Zhang et al. 2013b), background subtraction (Candès et al. 2011), and video denoising (Ji et al. 2010), among others.

3 Consistent Low-Rank Sparse Tracking

In this section, we present the proposed tracking algorithm based on temporally consistent low-rank sparse representations of particle samples.

3.1 Consistent Low-Rank Sparse Representation

In this work, particles are sampled around the previous object locations to predict the state $\mathbf{s}_t$ of the target at time

t, from which we crop the region of interest $\mathbf{y}_t$ in the current image and normalize it to the template size. The state transition function $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ is modeled by an affine motion model with a diagonal Gaussian distribution. The observation model $p(\mathbf{y}_t|\mathbf{s}_t)$ reflects the similarity between an observed image region $\mathbf{y}_t$ corresponding to a particle $\mathbf{s}_t$ and the templates of the current dictionary. In this paper, $p(\mathbf{y}_t|\mathbf{s}_t)$ is computed as a function of the difference between the consistent low-rank sparse representation of the target based on object templates, and its representation based on background templates. The particle that maximizes this function is selected to be the tracked target at each time instance.

At time t, we have n_0 sampled particles and the corresponding vectorized gray-scale image observations form a matrix $\mathbf{X_0} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_0}]$, where the observation with respect to i-th particle is denoted as $\mathbf{x}_i \in \mathbb{R}^d$. We represent each observation as a linear combination of templates from a dictionary $\mathbf{D}_t = [\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_m]$, such that $\mathbf{X}_0 = \mathbf{D}_t\mathbf{Z}_t$. Here, the columns of $\mathbf{Z}_t = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{n_0}]$ denote the representations of particle observations with respect to $\mathbf{D}_t$. The dictionary columns contain templates that are used to represent each particle including image observations of the tracked object and the background. Since our representation is constructed on the pixel level, misalignment between dictionary templates and particle observations may lead to tracking drifts. To alleviate this problem, the dictionary $\mathbf{D}_t$ can be constructed from an overcomplete set using the transformed templates of the target and background classes. In addition, this dictionary is progressively updated.

For efficient and effective tracking, we exploit temporal consistency to prune particles. A particle is considered temporally inconsistent if its observation is not linearly represented well by the dictionary $\mathbf{D}_t$ and the representation of the tracked target in the previous frame, denoted as $\mathbf{z}_0$. More specifically, if its ℓ_2 reconstruction error $\|\mathbf{x}_i - \mathbf{D}_t\mathbf{z}_0\|_2$ is above a predefined threshold σ, then it is pruned in the current frame. Temporal consistency is exploited in this work as the appearances of the tracked object and its representations do not vary much in a short time period. Consequently, this process effectively reduces the number of particles to be represented from n_0 to n, where $n_0 \gg n$ in most cases. In what follows, we denote the ones after pruning as *candidate particles*, their corresponding observations as $\mathbf{X} \in \mathbb{R}^{d\times n}$, and their representations as $\mathbf{Z} \in \mathbb{R}^{m\times n}$.

The representation of each candidate particle is based on the following observations. (1) After pruning, the candidate particle observations can be modeled by a low-rank subspace (i.e., $\mathbf{X}$ is low-rank) and therefore $\mathbf{Z}$ (i.e., their representations with respect to $\mathbf{D}_t$) is expected to be low-rank as discussed in Sect. 1. (2) The observation $\mathbf{x}_i$ of a good candidate particle can be modeled by a small number of nonzero coefficients in its corresponding representation $\mathbf{z}_i$. (3) The aim of object tracking is to search patches (with respect to particles) which have a representation similar to previous tracking results. In other words, a "good" representation should be consistent over time.

In this work, we formulate the tracking problem by

$$\min_{\mathbf{Z},\mathbf{E}} \lambda_1 \|\mathbf{Z}\|_* + \lambda_2 \|\mathbf{Z}\|_{1,1} + \lambda_3 \|\mathbf{Z} - \mathbf{Z_0}\|_{2,1} + \lambda_4 \|\mathbf{E}\|_{1,1} \tag{1}$$

$$\text{such that} \quad \mathbf{X} = \mathbf{D}_t\mathbf{Z} + \mathbf{E}$$

where

$$\|\mathbf{Z}\|_{p,q} = \left(\sum\nolimits_j \left(\sum\nolimits_i |[\mathbf{Z}]_{ij}|^p\right)^{\frac{q}{p}}\right)^{\frac{1}{q}} \tag{2}$$

and $\mathbf{E}$ is the error due to noise as well as occlusion. In this formulation, $\lambda_i, i = 1, \ldots, 4$ are weights that quantify the trade-off between different terms discussed below. In addition, $[\mathbf{Z}]_{ij}$ denotes the entry at the i-th row and j-th column of $\mathbf{Z}$. We denote the representation of the *previous* tracking result with respect to $\mathbf{D}_t$ as $\mathbf{z}_0$. The matrix $\mathbf{Z_0} = \mathbf{1}\mathbf{z}_0^\top$ is a rank one matrix, where each column is $\mathbf{z}_0$.

3.1.1 Low-Rank Representation: $\|\mathbf{Z}\|_*$

In our formulation, we minimize the matrix rank of the representations of all candidate particles together. Since the rank minimization problem is known to be computationally intractable (NP-hard) in general, we resort to minimizing its convex envelope using its nuclear norm $\|\mathbf{Z}\|_*$. In contrast to the ℓ_1 tracker, the particles at instance t are represented jointly rather than independently. The proposed joint representation capitalizes on the structure of particle representations which facilitates a more robust and computationally efficient solution. Instead of solving n independent ℓ_1 minimization problems by the interior point method as in the ℓ_1 tracker (Mei and Ling 2011), we consider a single rank minimization problem solved by a sequence of closed form update operations.

3.1.2 Sparse Representation: $\|\mathbf{Z}\|_{1,1}$

The templates in the dictionary $\mathbf{D}_t$ capture possible appearance variations of the target object and background, and only a small number of these templates is required to reliably represent the observation of each candidate particle. This sparse representation scheme has been shown to be robust to occlusion or noise in visual tracking (Mei and Ling 2011; Zhang et al. 2013a).

3.1.3 Temporal Consistency: $\|\mathbf{Z} - \mathbf{Z_0}\|_{2,1}$

To encourage temporal consistency in the representation of the tracking result, we compare the representations of the particles in the current frame $\mathbf{Z}$ to that in the previous frame $\mathbf{z}_0$ using the $\ell_{2,1}$ matrix norm. This approach effectively enforces temporal consistency for visual tracking although more sophisticated methods (e.g., weighted similarity function between $\mathbf{Z}$ and $\mathbf{z}_0$) can be employed. The use of the $\ell_{2,1}$ norm is motivated by its effect on the difference $\Delta\mathbf{Z} = \mathbf{Z} - \mathbf{Z}_0$. This norm encourages sparsity at the level of the columns of $\Delta\mathbf{Z}$ (at the particle level). In other words, the regularization norm encourages the representations of most particles (those represented well by the current object templates in $\mathbf{D}_t$) in the current frame to be similar to that of the previous tracking result. Equivalently, it allows only a small number of particles (those observations not represented well by object templates) to have representations different from the previous tracking result.

3.1.4 Reconstruction Error: $\|\mathbf{E}\|_{1,1}$

For robustness against sparse significant errors (e.g., due to occlusion), we seek to minimize the ℓ_1 norm of each column of $\mathbf{E}$. This sparse error assumption has been adopted in tracking (Mei and Ling 2011) and other applications (Wright et al. 2009). Unlike the ℓ_1 tracker (Mei and Ling 2011) that incorporates sparse error by augmenting $\mathbf{D}_t$ with a large number (i.e., $2d$) of trivial templates and computing the corresponding coefficients, we obtain the reconstruction error $\mathbf{E} \in \mathbb{R}^{d\times n}$. Furthermore, the values and support of columns in $\mathbf{E}$ are informative since they indicate the presence of occlusion (large values but sparse support) and whether a candidate particle is sampled from the background (large values with non-sparse support).

3.1.5 Adaptive Dictionary

The dictionary $\mathbf{D}_t$ is initialized by sampling image patches around the initial target position. For accurate tracking, the dictionary is updated in successive frames to model appearance change of the target object. To alleviate the problem of tracking drift, we augment $\mathbf{D}_t$ with representative templates of the background such that $\mathbf{D}_t = [\mathbf{D}_O\ \mathbf{D}_B]$ where $\mathbf{D}_O$ and $\mathbf{D}_B$ represent the target object and background templates respectively. Thus, the representation $\mathbf{z}_k$ of a particle is composed of an object representation $\mathbf{z}_k^O$ and a background representation $\mathbf{z}_k^B$. The tracking result $\mathbf{y}_t$ at instance t is the particle $\mathbf{x}_i$ such that

$$i = \arg\max_{k=1,\ldots,n} (\|\mathbf{z}_k^O\|_1 - \|\mathbf{z}_k^B\|_1). \tag{3}$$

This encourages the tracking result to be modeled well by object templates and *not* background templates. We also exploit discriminative information to design a systematic procedure for updating $\mathbf{D}_t$ (see Sect. 3.4).

3.2 Discussion

As shown in (1), we propose a generic formulation for robust object tracking with consistent low-rank sparse representation. With different setting of λ_1, λ_2, and λ_3, (1) reduces to various object tracking algorithms as follows:

- Low Rank Tracker (LRT): When $\lambda_2 = \lambda_3 = 0$, the proposed algorithm reduces to a low rank tracker. In this case, only the correlations among candidate particle observations are considered whereas sparsity and temporal consistency properties are not exploited.
- Sparse Tracker (ST): When $\lambda_1 = \lambda_3 = 0$, the proposed algorithm reduces to a sparse tracker. It is similar to the ℓ_1 tracker (Mei and Ling 2011) that encourages each particle observation to be represented well by a small number of templates.
- Low Rank Sparse Tracker (LRST): When $\lambda_1 \neq 0$, $\lambda_2 \neq 0$, and $\lambda_3 = 0$, the resulting tracker reduces to the LRST method (Zhang et al. 2012c). Compared to the LRT and ST methods, the LRST algorithm performs well empirically (Zhang et al. 2012c) as it enforces both the sparsity and low-rank properties.
- Consistent Low Rank Sparse Tracker (CLRST): When $\lambda_1 \neq 0$, $\lambda_2 \neq 0$, and $\lambda_3 \neq 0$, the proposed algorithm generalizes the LRST method with temporal consistency, thereby generating more stable tracking results.

It is worth emphasizing the difference between the proposed CLRST algorithm and several related tracking methods (Mei and Ling 2011; Zhang et al. 2012d).

