

Patchwise Joint Sparse Tracking with Occlusion Detection

Ali Zarezade, Hamid R. Rabiee, *Senior Member, IEEE*, Ali Soltani-Farani, Ahmad Khajenezhad

Abstract—This paper presents a robust tracking approach to handle challenges such as occlusion and appearance change. Here, the target is partitioned into a number of patches. Then, the appearance of each patch is modeled using a dictionary composed of corresponding target patches in previous frames. In each frame, the target is found among a set of candidates generated by a particle filter, via a likelihood measure that is shown to be proportional to the sum of patch-reconstruction errors of each candidate. Since the target’s appearance often changes slowly in a video sequence, it is assumed that the target in the current frame and the best candidates of a small number of previous frames, belong to a common subspace. This is imposed using joint sparse representation to enforce the target and previous best candidates to have a common sparsity pattern. Moreover, an occlusion detection scheme is proposed that uses patch-reconstruction errors and a prior probability of occlusion, extracted from an adaptive Markov chain, to calculate the probability of occlusion per patch. In each frame, occluded patches are excluded when updating the dictionary. Extensive experimental results on several challenging sequences shows that the proposed method outperforms state-of-the-art trackers.

Index Terms—Visual tracking, particle filter, dictionary, joint sparse representation, Markov chain

I. INTRODUCTION

VISUAL TRACKING is a classic computer vision problem that remains challenging after more than three decades. There is a high demand for visual tracking in many computer vision applications, where it is directly or indirectly required to track an object in a video sequence. To name a few, consider traffic control, robotics, sports, gaming, augmented reality, and gesture recognition.

Given a bounding box defining the object of interest (target) in the first frame of a video sequence, the goal of a tracker, is to determine the object’s bounding box in subsequent frames. Primary challenges encountered in visual tracking are target appearance change and occlusion, while other challenges arise from variation in illumination, scale, and camera motion (see Table I). A general tracker is composed of three main components: 1) A *Motion model*, which generates a number of potential bounding boxes as target candidates in each frame. 2) An *Appearance model*, which models the appearance of the target throughout the video sequence. 3) An *Observation Model*, which defines a measure of similarity (likelihood) to determine which candidate is most likely to be generated by the appearance model. This candidate is then selected as the target in the current frame. The focus of this paper is on the two latter components.

Recently, sparse representation has shown appealing results in computer vision applications, such as face recognition [1], image super-resolution [2], background subtraction [3], denoising [4], and classification [5]. Motivated by the work of Wright et. al [1] in face recognition, [6] used sparse representation for object tracking, which paved the way for other sparse trackers [7]–[11]. The main idea of [6] is to model the object with a linear combination of few target images in previous frames (target templates) plus a linear combination of canonical basis vectors (trivial templates). The true candidate is distinguished from other candidates, since it is expected to be represented mostly by the target templates. Accordingly, the likelihood of a candidate is selected to be inversely proportional to its reconstruction error with the target templates. The main drawback of this method is that, in cases of severe occlusion the nonzero coefficients rarely correspond to the target template [1] and therefore the likelihood measure will no longer indicate the “true” candidate. Many other trackers [6]–[8], [12]–[17] have also taken a holistic view, in which the target is treated as a single entity. In contrast, other algorithms view the object image as a collection of small patches in order to better handle partial occlusion [9]–[11], [18]. In these methods, occlusion only affects a small subset of the patches where it occurs, and can therefore be treated more efficiently [19].

In this paper, we propose a robust tracking algorithm by taking advantage of the similarity between target objects in consecutive frames (temporal similarity assumption). Inspired by the success of patchwise methods and sparse representation for visual tracking, we consider the target object as a collection of non-overlapping patches and assume that a true candidate patch accompanied by the target patches found in previous frames, have sparse representations with a common sparsity pattern (i.e. belong to a common subspace). This is accomplished by using a regularization term that induces joint-sparsity. The appearance of each patch is modeled by a collection of corresponding target patches in previous frames, called a patch template. To increase the discriminative power of the algorithm, we model the object appearance using a dictionary composed of all the patch templates. This enables us to measure the overall likelihood of a candidate as a sum of patchwise reconstruction errors. The advantage of this likelihood measure is that in cases of partial occlusion, as long as there are patches which are not totally occluded, it will indicate the best candidate. Another advantage of building the appearance model using patch templates, is that we are able to take advantage of the patchwise reconstruction errors to determine occluded patches, and exclude them from the

A. Zarezade, H. R. Rabiee, A. Soltani-Farani, and A. Khajenezhad are with the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.

dictionary update procedure. To summarize, the contributions of this work are as follows:

- Introduction of a patchwise likelihood measure computed as the sum of patch-reconstruction errors, hence increasing tracker robustness to misalignment and partial occlusion.
- Taking into account the temporal similarity assumption through joint sparse representation, and therefore increasing tracker robustness to appearance change.
- Removing occluded patches from the dictionary update procedure, using patchwise occlusion detection based on an adaptive Markov model.

The remainder of this paper is organized as follows. Section II discusses prior work by categorizing tracking algorithms into generative and discriminative classes, and provides a more detailed description of methods that exploit sparse representation. In Section III, the motion model used in our algorithm, which is based on particle filter is introduced. The proposed tracker is introduced in Section IV. Experimental results are presented in Section V, and the concluding remarks are provided in Section VI.

II. PRIOR WORKS AND MOTIVATION

The literature on visual object tracking is extensive [20], and two major categories of visual tracking are discriminative and generative [7], [8], [10]. In this section, we briefly review the main works of each category. Afterwards, a more detailed review of significant tracking methods based on sparse representation is given. For a recent survey of sparse coding based trackers, the interested reader may refer to [21].

A. Discriminative Trackers

In discriminative methods, a classifier is trained to discriminate the target object from the background. Often, a likelihood measure is considered as the classification score. Regarding the training scheme of the classifier, the appearance model may be static [22] or adaptive [23]–[26]. In static models, the classifier is learned off-line using a training set consisting of object and background images. For example, Avidan proposed an off-line trained SVM for car tracking [22]. Although, static discriminative trackers are robust to the extent of the available training data, they can't handle previously unseen appearance variations. In adaptive approaches, the classifier is trained online during tracking and bootstraps itself with positive and negative training data sampled from the target and background. In [23]–[25], [27], [28], an ensemble of weak classifiers is learned online and combined to create a stronger classifier. An important issue that causes drift in discriminative trackers is uncertainty in labeling training data for updating the classifier. To overcome this problem, Grabner et al. proposed an online semi-boosting algorithm [29], in which labeled samples are collected only from the first frame. Babenko et al. [25], utilized the idea of multiple instance learning (MIL), in which labels are assigned to a bag of instances instead of each individual instance, to transfer the ambiguity in labeling to the learning algorithm. In [30] both ideas of MIL and semi-supervised learning are combined to take advantage of both approaches.

B. Generative Trackers

Generative trackers attempt to model the object appearance and measure the likelihood that a candidate was generated by the model. Generative models can also be static or adaptive. The simplest appearance model is a single image of the target, called a *template*. Early trackers modeled the object appearance using a fixed template and searched for the candidate with minimum sum of squared distance (SSD) to the template [31]. The mean-shift tracker [32], [33] uses an adaptive appearance model, which is the best candidate of the previous frame. The likelihood measure is defined to measure the similarity between intensity histograms of the appearance model and a candidate. Fragment-based tracker [18] is one of the successful extensions of mean-shift tracker. To handle occlusion it divides the candidate image into a collection of overlapping patches and finds the likelihood by aggregating all patches.

The main limitation of template-based appearance models is that illumination variation can't be handled efficiently. The work of Turk and Pentland [34] called Eigenfaces, addressed this limitation. Their technique employed principal component analysis (PCA) to model the target object (face). To overcome the problems caused by illumination variation, it is assumed that the target object belongs to the subspace spanned by a few of the principle components. It has been theoretically proven that an object image under Lambertian reflections belongs to a low dimensional subspace [35]. Also it has been empirically observed that even under mild pose change an object belongs to a low-dimensional subspace [1], [6], [36]. Based on these properties, Black and Jepson utilized Eigenimage and optical flow to introduce the Eigentracker [37]. Since the object appearance may change during the video sequence, the appearance model needs to be adaptive and should be updated during tracking. Ross et al. exploit eigenbasis update algorithms and proposed an incremental learning method [38] which updates eigenbases of the object subspace using incremental SVD with a forgetting factor.

C. Sparse Trackers

After the aforementioned developments in generative models, sparse representation as a variant of subspace models, was used for visual tracking leading to state-of-the-art results [6]–[12]. Generally, a candidate is represented using a linear combination of a few elements (atoms) from a dictionary composed of a number of previously found target images. The coefficients of this representation are used to find the best candidate. Apart from the ability to handle illumination and mild pose changes, these trackers attempt to tackle occlusion. Based on how the sparse representation is used, sparse trackers can also be categorized into generative and discriminative approaches.