- The ℓ_1 tracker (Mei and Ling 2011) can be considered as a special case of Zhang et al. (2012d) and the proposed CLRST algorithm. In the ℓ_1 tracker (Mei and Ling 2011), the sparse representations of particles are learned for each particle independently. However, in Zhang et al. (2012d) and CLRST, multi-task learning and low-rank sparse learning are adopted respectively to consider the relationship among particle observations.
- Both the MTT (Zhang et al. 2012d) and CLRST algorithms consider the structure among particle observations. However, the assumptions to model this structure are different. In the MTT method, the tracking problem is formulated within the multi-task learning framework by using L_{21} norm ($\|\mathbf{Z}\|_{2,1}$) such that particle observations are modeled by only a few (but the same) dictionary

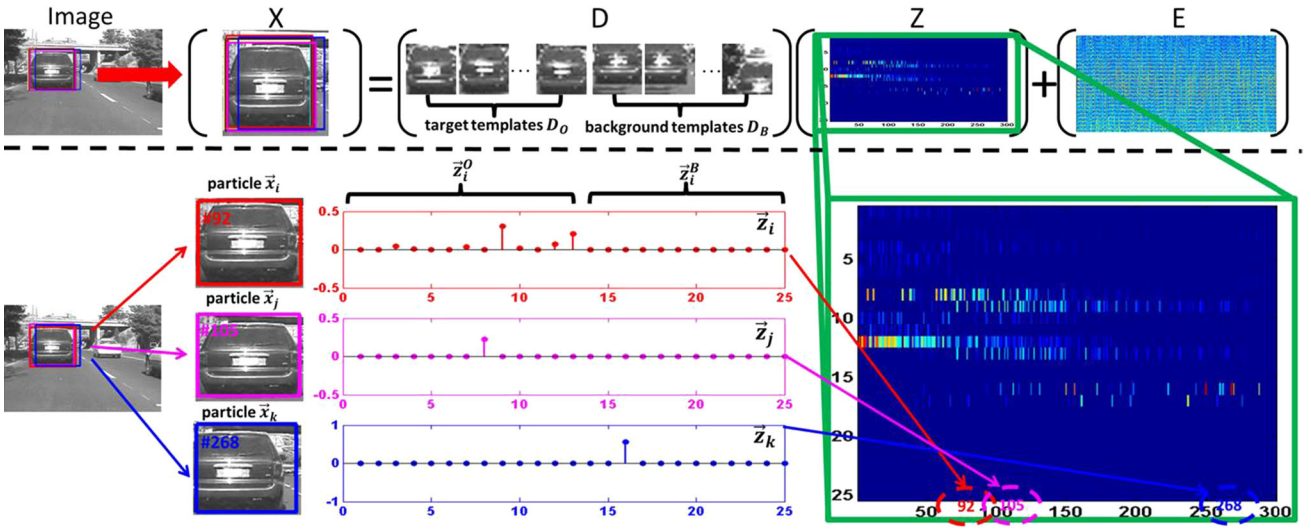

Fig. 3 Schematic example of the proposed CLRST algorithm. The representation $\mathbf{Z}$ of sample particles (after pruning) $\mathbf{X}$ with respect to dictionary $\mathbf{D}$ (set of object and background templates) is learned by solving (1). Notice that $\mathbf{Z}$ is sparse (i.e., few dictionary templates are used) and low-rank (i.e., dictionary templates are reused for representation). Here, the rank of $\mathbf{Z}$ is 11. The particle $\mathbf{x}_i$ is selected among all other particles as the tracking result, since $\mathbf{x}_i$ is best represented by object templates only

elements to make all columns of $\mathbf{Z}$ similar to each other, which indirectly forces the rank of the representation matrix $\mathbf{Z}$ to be one. Different from the MTT tracker, in the CLRST algorithm, the tracking problem is posed within the low-rank learning framework which enforces the target observations to lie in a low dimensional subspace without explicitly requiring the same dictionary elements. Different from the MTT tracker, by design the CLRST algorithm assumes the rank of the representation matrix $\mathbf{Z}$ low which can be one or greater than one. In the proposed tracking method, the particles are randomly sampled with a Gaussian distribution near a target object, and the observations of some particles may be different to each other. Consequently the rank of the representation matrix $\mathbf{Z}$ is not necessarily one as assume by the MTT tracker. Compared with the MTT tracker, the assumption of the CLRST algorithm is less restricted and more amenable to outlier particles. Experimental results in Sect. 5 provide empirical evidence of this crucial algorithmic difference.

– The use of low-rank property facilitates learning effective sparse representation for object tracking. We use the ℓ_1 tracker as one example to show why the low-rank property helps object tracking. In the ℓ_1 tracker, the representation of each particle is learned independently. Due to the low-rank property, our tracker learns the representations of all particles jointly. Namely, the sparse representations of observations are learned jointly by considering all particles and the low-rank property (i.e., solving one rank minimization problem by a sequence of closed form update operations). However, in the ℓ_1 tracker, each particle is processed independently without considering the other particles (i.e., solving n ℓ_1 optimization problems where n is the number of particles). Thus, the learned sparse representation by our algorithm is more compact and robust for object tracking.

Algorithm 1: *Consistent Low-Rank Sparse Tracking (CLRST) Algorithm.*

Input:

- Current frame at t
- Dictionary template $\mathbf{D_{t-1}} = [\mathbf{D_O}\ \mathbf{D_B}]$
- All n particles $\mathbf{s}_{t-1}$
- Representation of previous tracking result $\mathbf{z_0}$

Output:

- Tracked target $\mathbf{y}_t$
- Current states $\mathbf{s}_t$
- Updated target templates $\mathbf{D_t} = [\mathbf{D_O}\ \mathbf{D_B}]$
- Updated representation $\mathbf{z_0}$

1: Generate n_0 particles
2: Obtain mapped observations corresponding to all particles $\mathbf{s}_t$ to get $\mathbf{X_0}$
3: Use consistency property to prune and obtain candidate particle observations $\mathbf{X}$ based on the reconstruction error $\|\mathbf{X_0} - \mathbf{DZ_0}\|_2$, where $\mathbf{Z_0} = \mathbf{1}\mathbf{z}_0^\top$ (see Sect. 3.1 for details).
4: Compute low-rank sparse representation $\mathbf{Z}$ for $\mathbf{X}$ by solving (1) with Algorithm 3
5: Calculate difference score $\Delta z_i = \|\mathbf{z}_i^O\|_1 - \|\mathbf{z}_i^B\|_1, i = 1, 2, \ldots, n$
6: Set $p(\mathbf{y}_t|\mathbf{s}_t)$ as the difference score for each particle
7: Select the particle with the highest value of Δz_i as the current tracking result
8: Update $\mathbf{z_0}$ based on the tracking result $\mathbf{z}_i$
9: Update dictionary template via Algorithm 2

Algorithm 1 shows the main steps of our CLRST method, and Fig. 3 illustrates how the candidate particle observations

are used for tracking. Given observations of all pruned particles $\mathbf{X}$ (e.g., 300 particles sampled around the tracked car are retained) and the current dictionary $\mathbf{D} = [\mathbf{D_O}\ \mathbf{D_B}]$, we learn the representation matrix $\mathbf{Z}$ by solving (1). Note that smaller values are shown with darker color. Clearly, $\mathbf{Z}$ is sparse (i.e., small number of templates used) and low-rank (i.e., templates are reused among particles). The particle observation $\mathbf{x}_i$ is selected as the current tracking result $\mathbf{y}_t$ as its difference ($\|\mathbf{z}_i^O\|_1 - \|\mathbf{z}_i^B\|_1$) is largest among all particles. Since the particle observation $\mathbf{x}_j$ can be considered as a misaligned representation of the target, it is not modeled well by the object dictionary $\mathbf{D}_O$ (i.e., $\mathbf{z}_j^O$ has small values). On the other hand, the particle observation $\mathbf{x}_k$ is represented well by the background dictionary $\mathbf{D}_B$ (i.e., $\mathbf{z}_k^B$ has large values). As illustrated in this example, the tracking drift problem is alleviated by the proposed formulation.

3.3 Solving (1)

Unlike existing algorithms that only focus on one of the two convex and non-smooth regularizers (based on ℓ_1 or low-rank constraints) the objective function in (1) consists of both terms. To solve this complex objective function, we introduce three equality constraints and slack variables:

$$\min_{\mathbf{Z}_{1-4},\mathbf{E}} \lambda_1 \|\mathbf{Z}_1\|_* + \lambda_2 \|\mathbf{Z}_2\|_{1,1} + \lambda_3 \|\mathbf{Z_3}\|_{2,1} + \lambda_4 \|\mathbf{E}\|_{1,1}$$
$$\text{such that: } \begin{cases} \mathbf{X} = \mathbf{D}\mathbf{Z}_4 + \mathbf{E} \\ \mathbf{Z}_4 = \mathbf{Z}_1 \\ \mathbf{Z}_4 = \mathbf{Z}_2 \\ \mathbf{Z}_4 = \mathbf{Z}_3 + \mathbf{Z}_0. \end{cases} \tag{4}$$

This transformed problem can be minimized using the conventional Inexact Augmented Lagrange Multiplier (IALM) method that has attractive convergence properties for non-smooth optimization which has been used in matrix rank minimization problems (Peng et al. 2011). It is an iterative method that augments the conventional Lagrangian function with quadratic penalty terms that allow closed form updates for each unknown variable. The updates are in closed form from (5) to (7), where $\mathcal{S}_\lambda([\mathbf{A}]_{ij}) = \text{sign}([\mathbf{A}]_{ij}) \max(0, |[\mathbf{A}]_{ij}| - \lambda)$ is the soft-thresholding operator, $\mathcal{J}_\lambda(\mathbf{A}) = \mathbf{U}_A \mathcal{S}_\lambda(\Sigma_A) \mathbf{V}_A^\top$ is the singular value soft-thresholding operator, and $\mathcal{L}_\lambda(\mathbf{a}_i) = \max(0, 1 - \frac{\lambda}{\|\mathbf{a}_i\|_2})\mathbf{a}_i$. We denote $\mathbf{a}_i$ as the i-th column of matrix $\mathbf{A}$ and $\mathbf{A} = \mathbf{U_A} \Sigma_\mathbf{A} \mathbf{V_A}^\top$ as the singular value decomposition. The technical details of solving this optimization problem are presented in Sect. 4.

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \left[\frac{1}{2} \|\mathbf{X} - \mathbf{A}\|_F^2 + \lambda \|\mathbf{X}\|_{1,1} \right] = \mathcal{S}_\lambda(\mathbf{A}) \tag{5}$$

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \left[\frac{1}{2} \|\mathbf{X} - \mathbf{A}\|_F^2 + \lambda \|\mathbf{X}\|_* \right] = \mathcal{J}_\lambda(\mathbf{A}) \tag{6}$$

Algorithm 2: *Dictionary Template Update*

Input:

- Target templates of $\mathbf{D_O}$.
- Background templates of $\mathbf{D_B}$.
- $\mathbf{y}_t$, which is the newly chosen tracking target.

Output:

- Dictionary $\mathbf{D_t} = [\mathbf{D_O}\ \mathbf{D_B}]$.