Generative sparse approaches model appearance using a dictionary composed of two parts, one of which models the target object while the other is used to model occlusion. The likelihood measure is usually defined as a function of the reconstruction error of a candidate using coefficients corresponding to target atoms in the dictionary. As was previously mentioned, the dictionary used by the ℓ_1 tracker

[6] is composed of target (object) and trivial (occlusion) templates. Due to the large size of the trivial templates, the computational complexity of this tracker is high, and since it lacks an occlusion detection method, the dictionary may be contaminated by occlusion which causes the tracker to drift and eventually lose the target. Mei et. al addressed these two drawbacks [7] by solving the ℓ_1 minimization for a reduced set of candidates that are likely to possess a lower reconstruction error and then detecting occlusion by deriving an occlusion map from the sparse coefficients. The dictionary was updated for candidates with occlusion less than some threshold. To increase tracker accuracy, a new occlusion-aware minimization is proposed in [8], which is solved using a customized version of APG [39]. In [14], the problem of finding the sparse representation of the candidates is cast into low-rank sparse matrix learning. The sparse representation of all candidates is found simultaneously by minimizing their reconstruction error accompanied with an ℓ_1 and low-rank regularization term. To take advantage of the relationship between particles and increase tracker speed, Zhang et. al proposed to group all the candidates in a frame and find their joint-sparse representation, simultaneously [12], [15]. Since the number of candidates that don't belong to the target object's subspace is usually large, the joint-sparse minimization is likely to compensate for these candidates rather than accurately representing the fewer candidates that belong to the target object's subspace. Therefore, the assumption made by Zhang et. al that all candidates are related through a common low-dimensional subspace seems unrealistic.

Discriminative sparse trackers, employ sparse representation to discriminate the target from its background. In [17], the dictionary is learned from SIFT descriptors extracted from a general dataset of object images, and a linear classifier is learned online with positive and negative samples. To better discriminate the target from background, in [10] a dictionary is learned online from target patches. The sparse representation of candidate patches are obtained via Local Linear Coding (LLC) [40]. Finally, the best candidate is found via regularized Mean-Shift. In [13], a sparse measurement matrix is adopted to extract low dimensional discriminative features and the tracking problem is formulated as an online naive Bayes classifier. In [9], a dictionary composed of overlapping patches from the target is used to find the sparse representation of candidate patches. A method called alignment-pooling is proposed where the likelihoods are found by summation over a feature vector composed of the similarity of each candidate patch to related patches in the target dictionary.

III. PARTICLE FILTER

Visual tracking problem may be cast as Bayesian filtering. The goal of Bayesian filtering is to find the posterior pdf of a system state \mathbf{x}_t , given all observations $\mathbf{z}_{1:t}$. Bayesian filtering is composed of two steps: prediction and update. In the prediction step, with Markov assumption, the prior pdf of the state is obtained using the Chapman-Kolmogorov equation as:

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

When a new observation \mathbf{z}_t becomes available, in update step, the posterior is found via Bayes' rule as:

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

Solving (1) and (2) gives the optimal solution of the filtering problem. Particle filter represents an analytical method for solving these equations. The key idea is to represent the posterior by samples (particles) and their associated probabilities (weights). Therefore, updating the posterior is equivalent to updating the particles and their weights. So far, many variants of particle filters have been developed [41]. We use Sequential Importance Resampling (SIR) filter [42] in which particles are resampled in each frame to reduce the degeneracy problem. The update equations of SIR filter are:

$$\mathbf{x}_t^i \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^i), \mathbf{w}_t^i \sim p(\mathbf{z}_t|\mathbf{x}_t^i) \quad (3)$$

where \mathbf{x}_t^i is the i th particle in frame t . In visual tracking, state \mathbf{x}_t denotes the affine motion parameters which define the object region in frame t , and observations $\mathbf{z}_{1:t}$ are a collection of frames observed so far. We notice from SIR equations (3) that visual tracking requires a likelihood evaluation measure $p(\mathbf{z}_t|\mathbf{x}_t^i)$ and the state transition probability $p(\mathbf{x}_t|\mathbf{x}_{t-1}^i)$. Without any prior knowledge, $p(\mathbf{x}_t|\mathbf{x}_{t-1}^i)$ is considered to be a zero mean Gaussian with predefined variance Σ . To evaluate $p(\mathbf{x}_t|\mathbf{x}_{t-1}^i)$, first the candidate region associated with particle \mathbf{x}_{t-1}^i is cropped from frame t , then the probability that it is generated by the appearance model is considered as its likelihood.