Initialize: $\alpha = 0$ and ϵ are predefined parameters at $t = 1$.
$\mathbf{z}_i$ is the solution of (1). Set $\Delta z_i = \|\mathbf{z}_i^O\|_1 - \|\mathbf{z}_i^B\|_1$.
$\boldsymbol{\omega}$ is the current weight vector of templates in $\mathbf{D}_O$.
Update weights according to the coefficients of the target templates: $\omega_k \leftarrow \omega_k \exp(\mathbf{z}_i^O(k)) \forall k = 1, \cdots, m_O$.
$\alpha \leftarrow \max(\alpha, \Delta z_i)$.
if $(\Delta z_i < \epsilon\alpha)$ **then**
  $\alpha \leftarrow 0$; $r \leftarrow \arg\min_{k=1,\ldots,m_O} \omega_k$.
  $\mathbf{D}_O(:, r) \leftarrow \mathbf{y}_t$. /*replace template with $\mathbf{y}_t$ */
  $\omega_r \leftarrow \text{median}(\boldsymbol{\omega})$. /*replace weight*/
end if
Normalize $\boldsymbol{\omega}$ such that $\|\boldsymbol{\omega}\|_1 = 1$.
Update $\mathbf{D}_B$ based on the current tracking result.

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \left[\frac{1}{2} \|\mathbf{X} - \mathbf{A}\|_F^2 + \lambda \|\mathbf{X}\|_{2,1} \right] = \mathcal{L}_\lambda(\mathbf{A}) \tag{7}$$

3.4 Dictionary Update

It is well known that tracking algorithms with a fixed appearance dictionary of templates is not effective to account for appearance change in complex scenes. However, small errors are likely to be introduced and accumulated if the templates are updated too frequently, thereby making the tracker drift away from the target. Numerous approaches have been proposed for template update to alleviate the tracking drift problem (Matthews et al. 2004; Kaneko and Hori 2003). In this work, we address this issue by dynamically updating the templates in $\mathbf{D}_t$.

To initialize the object and background dictionaries, we sample equal-sized patches at and around the initial position of the target. We shift the initial bounding box by 1 to 3 pixels in each dimension similar to Huang et al. (2009), Mei and Ling (2011) and thus obtain $m_O = 13$ object templates for the object dictionary D_O. In addition, we initialize the background dictionary $\mathbf{D}_B$, with image patches randomly sampled at a sufficient distance from the surrounding background based on the initial tracking result in a way similar to Grabner et al. (2006), Babenko et al. (2009), Zhong et al. (2012) and obtain $m_B = 12$ background templates. All templates are normalized to half the size of the target object manually initialized.

Each object template in $\mathbf{D}_O$ is associated with a weight ω_i proportional to the frequency that it is selected for tracking. The weight of an object template in $\mathbf{D}_O$ is updated based on how frequently that template is used in representing the current tracking result $\mathbf{z}_i$ (computed from (3)). If $\mathbf{z}_i$ is adequately

represented (based on a predefined threshold) by the current dictionary, then there is no need to update it. Otherwise, the object template with the smallest weight is replaced by the current tracking result, and its weight is set to the median of the current normalized weight vector $\boldsymbol{\omega}$. The main steps of the template update for $\mathbf{D}_O$ are summarized in Algorithm 2. On the other hand, the background dictionary $\mathbf{D}_B$ is updated at every frame by re-sampling patches at a sufficient distance from the tracking result.

4 Optimization

In this section, we present algorithmic details on how to solve the optimization problem (4). By introducing augmented Lagrange multipliers (ALM) to incorporate the equality constraints into the objective function, we obtain the Lagrangian function in (8) that can be optimized through a sequence of simple closed form update operations in (9) where $\mathbf{Y}_1$, $\mathbf{Y}_2$, $\mathbf{Y}_3$ and $\mathbf{Y}_4$ are Lagrange multipliers, and $u_1 > 0$, $u_2 > 0$, $u_3 > 0$ and $u_4 > 0$ are four penalty parameters.

$$
\begin{aligned}
&L(\mathbf{Z}_{1,\ldots,4}, \mathbf{E}, \mathbf{Y}_{1,\ldots,4}, u_{1,\ldots,4}) \\
&\quad = \lambda_1 \|\mathbf{Z}_1\|_* + \lambda_2 \|\mathbf{Z}_2\|_{1,1} + \lambda_3 \|\mathbf{Z}_3\|_{2,1} + \lambda_3 \|\mathbf{E}\|_{1,1} \\
&\qquad + \mathrm{tr}\left[\mathbf{Y}_1^\top (\mathbf{X} - \mathbf{D}\mathbf{Z}_4 - \mathbf{E})\right] + \frac{u_1}{2} \|\mathbf{X} - \mathbf{D}\mathbf{Z}_4 - \mathbf{E}\|_F^2 \\
&\qquad + \mathrm{tr}\left[\mathbf{Y}_2^\top (\mathbf{Z}_4 - \mathbf{Z}_1)\right] + \frac{u_2}{2} \|\mathbf{Z}_4 - \mathbf{Z}_1\|_F^2 \\
&\qquad + \mathrm{tr}\left[\mathbf{Y}_3^\top (\mathbf{Z}_4 - \mathbf{Z}_2)\right] + \frac{u_3}{2} \|\mathbf{Z}_4 - \mathbf{Z}_2\|_F^2 \\
&\qquad + \mathrm{tr}\left[\mathbf{Y}_4^\top (\mathbf{Z}_4 - \mathbf{Z}_3 - \mathbf{Z}_0)\right] + \frac{u_3}{2} \|\mathbf{Z}_4 - \mathbf{Z}_3 - \mathbf{Z}_0\|_F^2
\end{aligned} \tag{8}
$$

$$
\Rightarrow \min_{\mathbf{Z}_{1,\ldots,4}, \mathbf{E}, \mathbf{Y}_{1,\ldots,4}, u_{1,\ldots,4}} L(\mathbf{Z}_{1,\ldots,4}, \mathbf{E}, \mathbf{Y}_{1,\ldots,4}, u_{1,\ldots,4}) \tag{9}
$$

The above problem can by solved by either exact or inexact ALM algorithms (Glowinski and Marrocco 1975; Gabay and Mercier 1976; Boyd et al. 2011), and we take an inexact approach in this work for computational efficiency. The main steps of this exact approach are summarized in Algorithm 3, and the convergence properties can be proved similar to those in Boyd et al. (2011). In (9), the variables can be viewed as two groups: local variables ($\mathbf{Z}_1, \mathbf{Z}_2, \mathbf{Z}_3, \mathbf{E}$) and global variable $\mathbf{Z}_4$. By updating these two group variables iteratively, the convergence can be guaranteed (Boyd et al. 2011). We note that an IALM algorithm is an iterative method that solves for each variable in a coordinate descent manner. That is, each iteration of IALM involves update of each variable one at a time, with the other variables fixed to their most recent values. Consequently, we obtain six update steps corresponding to all the variables. We note that steps 3-8 of Algorithm 3 all have closed form solutions.

Step 1. Update $\mathbf{Z}_1$: Updating $\mathbf{Z}_1$ requires the solution to the problem (10) which can be computed in closed form (11), where $\mathcal{J}_\lambda(\mathbf{X}) = \mathbf{U}\mathcal{S}_\lambda(\Sigma)\mathbf{V}^\top$ is a thresholding operator with respect to a singular value λ; $\mathcal{S}_\lambda(\mathbf{X}_{ij}) = \mathrm{sign}(\mathbf{X}_{ij})\max(0, |\mathbf{X}_{ij}| - \lambda)$ is the soft-thresholding operator; and $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^\top$ is the singular value decomposition of $\mathbf{X}$.

$$
\mathbf{Z}_1^* = \arg\min_{\mathbf{Z}_1} \frac{\lambda_1}{u_2} \|\mathbf{Z}_1\|_* + \frac{1}{2} \left\| \mathbf{Z}_1 - \mathbf{Z}_4 - \frac{1}{u_2}\mathbf{Y}_2 \right\|_F^2 \tag{10}
$$

$$
\Rightarrow \boxed{\mathbf{Z}_1^* = \mathcal{J}_{\frac{\lambda_1}{u_2}}\left(\mathbf{Z}_4 + \tfrac{1}{u_2}\mathbf{Y}_2\right)} \tag{11}
$$

Step 2. Update $\mathbf{Z}_2$: $\mathbf{Z}_2$ is updated by solving the optimization problem (12) with the closed form solution (13).

$$
\mathbf{Z}_2^* = \arg\min_{\mathbf{Z}_2} \frac{\lambda_2}{u_3} \|\mathbf{Z}_2\|_{1,1} + \frac{1}{2} \left\| \mathbf{Z}_2 - \mathbf{Z}_4 - \frac{1}{u_3}\mathbf{Y}_3 \right\|_F^2 \tag{12}
$$

$$
\Rightarrow \boxed{\mathbf{Z}_2^* = \mathcal{S}_{\frac{\lambda_2}{u_3}}\left(\mathbf{Z}_4 + \tfrac{1}{u_3}\mathbf{Y}_3\right)} \tag{13}
$$

Step 3. Update $\mathbf{Z}_3$: $\mathbf{Z}_3$ is updated by solving the optimization problem (14) with the closed form solution (15).

$$
\mathbf{Z}_3^* = \arg\min_{\mathbf{Z}_3} \frac{\lambda_3}{u_4} \|\mathbf{Z}_3\|_{2,1} + \frac{1}{2} \left\| \mathbf{Z}_3 - \mathbf{Z}_4 + \mathbf{Z}_0 - \frac{1}{u_4}\mathbf{Y}_4 \right\|_F^2 \tag{14}
$$

$$
\Rightarrow \boxed{\mathbf{Z}_3^* = \mathcal{L}_{\frac{\lambda_3}{u_4}}\left(\mathbf{Z}_4 - \mathbf{Z}_0 + \tfrac{1}{u_4}\mathbf{Y}_4\right)} \tag{15}
$$

Step 4. Update $\mathbf{E}$: $\mathbf{E}$ is updated by solving the optimization problem (16) with the closed form solution (17).

$$
\mathbf{E}^* = \arg\min_{\mathbf{E}} \frac{\lambda_4}{u_1} \|\mathbf{E}\|_{1,1} + \frac{1}{2} \left\| \mathbf{E} - \mathbf{D}_t\mathbf{Z}_4 + \mathbf{X} - \frac{1}{u_1}\mathbf{Y}_1 \right\|_F^2 \tag{16}
$$

$$
\Rightarrow \boxed{\mathbf{E}^* = \mathcal{S}_{\frac{\lambda_4}{u_1}}\left(\mathbf{D}_t\mathbf{Z}_4 - \mathbf{X} + \tfrac{1}{u_1}\mathbf{Y}_1\right)} \tag{17}
$$

Algorithm 3: *Consistent Low-Rank Sparse Representation (Solving (4))*

Input : data matrix $\mathbf{X}$, parameters λ_1, λ_2, λ_3, λ_4, and ρ
Output: $\mathbf{Z}$ and $\mathbf{E}$

Initialize $\mathbf{Z_4} = \mathbf{0}, \mathbf{E} = 0, \mathbf{Y}_1 = 0, \mathbf{Y}_2 = 0, \mathbf{Y}_3 = 0, \mathbf{Y}_4 = 0$
while *not converged* **do**
  Fix other variables and update $\mathbf{Z}_1$ (11)
  Fix other variables and update $\mathbf{Z}_2$ (13)
  Fix other variables and update $\mathbf{Z}_3$ (15)
  Fix other variables and update $\mathbf{E}$ (17)
  Fix other variables and update $\mathbf{Z}_4$ (19)
  Update multipliers and parameters (20)
  Update final solution $\mathbf{Z} \leftarrow \mathbf{Z}_4$
end

Step 5. Update $\mathbf{Z}_4$: $\mathbf{Z}_4$ is updated by solving the optimization problem (18) with the closed form solution (19).

$$\begin{aligned}\mathbf{Z}_4^* = \arg\min_{\mathbf{Z}_4} \operatorname{tr}\left[\mathbf{Y}_1^\top(\mathbf{X}-\mathbf{D}\mathbf{Z}_4-\mathbf{E})\right] \\ + \frac{u_1}{2}\|\mathbf{X}-\mathbf{D}\mathbf{Z}_4-\mathbf{E}\|_F^2 \\ + \operatorname{tr}[\mathbf{Y}_2^\top(\mathbf{Z}_4-\mathbf{Z}_1)] + \frac{u_2}{2}\|\mathbf{Z}_4-\mathbf{Z}_1\|_F^2 \\ + \operatorname{tr}[\mathbf{Y}_3^\top(\mathbf{Z}_4-\mathbf{Z}_2)] + \frac{u_3}{2}\|\mathbf{Z}_4-\mathbf{Z}_2\|_F^2 \\ + \operatorname{tr}[\mathbf{Y}_4^\top(\mathbf{Z}_4-\mathbf{Z}_3-\mathbf{Z}_0)] + \frac{u_4}{2}\|\mathbf{Z}_4-\mathbf{Z}_3-\mathbf{Z}_0\|_F^2 \end{aligned} \tag{18}$$

$$\Rightarrow \boxed{\mathbf{Z}_4^* = \mathbf{G_1}\left[\mathbf{D}^\top(\mathbf{X}-\mathbf{E}) + \mathbf{G_2} + \mathbf{G_3}\right]} \tag{19}$$

where $\mathbf{G_2} = \frac{u_2}{u_1}\mathbf{Z}_1 + \frac{u_3}{u_1}\mathbf{Z}_2 + \frac{u_4}{u_1}(\mathbf{Z}_3+\mathbf{Z}_0)$, $\mathbf{G_1} = \left(\mathbf{D}^\top\mathbf{D} + \frac{u_2}{u_1}\mathbf{I} + \frac{u_3}{u_1}\mathbf{I} + \frac{u_4}{u_1}\mathbf{I}\right)^{-1}$, and $\mathbf{G_3} = \frac{1}{u_1}\left(\mathbf{D}^\top\mathbf{Y}_1 - \mathbf{Y}_2 - \mathbf{Y}_3 - \mathbf{Y}_4\right)$.

Step 6. Update Multipliers $\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3, \mathbf{Y}_4$: We update the Lagrange multipliers (20), where $\rho > 1$.

$$\begin{cases} \mathbf{Y}_1 = \mathbf{Y}_1 + u_1(\mathbf{X}-\mathbf{D}\mathbf{Z}_4-\mathbf{E}) \\ \mathbf{Y}_2 = \mathbf{Y}_2 + u_2(\mathbf{Z}_4-\mathbf{Z}_1) \\ \mathbf{Y}_3 = \mathbf{Y}_3 + u_3(\mathbf{Z}_4-\mathbf{Z}_2) \\ \mathbf{Y}_4 = \mathbf{Y}_4 + u_4(\mathbf{Z}_4-\mathbf{Z}_3-\mathbf{Z}_0) \\ u_1 = \rho u_1;\ u_2 = \rho u_2;\ u_3 = \rho u_3;\ u_4 = \rho u_4 \end{cases} \tag{20}$$

The IALM algorithm that solves (4) is shown in Algorithm 3, where convergence is reached when the change in the objective function or solution $\mathbf{Z}$ is below a pre-defined threshold (e.g., $\tau = 10^{-3}$ in this work). In addition, we set $u_1 = u_2 = u_3 = u_4$. Here, we note that other penalty update rules and stopping criteria can be used for this optimization problem as discussed in Boyd et al. (2011).

Computational Complexity: At each frame, each of the n_0 particles drawn in the LRST, ST, and LRT methods requires solving one optimization as no temporal consistency is exploited for pruning. In the proposed CLRST algorithm, the number of particles is pruned from n_0 to n candidate particles which are represented jointly by solving (1). The computational bottleneck of the LRT, LRST and CLRST methods lies in computing singular value decomposition (SVD) of $\mathbf{Z}$. The time complexity for solving one SVD of an $m \times n$ matrix is $\mathcal{O}(\min\{mn^2, m^2n\})$ in general. Since $m < n$ in this work, the computational complexity of SVD here is $\mathcal{O}(m^2n)$. This complexity does not account for the low-rank nature of $\mathbf{Z}$. It is worthwhile noting that more computationally efficient SVD methods exist for sufficiently low-rank matrices (Brand 2006) which can be used to reduce the time complexity of the proposed algorithm. For the LRT and LRST methods, the time complexity is $\mathcal{O}(m^2n_0)$. In addition, we do not need to perform a matrix inverse operation at each iteration to compute $\mathbf{G}_1$ in Step 5. All what it entails is the inversion operation once with the complexity of $\mathcal{O}(m^3)$. In this case, (19) reduces to simple matrix multiplication operations, and its time complexity is $\mathcal{O}(m^2n)$. As a result, the total computation cost is $\mathcal{O}(m^3 + m^2n\epsilon^{-0.5})$, where the number of IALM iterations is $\mathcal{O}(\epsilon^{-0.5})$. In contrast, the time complexity for LRT and LRST is $\mathcal{O}(m^3 + m^2n_0\epsilon^{-0.5})$.

The proposed ST tracking method is efficient as the solution involves soft-thresholding operations. This complexity is similar to that of the recent ℓ_1 tracking algorithms (Bao et al. 2012; Li et al. 2011). We note that several fast techniques (Mei et al. 2011; Bao et al. 2012; Li et al. 2011) can also be applied to the proposed tracking methods for additional speedup. In contrast, the computational complexity of the ℓ_1 tracker (Mei and Ling 2011) is $\mathcal{O}(n_0d^2)$ since the number of dictionary templates (object and trivial) is $(m + 2d)$ and n_0 Lasso problems are solved independently. Empirical results show that the proposed trackers are at least two orders of magnitude faster than ℓ_1 tracker in general. For example, the average run-time for the CLRST and ℓ_1 methods are 0.63 and 340 seconds respectively, when $m = 25$, $n_0 = 400$, and $d = 32 \times 32$. Furthermore, better tracking results can be achieved with larger values of m and d without increasing computational overhead significantly. Compared to the ST, LRT, and LRST methods, the CLRST algorithm is most efficient, since it involves fewer particles (as $n \ll n_0$ from the pruning process).

5 Experimental Results

In this section, we present experimental results on evaluation of the proposed tracking algorithm against several state-of-the-art methods.

5.1 Datasets

For thorough evaluations, we use a set of 25 challenging videos with ground truth object locations including *One Leave Shop Reenter1cor (OLSR)*, *One Shop One Wait2cor (OSOW)*, *biker*, *bolt*, *car11*, *car4*, *carchase*, *coke11*, *david indoor (david)*, *faceocc2*, *faceocc*, *football*, *girl*, *panda*, *shaking*, *singer*, *singer low frame rate (singerlfr)*, *skating*, *skating low frame rate (skatinglfr)*, *soccer*, *surfer*, *sylv*, *trellis70*, *tudcrossing*, and *volkswagen*.

These videos contain complex scenes with challenging factors, e.g., cluttered background, moving camera, fast movement, large variation in pose and scale, occlusion, shape deformation and distortion.

5.2 Evaluated Algorithms

We compare the proposed tracking methods (CLRST, LRST, LRT and ST) with 14 state-of-the-art visual trackers including visual tracking by decomposition (VTD) (Kwon and Lee 2010), ℓ_1 tracker (Mei and Ling 2011), incremental visual tracking (IVT) method (Ross et al. 2008), online multiple instance learning (MIL) (Babenko et al. 2009) method, fragments-based (Frag) (Adam et al. 2006) tracking method online Adabost boosting (OAB) method (Grabner et al. 2006), multi-task tracking (MTT) (Zhang et al. 2012d) method, circulant structure tracking (CST) method (Henriques et al. 2012), real time compressive tracking (RTCT) method (Zhang et al. 2012a), tracking by detection (TLD) method (Kalal et al. 2010), context-sensitive tracking (CT) method (Dinh et al. 2011), distribution field tracking (DFT) method (Sevilla-Lara and Learned-Miller 2012), sparse collaborative model (SCM) (Zhong et al. 2012), and Struck (Hare et al. 2011). For fair comparisons, we use the publicly available source or binary codes provided by the authors. In addition, we use the same initialization and parameter settings in all experiments.

5.3 Evaluation Criteria

Two metrics are used to evaluate tracking performance. The first metric is the center location error which is the Euclidean distance between the central location of a tracked target and the manually labeled ground truth. The other is an overlap ratio based on the PASCAL challenge object detection score (Everingham et al. 2010). Given the tracked bounding box ROI_T and the ground truth bounding box ROI_{GT}, the overlap score is computed as $score = \frac{area(ROI_T \cap ROI_{GT})}{area(ROI_T \cup ROI_{GT})}$. To rank the tracking performance, we compute the average overlap score across all frames of each image sequence.

Table 1 Effects of σ on the average number of particles after pruning (average particle number), average computational cost per frame (seconds), and average overlap score

σ	0.5	0.8	1.0	1.2	1.5
Average particle numbers	17	76	177	369	439
Average computational cost	0.34	0.75	1.78	3.71	3.97
Average overlap score	0.63	0.67	0.70	0.62	0.62

5.4 Implementation Details

All our experiments are carried out in MATLAB on a 2.66 GHz Intel Core2 Duo machine with 6 GB RAM. The template size d is set to half the size of the target object manually initialized in the first frame. We use the affine transformation where the state transitional probability, $p(\mathbf{s}_t|\mathbf{s}_{t-1})$, is modeled by a zero-mean Gaussian distribution and a diagonal covariance matrix $\boldsymbol{\sigma_0}$ with values (0.005, 0.0005, 0.0005, 0.005, 4, 4): $p(\mathbf{s}_t|\mathbf{s}_{t-1}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\sigma_0})$. The definition of $p(\mathbf{y}_t|\mathbf{s}_t)$ is $p(\mathbf{y}_t|\mathbf{s}_t) = \Delta z_i, i = 1, 2, \ldots, n$. The representation threshold ϵ in Algorithm 2 is set to 0.5. The parameter σ is set to 1.0 in the CLRST method to prune particles. The number of particles n_0 is set to 500 and the total number of templates m is set to 25.