IV. PROPOSED METHOD

Given the initial state of the target, the goal is to find the target object among the set of candidates generated by the particle filter, in subsequent frames. In order to accomplish this task, we need to model the target's appearance and find the candidate that is most likely to be generated by this model. We use a dictionary composed of image patches selected from targets found in previous frames to model the target's appearance. The best candidate is assumed to have the least reconstruction error sum, over its patches. In order to support changes in the appearance of the target object, once the best candidate is found, the model needs to be updated accordingly. We attempt to exclude occluded image patches when updating the dictionary with the best candidate. In this section, we describe these steps in detail and intuitively justify the proposed approach.

A. Notation

In the sequel, bold lower case and bold upper case letters are used for vectors (\mathbf{d}) and matrices (\mathbf{D}), respectively. For any variable, the superscript (i) is used to indicate that the variable is related to the i th patch. Suppose that $\Lambda \subset \{1, 2, \dots, n\}$ is a subset of indices, the complement of which is denoted by $\Lambda^c = \{1, 2, \dots, n\} \setminus \Lambda$. By \mathbf{d}_Λ we mean the vector obtained from \mathbf{d} by eliminating indices inside Λ^c . Similarly, by \mathbf{D}_Λ we mean the matrix obtained by removing from \mathbf{D} the columns indexed by Λ^c .

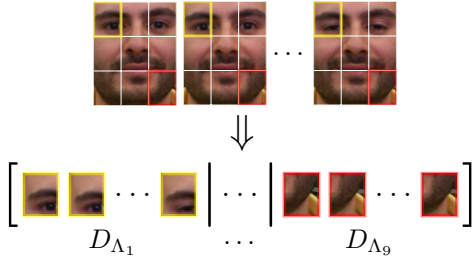


Fig. 1. **Appearance Dictionary.** Target appearance is modeled by a dictionary composed of patch templates. Each patch template D_{Λ_i} is used to model the corresponding target patch, and is composed of a number of target patches found in previous frames.

B. Appearance Model

Given a dictionary composed of a number of the best candidates found in previous frames, the target's appearance is modeled with a linear combination of a few elements (atoms) from this dictionary. In other words, it is assumed that the target belongs to a low-dimensional linear subspace spanned by a few of the dictionary atoms. Linear subspaces are rich appearance models that can handle illumination and pose change. It has been theoretically proven that an object under different illumination lies on a low-dimensional subspace [35]. Also, [36] empirically showed that a dictionary of pose varying face images is able to model pose change. We take a patchwise view and model the target's appearance with a dictionary D , composed of patches selected from n previous best candidates (see Fig. 1):

$$D = \left[\overbrace{\mathbf{d}_1^{(1)} \dots \mathbf{d}_n^{(1)}}^{D_{\Lambda_1}} \mid \dots \mid \overbrace{\mathbf{d}_1^{(m)} \dots \mathbf{d}_n^{(m)}}^{D_{\Lambda_m}} \right] \quad (4)$$

where $\mathbf{d}_i^{(j)} \in \mathbb{R}^M$ is a vectorized grayscale image of the j th patch of the i th target placed in the dictionary, Λ_j denotes the set of indices of the atoms that belong to the j th patch in all targets, D_{Λ_j} is the j th patch template, and $D \in \mathbb{R}^{M \times N}$. The number of dictionary atoms is $N = nm$, where n and m are the number of targets and number of patches in each target image, respectively.

Before further discussing the model, we look into two issues regarding the proposed appearance model, namely feature extraction and patchwise treatment of the target. Among features used by sparse trackers are grayscale [6], [9], [10], SIFT [17], block-division based covariance [43], and covariance matrix [44]. Most feature extraction methods involve only linear (or approximately linear) operations, and it has been shown that linear transform of grayscale features under mild conditions has no considerable effect on face recognition using sparse representation [1]. On the other hand, feature extraction methods such as SIFT or Haar [25], [28], [29] are usually accompanied by information loss, which reduce the discriminative ability of the tracker. Considering these observations and to reduce computational complexity, we use grayscale features. The other major issue is the target representation scheme. The common approach is to view the target as a single entity, in which the dictionary is a collection of vectorized images of

the target in previous frames, in contrast to the dictionary in (4) which is composed of vectorized patches. Wright et. al [1] showed that severe occlusions cause a considerable decrease in face recognition performance. Also, the results of a recent benchmark [19] shows that trackers with a patchwise view [9], [10], [18] are more robust to occlusion than methods with a holistic view [6], [8], [12]. The advantages of the proposed target appearance model are twofold: improved performance in handling occlusion and considerable speed up due to reduced dictionary size compared to [6].

In the current frame, the i th patch of the target \mathbf{y} , denoted by $\mathbf{y}^{(i)}$, is assumed to belong to a low-dimensional subspace spanned by the columns of the i th patch template D_{Λ_i} . Considering the appearance variations of the target and presence of noise, each patch template is expected to be a full-rank matrix. Since different patch templates D_{Λ_j} correspond to different non-overlapping patches of the target, and each patch template is likely to be full-rank, it is reasonable to assume that the collection of patch templates $D_{\Lambda_j}, j \neq i$, denoted by $D_{\Lambda_i^c}$ is likely to be overcomplete. Hence, any occlusion or noise in $\mathbf{y}^{(i)}$ may be modeled as $e^{(i)} = D_{\Lambda_i^c} \mathbf{c}_{\Lambda_i^c}^{(i)}$ leading to our representation of $\mathbf{y}^{(i)}$ as:

$$\mathbf{y}^{(i)} = D_{\Lambda_i} \mathbf{c}_{\Lambda_i}^{(i)} + e^{(i)} = D_{\Lambda_i} \mathbf{c}_{\Lambda_i}^{(i)} + D_{\Lambda_i^c} \mathbf{c}_{\Lambda_i^c}^{(i)} = D \mathbf{c}^{(i)} \quad (5)$$

where $\mathbf{c}^{(i)}$ is the vector of representation coefficients. As mentioned before, the appearance model dictionary D is expected to be overcomplete. Therefore, the system of linear equations (5) is underdetermined, and has no unique solution for $\mathbf{c}^{(i)}$. When occlusion and noise is negligible, we can expect $\mathbf{c}^{(i)}$ to be sparse. This sparsity may be imposed by solving:

$$\underset{\mathbf{c}^{(i)}}{\operatorname{argmin}} \|\mathbf{c}^{(i)}\|_0 \quad \text{s.t.} \quad \mathbf{y}^{(i)} = D \mathbf{c}^{(i)} \quad (6)$$

Taking into account any shortcomings due to the linear nature of the model, a relaxed form [45] is usually solved by using at most L atoms:

$$\underset{\mathbf{c}^{(i)}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{y}^{(i)} - D \mathbf{c}^{(i)}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{c}^{(i)}\|_0 \leq L \quad (7)$$

Considering that the target's appearance changes smoothly, it is plausible to assume that the target objects, over a few consecutive frames, belong to the same subspace. We apply this assumption by enforcing the sparse representation of the current target patch $\mathbf{y}^{(i)}$ to be jointly sparse (i.e. have the same sparsity pattern) with the same patch in the previous k best candidates, i.e. $\mathbf{y}_{t-k}^{*(i)}, \dots, \mathbf{y}_{t-1}^{*(i)}$. To this end, we solve the following optimization problem:

$$\underset{\mathbf{C}^{(i)}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{Y}^{(i)} - D \mathbf{C}^{(i)}\|_F^2 \quad \text{s.t.} \quad \|\mathbf{C}^{(i)}\|_{2,0} \leq L \quad (8)$$

where $\mathbf{Y}^{(i)} = [\mathbf{y}_{t-k}^{*(i)}, \dots, \mathbf{y}_{t-1}^{*(i)}, \mathbf{y}^{(i)}]$, the columns of $\mathbf{C}^{(i)}$ are the corresponding sparse representations, and $\|\mathbf{C}^{(i)}\|_{2,0}$ is the ℓ_0 pseudo-norm computed over the ℓ_2 norms of rows of $\mathbf{C}^{(i)}$, i.e. counts the number of nonzero rows of $\mathbf{C}^{(i)}$ (see Fig. 2).

This optimization problem belongs to the class of NP-Hard problems [46], and is therefore usually solved approximately, by using a greedy algorithm or convex relaxation. Different

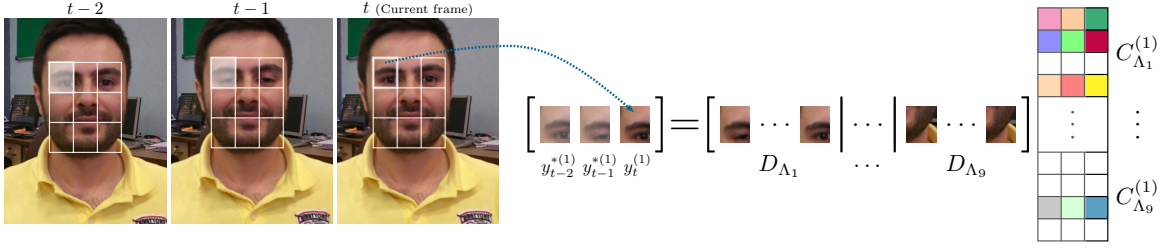


Fig. 2. **Joint sparse representation of a patch.** To find the sparse representation of a candidate patch, it is grouped with corresponding patches from the two previous best candidates. The joint-sparse minimizations (8) or (9), force the sparse coding of the signals in a group to have similar support, as depicted in the row-sparse matrix $C^{(1)}$.