We first do parameter analysis for our proposed method and show how the parameters affect the performance and how to decide their values in Sect. 5.5.3. We present qualitative and quantitative results of the tracking methods in Sects. 5.6 and 5.7. The run-time performance is discussed in Sect. 5.8. The videos are available at http://faculty.ucmerced.edu/mhyang/project/clrst, and the MATLAB source code will be made available soon.

5.5 Parameter Analysis

Several parameters play important roles in the proposed tracking algorithm. In this section, we show how to determine their values and their effects on tracking performance.

5.5.1 Effect of σ

As discussed in Sect. 3.1, a particle $\mathbf{x}_i$ is pruned if the corresponding ℓ_2 reconstruction error $\|\mathbf{x}_i - \mathbf{D}_t\mathbf{z}_0\|_2$ is above a predefined threshold σ. Thus, σ is closely related to the number of particles after pruning, the computational cost per frame, and the tracking performance. To analyze the effect of σ on tracking performance, we use different σ on five videos with 1912 frames. To simplify this problem, we assume that σ can be parameterized by a discrete set $\Sigma = \{0.5, 0.8, 1.0, 1.2, 1.5\}$. Experimental results, shown in Table 1, indicate that it is a good trade-off to set the value of σ to 1.

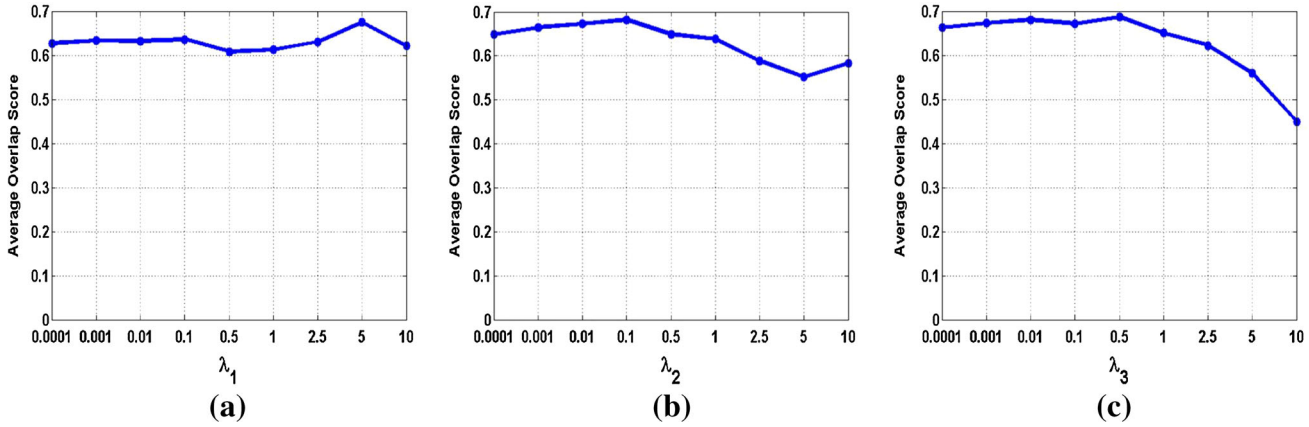

Fig. 4 Effects of λ_1, λ_2, and λ_3 on tracking performance

5.5.2 *Effect of* λ

There are four parameters λ_1, λ_2, λ_3, and λ_4 in the objective function (1). To demonstrate the effects of these parameters, we fix one parameter value one at a time and vary the values of the others. As λ_1, λ_2, and λ_3 are related to the coefficients $\mathbf{Z}$, and λ_4 is related to $\mathbf{E}$, we fix the $\lambda_4 = 1$ and change other parameter values.

For sensitivity analysis, all the λ_i, $i = 1, \ldots, 3$ are parameterized by a discrete set Λ, where Λ={$1e^{-4}$, $1e^{-3}$, 0.01, 0.1, 0.5, 1.0, 2.5, 5.0, 10.0}. We evaluate different combinations of these values on five videos with 1912 frames. For each combination, we compute the average overlap score from all frames. As a result, for each $\lambda_1 \in \Lambda$ with different λ_2 and λ_3, we obtain 9×9 average overlap scores. Given a fixed λ_1, the tracking performance is stable for different λ_2 and λ_3, we average the 9×9 scores. For different λ_1, we obtain the corresponding results as shown in Fig. 4a. Figure 4 shows the sensitivity analysis of λ_i, $i = 1, \ldots, 3$. Overall, the proposed algorithm is robust to different settings as long the value is within reasonable ranges. From on these results, we can set $\lambda_1 = 5$, $\lambda_2 = 0.1$, $\lambda_3 = 0.5$, and $\lambda_4 = 1.0$ for the objective function (1).

5.5.3 *Adaptive Parameter Learning:* λ_3

Temporal consistency allows for more stable tracking especially when the appearance of the tracked target is smoothly varying over time. As shown in (1), λ_3 is the weight of temporal consistency and decides how important this property is for object tracking at each time instance. Here, we denote the representation of the current tracking result as $\mathbf{z}$ and its representation in the previous frame as $\mathbf{z}_0$. At a time instance when the object undergoes smooth appearance change (i.e., small values of $\Delta z_0 = \|\mathbf{z} - \mathbf{z}_0\|_2$), temporal consistency is useful for robust object tracking, so the parameter λ_3 should be large in this case. However, when appearance significant change

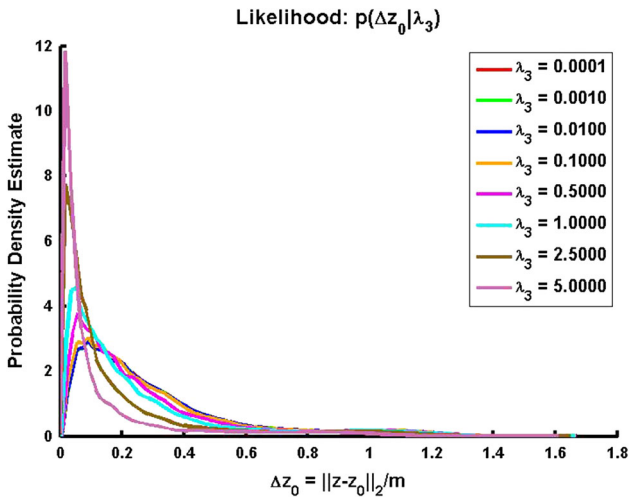

Fig. 5 Learning a non-parametric likelihood model $p(\Delta z_0|\lambda_3)$ for a discrete value set of λ_3. When the change in representation is small (i.e., Δz_0 is small), the choice of λ_3 is important and the most likely value of λ_3 is inversely proportional to Δz_0. When the change in representation is significant (i.e., Δz_0 is large), the temporal consistency constraint is suppressed (i.e., $\lambda_3 = 0$.)

(i.e., large values of Δz_0) occurs, temporal consistency is not helpful and should be suppressed. In this case, λ_3 should be set to a small value. As such, this parameter should be adaptively changed for different frames in the same sequence, as well as, across different sequences. In this section, we fix the values of the parameters λ_1, λ_2 and λ_4 as obtained in Sect. 5.5.2, and discuss how λ_3 (related to temporal consistency) can be adaptively designed based on target appearance changes.

The basic idea of adaptive parameter learning is to learn the distribution of pair pattern $(\lambda_3, \Delta z_0)$ from the training videos, which are not used for test sequences. At each frame, our goal is to select the best λ_3 and its corresponding Δz_0 with the labeled target position (ground truth) in the training set. The learning details are as follows. Given one frame, for

each $\lambda \in \Lambda$, we obtain the corresponding tracking result by solving (1). To simplify the problem, we further assume that λ_3 can be parameterized by a discrete set denoted as Λ. In this work, we set $\Lambda = \{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}, 0.01, 0.05, 0.1, 0.5, 1.0, 2.5, 5.0\}$. We then use the ground truth at that frame to select the closest one from all the tracking results and obtain the best pair pattern $(\lambda_3, \Delta z_0)$. Consequently, for each value $\lambda_3 \in \Lambda$, we construct its likelihood $p(\Delta z_0|\lambda_3)$ with non-parametric kernel density estimation, and obtain a family of distributions, each of which is estimated in a data-driven manner. A set of these probabilities are shown in Fig. 5. We note that all the curves have single modes, which validate our assumption (small values of Δz_0 occur at larger values of λ_3 and vice versa). We also observe that some curves ($\lambda_3 = 0.0001, 0.001, 0.01$) are similar with almost the same tracking results. Given a test frame with Δz_0, we obtain the best λ_3 by selecting $\lambda_3 \in \Lambda$ that maximizes the likelihood and use the λ_3 for the next unseen video frame.

5.6 Qualitative Comparison

Figures 6 and 7 show the tracking results of 18 trackers on 25 sequences. The tracking results are discussed below based on the main challenging factors in each video.

Occlusion: In the *OLSR* sequence shown in Fig. 6a, the target woman walking down the corridor is partially occluded by a man. The OAB method loses track of the woman and starts to follow the man when partial occlusion occurs at frame 28. Other trackers are able track the woman accurately except the MIL and TLD methods. Among all trackers, the MTT, CLRST, and DFT methods perform well. Some tracking results on the *OSOW* sequence are shown in Fig. 6b. All methods track the target person well where limited partial occlusion occurs. Figure 6c shows some results on the *faceocc* sequence where the target face is heavily occluded by a magazine. Most sparse trackers perform well in this sequence whereas the OAB, RTCT, MIL and CT methods drift away from the target when heavy occlusion occurs. Figure 6d shows some results on the *tudcrossing* sequence where the target vehicle is occluded by crossing pedestrians. The MIL, VTD, OAB, CST, Struck, and CT methods drift away from the target object when occlusion occurs. On the other hand, the ℓ_1, DFT, TLD, and the proposed CLRST methods perform well in this sequence.

Illumination: Some tracking results of the *car4* sequence are shown in Fig. 6e. The OAB, Frag, DFT, and VTD methods start to drift from the target at frame 187 when the vehicle goes underneath the overpass. The MIL algorithm starts drift away from the target object at frame 200, and the RTCT and SDG methods start to drift at frame 233, The ST, CT, TLD, Struck, and ℓ_1 methods are able to track the target although with some errors. On the other hand, the target object is successfully tracker by the IVT, LRT, LRST and CLRST algorithms throughout the entire sequence despite large illumination changes. Figure 6f shows some tracking results on the *car11* sequence. Most trackers (RCT, TLD, MIL, OAB, CT, ST, and ℓ_1) drift away from the target vehicle at different frames. The Struck, IVT, VTD, LRT, LRST and CLRST methods perform well throughout the whole sequence. Overall, the CLRST, Struck, and LRST trackers perform better than the others as shown in Tables 2 and 3. The *singer* sequence contains significant illumination, scale, and viewpoint changes and most trackers drift away from the target object as shown in Fig. 6g. The VTD and proposed trackers perform well in this sequence. The *singerlfr* sequence contains the same scenes as the *singer* video except at low frame rate. Similar to the tracking results in the *singer* sequence, the proposed tracking algorithms (except the ST approach) and the VTD method perform well in the *singerlfr* sequence.