convex relaxations have been proposed for this problem which often take the form of:

$$\arg \min_{C^{(i)}} \frac{1}{2} \|\mathbf{Y}^{(i)} - \mathbf{D}C^{(i)}\|_F^2 + \lambda \|C^{(i)}\|_{p,q} \quad (9)$$

where $\|C^{(i)}\|_{p,q}$ is the ℓ_q norm computed over the ℓ_p norms of rows of $C^{(i)}$, and λ is a regularization parameter, which balances reconstruction error with model complexity. For (9) to become a convex problem, we require $p, q \geq 1$. This is the well known convex formulation of the joint sparse recovery [47] or simultaneous sparse approximation [48] problem, also known as the Multiple Measurement Vector (MMV) [49] problem in the compressed sensing community. Other convex formulations for this problem follow the general ℓ_q/ℓ_p form with $q \geq 1$ and $p = 1$, among which ℓ_2/ℓ_1 and ℓ_∞/ℓ_1 formulations are more widely used. We adhere to the ℓ_2/ℓ_1 form because the objective function of (9) is strongly convex for $k > 0$, and therefore has a unique solution. Several algorithms exist that efficiently solve this problem [49]–[52].

A common greedy strategy used to solve (8) is the Simultaneous Orthogonal Matching Pursuit (SOMP) algorithm [53] which is an extension of the well known Orthogonal Matching Pursuit (OMP) algorithm. In SOMP, it is initially assumed that $C^{(i)} = 0$, i.e. no atoms are used in the representation. Then in each iteration, the remainder $\mathbf{R} = \mathbf{Y}^{(i)} - \mathbf{D}C^{(i)}$ is computed, and the dictionary atom \mathbf{d}_j that has the largest correlation with \mathbf{R} , i.e. $\arg \max_j \|\mathbf{R}^T \mathbf{d}_j\|_2$, is added to the set of active atoms in the representation. The sparse representation, $C^{(i)}$ is then updated to use the set of active atoms.

Among the algorithms proposed to solve (9), Malioutov, et al. [50] show that the optimization may be posed as Second Order Cone Programming (SOCP) for which off-the-shelf optimizers are available. The inner loops of SOCP are computationally expensive and the algorithm is only suitable for small-size problems [50], [51]. The work of Lu, et al. [51] extends the Alternating Directions Method (ADM) of [54] to solve the MMV recovery. Although the proposed algorithm is quite fast within an acceptable solution accuracy, there is no guarantee that each iteration of the algorithm will reduce the objective function. We employ the regularized M-FOCUSS algorithm of Cotter et al. [49] which is simple, efficient, and also guaranteed to reduce the objective function in each iteration. The algorithm works by estimating the ℓ_2 norm of each row of $C^{(i)}$, and then updating $C^{(i)}$ based on that

estimate. The update rule may be written as:

$$C^{(i)} = \Lambda \mathbf{D}^T (\mathbf{D} \Lambda \mathbf{D}^T + \lambda \mathbf{I})^{-1} \mathbf{Y}^{(i)} \quad (10)$$

where $\Lambda = \text{diag}(\|C_j^{(i)}\|_2)$ is computed using the previous estimate of $C^{(i)}$. The algorithm may be initialized from any random point for which all rows of $C^{(i)}$ have a nonzero norm, and is terminated when the difference between consecutive estimates of $C^{(i)}$ is smaller than some threshold.

C. Observation Model

The model defined in (5) represents both the target object and any occlusion or noise, through the sparse coefficients. For a given sparse representation $\mathbf{c}^{(i)}$ of patch i , $\mathbf{c}_{\Lambda_i}^{(i)}$ are the coefficients corresponding to the patch template i , and represent the target, while $\mathbf{c}_{\Lambda_c}^{(i)}$ represents any occlusion or noise. In order to measure the likelihood that a given candidate is generated by the target model, we only pay attention to the portion of sparse coefficients that represent the target, and disregard the representation of the occlusion which we later use for occlusion detection in Section IV-D.