Background Clutter: The *football* sequence includes scenes with cluttered background. The RTCT, OAB, Frag, CT, ℓ_1, and ST trackers drift away from the target at different frames when similar objects appear in close proximity to the target object. Overall, the proposed LRT, LRST and CLRST algorithms successfully track the target object. Some tracking results on the *soccer* sequence are shown in Fig. 6j. The CLRST and LRST methods accurately track the target object despite scale and pose changes as well as occlusion by confetti. In contrast, other methods (IVT, Struck, ℓ_1, OAB, MIL, and Frag) fail to track the target object reliably.

Illumination and Pose Variation: Figure 6k shows some tracking results on the *david* sequence where the Frag, RTCT, CT, VTD, Struck, and SDG methods fail at different frames. The OAB method drifts away from the target object at frame 550. The MIL and ℓ_1 trackers adequately track the target, but with certain drift, especially at frames 690 and 500, respectively. The IVT, MTT, DFT, TLD, CST ST, LRT, LRST and CLRST algorithms are able to track the target accurately throughout the sequence. The *trellis70* sequence is captured in an outdoor environment where the object appearance changes significantly as a result of cast shadows, motion and head pose variation. Figure 6l shows some results where almost all the methods do not perform well in this sequence. However, the CLRST, LRST, LRT, and MTT methods perform better than the others. Some tracking results on the sequence *sylv* are shown Fig. 6m. The IVT tracker fails at frame 613 as a result of a combination of large pose and illumination change. The Struck, ST, LRT, LRST and CLRST methods are able to track the target in this sequence, whereas the Frag, MIL, VTD, OAB, DFT, CT, CST, and ℓ_1 trackers slightly drift away from the target.

Occlusion and Pose Variation: The can in the *coke11* sequence is frequently occluded and rotated. Figure 7a shows some tracking results where the CLRST, LRST, LRT, ST, ℓ_1, SDG, MTT, Struck, OAB, and MIL methods perform

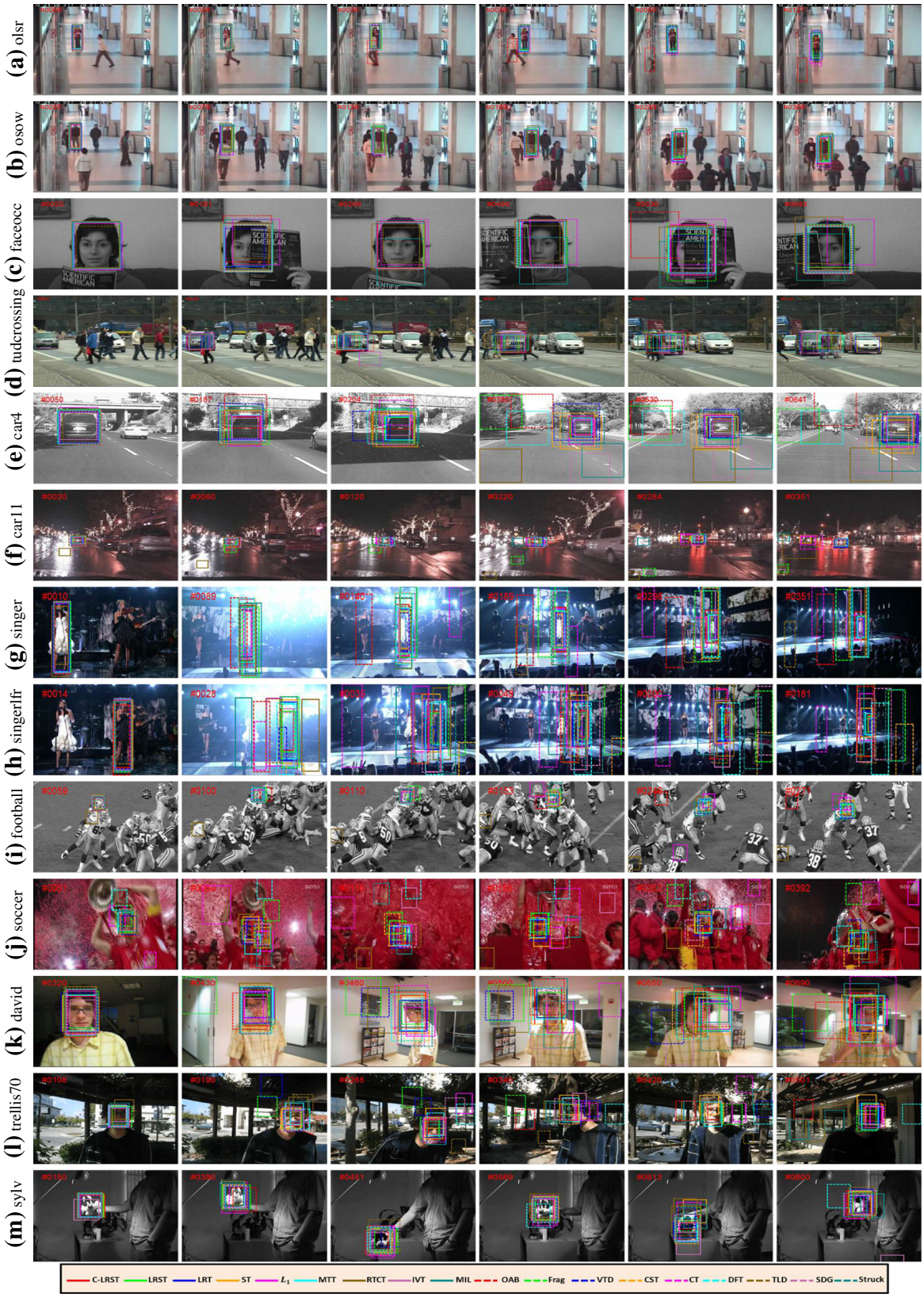

Fig. 6 Tracking results of 18 trackers (denoted in *different colors*) on 13 image sequences. Frame numbers are displayed *red*. See text for details. Results best viewed on high-resolution displays (Color figure online)

Fig. 7 Tracking results of 18 trackers (denoted in *different colors*) on 12 image sequences. Frame numbers are displayed in *red*. See text for details. Results best viewed on high-resolution displays (Color figure online)

Table 2 The average center location error of 18 different trackers on 25 different image sequences

Video	CLRST	LRST	LRT	ST	ℓ_1	MTT	RTCT	IVT	MIL	OAB	Frag	VTD	CST	CT	DFT	TLD	SDG	Struck
OLSR	**1.8**	3.0	1.9	1.9	1.9	*1.7*	8.3	1.9	9.8	68.3	4.0	2.4	1.9	4.1	**1.8**	10.9	2.0	5.0
OSOW	*2.3*	6.8	2.6	2.9	2.8	2.5	15.2	3.0	11.6	4.6	5.6	2.7	4.6	2.7	3.8	11.1	**2.4**	4.7
Biker	*11.8*	27.7	59.0	25.3	29.4	**14.0**	16.0	76.8	29.6	22.0	104.4	17.3	18.4	121.6	122.6	86.9	73.7	48.0
Bolt	**8.4**	*7.3*	11.9	39.2	110.2	95.0	22.3	87.4	9.9	108.2	17.9	16.1	9.0	147.6	36.9	105.8	35.9	144.3
Car11	*1.3*	2.5	2.7	21.5	19.2	1.9	117.8	5.4	53.8	5.7	72.7	3.7	2.1	17.6	8.1	29.0	2.9	**1.8**
Car4	*2.0*	2.6	**2.1**	12.7	8.5	2.2	86.3	6.4	53.8	88.1	127.3	27.0	18.3	9.2	89.6	6.9	59.4	2.3
Carchase	**3.3**	9.2	44.2	44.2	21.7	10.9	19.1	18.5	20.4	4.2	11.1	44.4	3.7	5.9	46.3	3.9	3.9	*2.5*
Coke11	*3.6*	4.7	7.0	6.0	12.1	7.3	11.1	58.5	13.7	11.3	71.0	62.7	5.9	5.8	16.3	11.6	5.2	**4.0**
David	*9.1*	13.5	13.3	15.8	16.2	16.0	32.4	**13.1**	30.3	26.4	73.0	64.9	19.7	36.0	15.4	16.6	61.6	46.7
Faceocc2	8.5	6.6	*5.8*	6.2	15.2	8.1	**6.0**	6.5	10.2	20.8	48.2	11.8	**6.0**	8.3	7.2	13.3	10.3	6.5
Faceocc	8.4	9.6	6.0	7.5	7.0	7.7	19.0	9.1	34.3	17.2	17.9	8.7	**4.5**	31.2	4.7	14.8	*4.3*	8.4
Football	*3.4*	4.4	4.3	14.6	15.4	4.7	123.3	5.2	8.0	53.3	6.3	**3.7**	7.2	41.7	5.2	6.0	4.1	6.9
Girl	3.8	4.0	4.7	*3.2*	5.0	4.5	17.4	4.2	12.4	11.0	7.4	11.4	38.2	12.7	19.1	8.3	**3.3**	18.6
Panda	8.3	10.3	7.9	8.5	34.8	9.8	**6.2**	39.7	**6.2**	7.5	*5.8*	30.3	92.4	55.7	33.7	22.2	8.6	*5.8*
Shaking	*2.7*	3.9	4.4	**3.2**	37.8	8.4	86.6	52.2	7.9	100.3	15.3	4.0	13.6	33.9	95.4	21.0	4.0	54.9
Singer	*1.5*	1.9	**1.6**	2.1	5.3	1.8	5.9	9.8	11.1	63.0	26.9	1.9	6.9	65.6	6.6	44.1	1.8	4.5
Singerlfr	4.5	*1.7*	6.3	8.6	20.3	**3.7**	13.3	15.7	42.3	20.5	51.9	9.8	98.1	93.5	24.2	16.6	39.8	43.0
Skating	*4.5*	5.0	4.8	**4.6**	20.1	7.4	84.9	74.9	49.2	39.3	63.3	5.0	5.2	35.7	57.4	99.3	28.0	51.9
Skatinglfr	*2.7*	**3.1**	3.4	29.5	31.1	3.4	62.4	57.9	44.0	39.8	53.3	6.2	41.5	61.3	80.8	52.6	128.0	45.7
Soccer	**11.8**	*8.7*	15.4	15.1	58.5	14.3	79.6	97.8	46.3	65.3	41.4	10.5	35.9	72.3	62.4	29.8	42.2	41.0
Surfer	10.0	14.0	10.3	10.2	28.0	22.3	29.8	75.1	8.4	**8.1**	186.1	8.7	78.4	*7.9*	139.4	12.5	111.4	9.2
Sylv	*4.4*	**5.2**	5.4	5.9	14.5	4.8	13.5	39.4	15.3	10.4	6.8	7.4	6.8	6.5	10.9	**5.2**	5.6	*4.4*
Trellis70	*5.8*	**6.8**	7.4	14.5	31.1	10.3	42.4	54.0	37.3	41.5	55.7	47.8	7.6	26.0	60.1	50.9	23.6	28.3
Tudcrossing	17.3	30.2	35.8	46.2	*6.8*	14.3	55.1	25.9	51.2	26.2	10.8	43.1	58.0	26.7	**10.6**	16.7	12.9	17.8
Volkswagen	5.6	4.9	**3.5**	3.6	181.3	9.3	288.0	130.5	19.4	10.1	241.6	11.0	182.9	10.0	253.5	**3.5**	4.2	*1.7*

On average, the proposed trackers (ST, LRT, LRST and CLRST) outperform the other 14 state-of-the-art trackers. For each sequence, the smallest and second smallest distances are denoted in italic and bold, respectively

well throughout the entire sequence. Tracking results on the *faceocc2* sequence are shown in Fig. 7b where most trackers drift away from the target when it is heavily occluded. As the proposed CLRST algorithm exploits sparse and low-rank representation as well as temporal consistency to account for occlusion, it performs well in this sequence. Figure 7c shows tracking results on the *panda* sequence where the target undergoes out-of-plane pose variation and shape deformation. The Frag, OAB, Struck, and MIL methods perform well which can be attributed to the local representation schemes or local discriminative features. While the proposed tracking methods perform as well as the above methods, the other trackers fail to track the target.

Abrupt Motion and Pose Variation: The *biker* sequence contains scenes with abrupt motion and large pose variation as shown in Fig. 7d. Most methods fail to track the target objects well when the target undergoes out-of-plane rotation and abrupt motion. Nevertheless, both MTT and CLRST algorithms perform well throughout the entire sequence with more stable tracking results. Figure 7e shows tracking results of the *surfer* sequence where the target person undergoes acrobat movements with 360 degrees out of plane rotation. The RTCT, DFT and Frag methods start to drift at frames 402, 415, and 418, respectively. The SDG, CST, and IVT methods drift at frames 480, 556, and 562 due to the abrupt motion. On the other hand, the CT, OAB, Struck, MIL and CLRST methods perform well. In the *bolt* sequence (Fig. 7f), several objects appear in the same scenes with rapid appearance change due to shape deformation and fast motion. The CLRST, LRST, LRT, MIL, and CST are able to track the target object in most frames. However, the IVT, VTD, ℓ_1, DFT, and MTT methods do not perform well when similar objects appear near the target.

Abrupt Motion, Pose Variation and Occlusion: Figure 7g shows tracking results for the *girl* sequence. The ST, LRT, LRST, CLRST, SDG, MTT and ℓ_1 methods are capable of tracking the target for the entire sequence. Other trackers experience drift at different time instances (Struck at frame

Table 3 The average overlap score of 18 different trackers on 25 different videos

Video	CLRST	LRST	LRT	ST	ℓ_1	MTT	RTCT	IVT	MIL	OAB	Frag	VTD	CST	CT	DFT	TLD	SDG	Struck
OLSR	**0.87**	0.77	0.86	0.86	0.86	*0.88*	0.71	0.86	0.67	0.17	0.78	0.81	0.84	0.73	0.86	0.68	0.86	0.77
OSOW	*0.88*	0.74	0.83	0.83	**0.87**	0.86	0.56	0.83	0.56	0.71	0.77	0.86	0.81	0.81	0.82	0.65	**0.87**	0.81
Biker	**0.59**	0.45	0.30	0.44	0.39	*0.68*	0.45	0.31	0.43	0.44	0.27	0.49	0.45	0.39	0.27	0.30	0.36	0.38
Bolt	0.60	*0.67*	0.53	0.36	0.20	0.03	0.31	0.02	0.56	0.02	0.44	0.46	**0.63**	0.07	0.37	0.16	0.35	0.17
Car11	*0.86*	0.76	0.73	0.46	0.52	0.80	0.00	0.51	0.22	0.55	0.10	0.66	0.80	0.43	0.52	0.28	0.68	**0.83**
Car4	*0.88*	**0.86**	*0.88*	0.49	0.62	0.80	0.24	0.74	0.27	0.22	0.23	0.47	0.47	0.64	0.23	0.57	0.30	0.49
Carchase	**0.82**	0.67	0.46	0.47	0.59	0.58	0.29	0.44	0.53	0.80	0.60	0.38	0.81	0.74	0.40	0.76	0.77	*0.85*
Coke11	**0.73**	0.72	0.71	0.72	0.46	0.68	0.47	0.10	0.43	0.41	0.06	0.06	0.61	0.59	0.43	0.45	0.64	*0.74*
David	*0.70*	0.50	0.56	0.53	0.50	0.53	0.41	0.36	0.42	0.43	0.23	0.26	0.50	0.39	0.57	**0.60**	0.30	0.38
Faceocc2	0.71	0.74	**0.78**	0.77	0.67	0.74	0.54	*0.79*	0.72	0.59	0.38	0.70	0.77	0.72	**0.78**	0.57	0.73	0.77
Faceocc	0.84	0.82	0.89	0.86	0.86	0.84	0.73	0.84	0.58	0.77	0.87	0.82	*0.92*	0.55	**0.91**	0.57	*0.92*	0.85
Football	*0.74*	0.69	**0.72**	0.64	0.45	0.66	0.02	0.64	0.52	0.23	0.59	0.74	0.57	0.39	0.68	0.60	0.70	0.60
Girl	**0.72**	0.69	0.68	*0.74*	0.68	0.71	0.32	0.68	0.45	0.53	0.60	0.55	0.35	0.56	0.38	0.59	0.71	0.41
Panda	0.49	0.41	0.48	0.45	0.27	0.43	0.52	0.14	0.51	0.48	**0.53**	0.29	0.15	0.35	0.18	0.35	0.47	*0.56*
Shaking	*0.81*	0.72	0.72	**0.78**	0.18	0.55	0.02	0.02	0.58	0.01	0.41	0.72	0.44	0.45	0.15	0.33	0.72	0.15
Singer	*0.79*	0.76	**0.77**	0.70	0.70	0.76	0.45	0.48	0.41	0.18	0.26	0.63	0.47	0.24	0.47	0.40	0.65	0.46
Singerlfr	**0.61**	*0.70*	0.37	0.32	0.19	0.50	0.20	0.24	0.10	0.12	0.13	0.30	0.11	0.11	0.22	0.16	0.16	0.14
Skating	*0.66*	0.60	**0.62**	0.61	0.47	0.51	0.01	0.07	0.23	0.37	0.19	0.61	0.55	0.38	0.20	0.07	0.51	0.29
Skatinglfr	*0.78*	0.71	**0.72**	0.40	0.53	**0.72**	0.12	0.11	0.21	0.26	0.21	0.57	0.35	0.16	0.15	0.29	0.16	0.36
Soccer	0.32	*0.38*	0.28	0.28	0.14	0.31	0.15	0.14	0.12	0.10	0.19	**0.35**	0.24	0.10	0.10	0.17	0.16	0.13
Surfer	0.48	0.41	0.51	0.51	0.16	0.27	0.15	0.16	**0.57**	*0.59*	0.03	0.56	0.21	0.53	0.03	0.41	0.05	0.56
Sylv	*0.82*	**0.78**	0.75	0.76	0.58	0.73	0.59	0.47	0.58	0.67	0.73	0.74	0.72	0.70	0.63	0.70	**0.78**	**0.78**
Trellis70	*0.78*	**0.76**	0.72	0.57	0.38	0.60	0.22	0.39	0.35	0.46	0.29	0.31	0.72	0.52	0.32	0.21	0.49	0.50
Tudcrossing	0.68	0.51	0.61	0.57	*0.84*	0.67	0.32	0.56	0.38	0.56	0.68	0.40	0.36	0.61	0.67	**0.71**	0.67	0.61
Volkswagen	0.51	0.54	**0.65**	0.64	0.01	0.40	0.01	0.02	0.27	0.37	0.01	0.24	0.01	0.60	0.02	*0.66*	0.62	0.63

On average, the proposed trackers (ST, LRT, LRST and CLRST) outperform the other 14 state-of-the-art trackers. For each sequence, the best and the second best scores are denoted in italic and bold, respectively

59, CT at frame 84, TLD at frame 120, RTCT at frame 216, DFT at frame 240, Frag at frame 248, CST at frame 310, MIL at frame 430, OAB and IVT at frame 436, and VTD at frame 477). In the *volkswagen* sequence, the target vehicle undergoes abrupt object motion, pose variation, and occlusions. Some the tracking results at frames are shown in Fig. 7h. The CST, IVT, ℓ_1, RTCT, MTT, OAB, and Frag methods fail to track the target around frame 10; the CT and VTD algorithms start to drift due to occlusions at frame 293 and 331, respectively; and the MIL tracker fails at frame 754. The other trackers (CLRST, LRST, LRT, ST, Struck, SDB, and TLD) are able to track the target almost throughout the entire sequence. Figure 7i shows some tracking results of the *carchase* sequence. The DFT method starts to drift at frame 139; the VTD, RTCT, IVT, and CT algorithms drift at frame 169 when occlusion occurs; and the LRT and ST algorithms fail at frame 233. The CLRST, TLD, CST, Struck, and OAB trackers perform well throughout this sequence.

Abrupt Motion, Illumination Change, and Occlusion: In the *shaking* sequence, the appearance of the target object changes significantly due to pose and illumination variations as well as occlusion. The proposed algorithms (ST, LRT, LRST and CLRST) successfully track the object as shown in Fig. 7j. The VTD, MIL, SDG, and MTT algorithms track the object well except for some large errors around frames 60 and 260. Other methods (OAB, IVT, ℓ_1, Struck, and Frag) fail to track the object when the the combined changes of pose and illumination as well as occlusion occur. The *skating* and *skatinglfr* sequences contain abrupt object motion, large changes of illumination, scale, and viewpoints, as well as occlusions. The VTD and proposed methods (ST, LRT, LRST and CLRST) are able to account for appearance well as shown in Fig. 7k. In Fig. 7l, the proposed methods (LRT, LRST and CLRST) track the target reliably despite fast object motion (as a result of low frame rate). Moreover, our CLRST performs slightly better than the MTT and VTD algorithms (e.g., frame 299) in this sequence.

5.7 Quantitative Comparison

Tables 2 and 3 show the center location errors and the overlap scores of 18 trackers on the 25 challenging image sequences.