Given the current frame image \mathbf{z} , and a region defined by the state \mathbf{x} which is provided by the motion model as a candidate for the target, we define the likelihood of the candidate image \mathbf{y} as the probability that it is generated by the appearance model dictionary \mathbf{D} through the portion of sparse coefficients that represent the target found by minimization (8) or (9). Hence we define:

$$p(\mathbf{z}|\mathbf{x}) \triangleq p(\mathbf{y}|\mathbf{D}, \mathbf{c}_{\Lambda_1}^{(1)}, \dots, \mathbf{c}_{\Lambda_m}^{(m)}) \quad (11)$$

Assuming independence between patches of a candidate, we can evaluate a candidate's likelihood as:

$$p(\mathbf{y}|\mathbf{D}, \mathbf{c}_{\Lambda_1}^{(1)}, \dots, \mathbf{c}_{\Lambda_m}^{(m)}) = \prod_i p(\mathbf{y}^{(i)}|\mathbf{D}_{\Lambda_i}, \mathbf{c}_{\Lambda_i}^{(i)}) \quad (12)$$

Considering a zero mean Gaussian error $\mathbf{e}^{(i)} \sim N(0, \sigma^{(i)}\mathbf{I})$ in equation (5), the likelihood of each patch is obtained from:

$$p(\mathbf{y}^{(i)}|\mathbf{D}_{\Lambda_i}, \mathbf{c}_{\Lambda_i}^{(i)}) = \exp\left(-\frac{\|\mathbf{y}^{(i)} - \mathbf{D}_{\Lambda_i} \mathbf{c}_{\Lambda_i}^{(i)}\|_2^2}{\sigma^{(i)2}}\right) \quad (13)$$

Taking the log-likelihood of (11), using (12) and (13), and assuming equal variance for all patches, we have the final measure for selecting the best candidate:

$$\log p(\mathbf{z}|\mathbf{x}) \propto -\sum_i \|\mathbf{y}^{(i)} - \mathbf{D}_{\Lambda_i} \mathbf{c}_{\Lambda_i}^{(i)}\|_2^2 \quad (14)$$

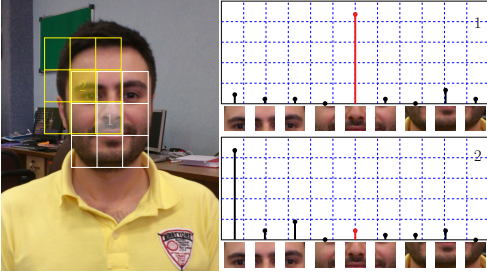


Fig. 3. **Misalignment effect.** Illustrative sparse representation of patch 1 (highlighted in white) and patch 2 (highlighted in yellow). The center patch of the white candidate is aligned with the target and has low reconstruction error in its corresponding patch template. The center patch in the yellow candidate is slightly shifted away from the target and is mainly reconstructed by a non-corresponding patch template, hence increasing its reconstruction error in the corresponding patch template.

Intuitively, this log-likelihood measure will favor the candidate with the lowest sum of reconstruction errors calculated for each patch separately, via its representation in the corresponding patch template. The main benefit of this new likelihood measure is due to patchwise treatment of the candidates. When a partial or even a severe occlusion occurs, as long as there are some patches which are not totally occluded, this measure will indicate the true best candidate. A pseudocode for the proposed tracker is presented in the Algorithm 1.

We expect to get better results by using some heuristic to select appropriate values for $\sigma^{(i)}$. As the target can not usually be inscribed inside a rectangle, marginal patches often contain the background. Moreover, occlusion usually occurs at the boundaries of the target, and therefore marginal patches are more affected. To alleviate this undesirable effect we can set the values of $\sigma^{(i)}$ s proportional to the inverse distance of the patch center from the center of the rectangle enclosing the target. In the experiments of Section V, we have used the simple case with equal variance for all patches.

Calculating the negative log-likelihood of a candidate as the sum of its patch-reconstruction errors, increases the robustness of the tracker to misalignments. When a candidate is slightly shifted away from the target, the reconstruction error of its patches will substantially increase. As depicted in Fig. 3, this is because the shifted candidate is no longer aligned with the target, hence the sparse representation of any of its patches is less likely to use atoms from its corresponding patch template. Since the reconstruction error is computed using the coefficients of the corresponding patch template, the log-likelihood of a misaligned candidate will be quite small.