Table 4 Average run-time for each frame (in seconds) of 7 trackers (CLRST, LRST, LRT, ST, ℓ_1 Mei and Ling 2011, ℓ_1^* Bao et al. 2012, and MTT Zhang et al. 2012d) with varying template sizes d and number of particles n_0

d	32×32							48×48						
	ℓ_1	ℓ_1^*	MTT	ST	LRT	LRST	CLRST	ℓ_1	ℓ_1^*	MTT	ST	LRT	LRST	CLRST
n_0														
100	84.5	0.31	0.41	0.61	0.66	0.79	0.71	601.3	1.42	2.6	1.19	1.51	1.34	0.93
200	178.9	0.56	1.46	1.19	2.04	1.35	0.75	1,238.3	2.23	5.6	2.73	3.11	2.74	1.00
300	256.4	0.65	2.71	1.79	1.95	2.36	0.65	1,969.9	3.51	5.0	4.09	4.50	4.26	0.79
400	340.5	1.20	4.69	2.51	3.03	3.03	0.63	2,665.2	4.66	11.5	5.24	5.78	6.52	0.86
500	422.5	1.37	4.85	3.15	3.32	3.47	0.71	3,158.6	6.26	19.4	6.16	8.19	7.12	0.77
600	497.1	1.38	5.97	3.74	4.29	4.35	0.60	3,915.9	7.93	21.0	7.31	8.33	8.58	0.74
700	572.3	1.81	6.17	4.13	4.72	4.66	0.55	4,429.9	9.33	22.4	9.14	9.32	10.04	0.75
800	659.4	1.94	7.62	4.61	5.41	5.39	0.75	5,080.4	9.54	23.8	10.05	10.85	10.97	0.76

For the proposed CLRST method, n is smaller than n_0 after particle pruning

The proposed trackers achieve the best or second best results (based on both criteria) in most sequences. Compared to the ST method, the LRT algorithm achieves much better results which shows merits of exploiting the underlying structure of candidate particle representations via low-rank constraints. Furthermore, the LRST algorithm outperforms the ST and LRT methods, which shows the advantages of jointly exploiting both low-rank and sparsity constraints. The CLRST algorithm performs better than the LRST method, which indicates that the use of temporal consistency facilitates visual tracking. In addition, we note that the CLRST tracking algorithm is more efficient than the other three trackers (see Sect. 5.8).

Among the sparse trackers, the LRST algorithm outperforms both ST and ℓ_1 method which can be explained by the difference of exploiting the underlying constraints among particle representations jointly. Compared with the MTT tracker that models the correlations among particle representations in a multi-task learning framework, the CLRST algorithm is computationally more efficient with better performance. The assumption of the MTT method is strict because it inherently assumes that the learned representations of all candidate particles should be described with a few but the *same* dictionary elements. However, the CLRST algorithm does not assume these representations to be similar but rather belong to a low-dimensional subspace (i.e., the matrix consisting of all candidate particle representations is low-rank). In addition, by exploiting the temporal consistency property, our CLRST achieves better runtime performance.

5.8 Run-Time Performance

Tracking algorithms based on sparse representations and particle filters (Mei and Ling 2011; Zhang et al. 2012d; Bao et al. 2012; Li et al. 2011) have been demonstrated to perform well in visual tracking against other methods (Kwon and Lee 2010; Ross et al. 2008; Babenko et al. 2009; Adam et al. 2006; Grabner et al. 2006; Kalal et al. 2010). However, the run-time of sparse trackers grows proportionally as the number of particles and templates in the dictionary. Table 4 shows the run-time performance of state-of-the-art algorithms based on sparse representation (Mei and Ling 2011; Zhang et al. 2012d; Bao et al. 2012; Li et al. 2011) and the proposed algorithms (ST, LRT, LRST, CLRST)[3]. The ℓ_1^* tracker (Bao et al. 2012) is more efficient than the ℓ_1 tracker by pruning particles and an approximate gradient descent algorithm. On the other hand, the MTT tracker (Zhang et al. 2012d) makes use of the correlation among particles to improve robustness and reduce computational load.

Table 4 shows that our trackers are more efficient than the ℓ_1 method. For example, when $m = 25$, $n = 400$, and $d = 1024$, the average run-time for the CLRST and ℓ_1 trackers are 0.63 and 340 seconds per frame respectively (i.e., CLRST is 560 times faster than ℓ_1). Note that, most of the computational gains with respect to the ℓ_1 method are due to the preliminary pruning process. In addition, the CLRST tracker is also more efficient than the real time ℓ_1^* method (Bao et al. 2012). Among all trackers, the CLRST algorithm is 4 to 5 times faster than the LRST method ($n = 400$, and $d = 1024$). This speedup can be attributed to the use of temporal consistency to prune particles. It is known that tracking performance tends to be better when m and d are increased. For the ℓ_1 tracker, the computational cost increases significantly with m and d. However, this increase is much less significant with the proposed tracking methods as shown in Table 4. While some trackers, e.g., RTCT (Zhang et al. 2012a) and TLD (Kalal et al. 2010), are faster than the proposed algorithms, the tracking performance results are less

[3] The results of Li et al. (2011) are not included in Table 4 since the source code is not available for evaluation and the implementation requires technical details as well as parameter settings not discussed.

accurate (Mei and Ling 2011; Zhang et al. 2012d; Bao et al. 2012; Li et al. 2011). Furthermore, the computational load of the CLRST algorithm can be further reduced by decreasing the target template size d (e.g., by projecting the templates onto a lower dimensional subspace).

6 Conclusion

In this paper, we propose a robust and efficient particle-filter based tracking algorithm that exploits the consistent, low-rank, and sparse nature of candidate particle representations using a dictionary of object and background templates. We model visual tracking as a low-rank sparse learning problem that is regularized by temporal consistency at the level of particles, and present an efficient solution. We extensively analyze the performance of our tracking algorithms against a number of competing state-of-the-art methods on 25 challenging image sequences. Qualitative and quantitative experimental results show that the proposed tracking algorithms outperform state-of-the-art methods, especially in the presence of partial occlusions, pose variations, illumination changes, and abrupt motion.

Acknowledgments This work is supported in part 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) and NSF CAREER Grant #1149783.

References

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).

Avidan, S. (2005). Ensemble tracking. In *Proceedings of 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 *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 983–990).

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*.

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.

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

Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, *415*(1), 20–30.

Cai, J., Candes, E., & Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, *20*(4), 1956–1982.

Candès, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal component analysis? *Journal of the ACM*, *58*(3), 11:1–11:37.

Collins, R. T., & Liu, Y. (2003). On-line selection of discriminative tracking features. In *Proceedings of the IEEE 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.

Dinh, T., Vo, N., & Medioni, G. (2011). Context tracker: Exploring supporters and distracters in unconstrained environments. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1177–1184).

Everingham, M., Gool, L., Williams, C., Winn, J., & Zisserman, A. (2010). The pascal visual object class (voc) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Gabay, D., & Mercier, B. (1976). A dual algorithm for the solution of nonlinear variational problems via finite element approximations. *Computers and Mathematics with Applications*, *2*(1), 17–40.

Glowinski, R., & Marrocco, A. (1975). Sur l'approximation, par elements finis d'ordre un, et la resolution, par penalisation—dualite, d'une classe de problemes de dirichlet non lineares. *Revue Francaise dAutomatique, Informatique, et Recherche Operationelle*, *9*(1), 41–76.

Grabner, H., Grabner, M., & Bischof, H. (2006). Real-time tracking via on-line boosting. In *Proceedings of British Machine Vision Conference* (pp. 1–10).

Hare, S., Saffari, A., & Torr, P. (2011). Struck: Structured output tracking with kernels. In *Proceedings of the IEEE International Conference on Computer Vision*.

Henriques, J., Caseiro, R., Martins, P., & Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In *Proceedings of European Conference on Computer Vision*.

Huang, J., Huang, X., & Metaxas, D. (2009). Learning with dynamic group sparsity. In *Proceedings of the IEEE International Conference on Computer Vision*.

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.

Ji, H., Liu, C., Shen, Z., & Xu, Y. (2010). Robust video denoising using low rank matrix completion. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

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).

Kalal, Z., Matas, J., & Mikolajczyk, K. (2010). P-N learning: Bootstrapping binary classifiers by structural constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

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).

Kristan, M., & Cehovin, L., et al. (2013). The visual object tracking vot2013 challenge results. In *ICCV2013 Workshops, Workshop on Visual Object Tracking Challenge*.

Kwon, J., & Lee, K. M. (2010). Visual tracking decomposition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1269–1276).

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).

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).

Liu, G., Lin, Z., & Yu, Y. (2010). Robust subspace segmentation by low-rank representation. In *Proceedings of the International Conference on Machine Learning*.

Liu, S., Song, Z., Liu, G., Xu, C., Lu, H., & Yan, S. (2012). Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Ma, S., Goldfarb, D., & Chen, L. (2011). Fixed point and bregman iterative methods for matrix rank minimization. *Journal Mathematical Programming: Series A and B, 128*, -1-1.

Matthews, I., Ishikawa, T., & Baker, S. (2004). The template update problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*, 810–815.

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 l_1 tracker with occlusion detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1257–1264).

Pang, Y., & Ling, H. (2013). Finding the best from the second bests—Inhibiting subjective bias in evaluation of visual tracking algorithms. In *Proceedings of the IEEE International Conference on Computer Vision*.

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 Transactions on Pattern Analysis and Machine Intelligence*, *34*(11), 2233–2246.

Recht, B., Fazel, M., & Parrilo, P. A. (2010). Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, *52*, 471.

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.

Salti, S., Cavallaro, A., & Stefano, L. D. (2012). Adaptive appearance modeling for video tracking: Survey and evaluation. *IEEE Transactions on Image Processing*, *21*(10), 4334–4348.

Sevilla-Lara, L., & Learned-Miller, E. (2012). Distribution fields for tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1910–1917).

Tsaig, Y., & Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, *52*, 1289–1306.

Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., & Ma, Y. (2009). Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(2), 210–27.

Wu, Y., Lim, J., & Yang, M.-H. (2013). Online object tracking: A benchmark. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

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.

Yu, Q., Dinh, T.B., & Medioni, G. (2008). Online tracking and reacquistion using co-trained generative and discriminative trackers. In *Proceedings of European Conference on Computer Vision* (pp. 678–691).

Zhang, K., Zhang, L., & Yang, M. -H. (2012). Real-time compressive tracking. In *Proceedings of European Conference on Computer Vision*.

Zhang, T., Ghanem, B., & Ahuja, N. (2012). Robust multi-object tracking via cross-domain contextual information for sports video analysis. In *International Conference on Acoustics, Speech and Signal Processing*.

Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2012). Low-rank sparse learning for robust visual tracking. In *Proceedings of European Conference on Computer Vision*.

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*.

Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2013). Robust visual tracking via structured multi-task sparse learning. *International Journal of Computer Vision*, *101*(2), 367–383.

Zhang, T., Ghanem, B., Liu, S., Xu, C., & Ahuja, N. (2013). Low-rank sparse coding for image classification. In *Proceedings of the IEEE International Conference on Computer Vision*.

Zhang, T., Ghanem, B., Xu, C., & Ahuja, N. (2013). Object tracking by occlusion detection via structured sparse learning. In *CVsports workshop in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Zhang, T., Jia, C., Xu, C., Ma, Y., & Ahuja, N. (2014). Partial occlusion handling for visual tracking via robust part matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

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*.