Since the target’s appearance often changes slowly, obtaining the sparse representation of a candidate patch, jointly with its corresponding patches from the best candidates of previous frames, increases the robustness of the tracker to “false” candidates. Grouping a “true” candidate patch with corresponding patches from the best previous candidates is expected to increase its chance of being represented by atoms from the correct patch template even in case of a moderate pose change, hence increasing its likelihood. Our intuition is that the joint sparse nature of the solution of minimization (9) is likely to tolerate an increase in the reconstruction error of

the “true” patch, in order to best represent the whole group of patches.

To illustrate the above discussions, we have depicted in Fig. 4, the likelihood measure for a set of regularly sampled candidates centered inside a rectangular region around the target, for various methods. As expected, the likelihood peaks at the center of the region, but more steeply for the proposed method. In other words, our likelihood measure has less variance compared to the other methods.

Algorithm 1 Proposed tracker

Input: Previous particles $\mathcal{X}_{t-1} = \{\mathbf{x}_{t-1}^i\}_{i=1}^N$, frame \mathbf{z}_t , dictionary \mathbf{D} , previous k targets $\{\mathbf{y}_{t-1}^*, \dots, \mathbf{y}_{t-k}^*\}$.

Output: Target \mathbf{y} , current particles $\mathcal{X}_t = \{\mathbf{x}_t^i\}_{i=1}^N$, dictionary \mathbf{D} .

- 1: Draw particles \mathcal{X}_t from \mathcal{X}_{t-1} using $\mathbf{x}_t^i \sim \mathcal{N}(\mathbf{x}_{t-1}^i, \Sigma)$.
 - 2: **for** $i = 1$ to N **do**
 - 3: Generate candidate \mathbf{y} corresponding to particle \mathbf{x}_t^i and resize it to a predefined size.
 - 4: Partition candidate \mathbf{y} in to equal-sized non-overlapping patches $\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \dots, \mathbf{y}^{(m)}\}$.
 - 5: **for** $j = 1$ to m **do**
 - 6: $\mathbf{Y}^{(j)} \leftarrow [\mathbf{y}_{t-k}^{*(j)}, \dots, \mathbf{y}_{t-1}^{*(j)}, \mathbf{y}^{(j)}]$
 - 7: $\mathbf{C}^{(j)} \leftarrow$ solution of minimization (8) or (9)
 - 8: $\mathbf{c}^{(j)} \leftarrow \mathbf{C}_{k+1}^{(j)}$
 - 9: $p(\mathbf{y}^{(j)} | \mathbf{D}_{\Lambda_i}, \mathbf{c}_{\Lambda_i}^{(j)}) \leftarrow \exp(-\|\mathbf{y}^{(j)} - \mathbf{D}_{\Lambda_j} \mathbf{c}_{\Lambda_j}^{(j)}\|_2^2)$
 - 10: **end for**
 - 11: $\log p(\mathbf{z}_t | \mathbf{x}_t^i) \leftarrow \sum_j \log p(\mathbf{y}^{(j)} | \mathbf{D}_{\Lambda_j}, \mathbf{c}_{\Lambda_j}^{(j)})$
 - 12: **end for**
 - 13: Select the candidate with the largest $\log p(\mathbf{z}_t | \mathbf{x}_t^i)$ as the target \mathbf{y} in current frame.
 - 14: Update dictionary \mathbf{D} with \mathbf{y} using Algorithm 2.
-

D. Dictionary Update

The appearance model needs to be updated over time in order to handle variations in the target’s pose and illumination. Commonly, the best candidate found in the current frame is used to update the appearance model. Many schemes have been presented to update generative and discriminative trackers. In most generative sparse trackers, the target is modeled with a dictionary and is updated by replacing its atoms. Two issues are important when updating the dictionary. The first issue is the policy used to select the dictionary atom(s) that are going to be replaced, while the second issue is related to detecting and perhaps removing any occlusion present in the new atom(s). If severely occluded atoms are allowed to enter, and hence corrupt the dictionary, the tracker is prone to produce erroneous results leading to further corruption of the dictionary and gradual failure of the tracker. This is known as the problem of drift. Therefore, an accurate occlusion detection approach is essential to alleviate this problem.

To select an old atom to be replaced with a newly found target, the policy used by [6]–[8], [12], [15] is based on assigning weights to the dictionary atoms. The weight assigned to each atom is proportional to its corresponding coefficient in the sparse representation of the best candidate. The weights are updated in each frame, after the best candidate is found. The atom with the lowest weight is selected to be replaced by the target. To alleviate the drift problem, similar to [9], we randomly select dictionary atoms according to a predefined

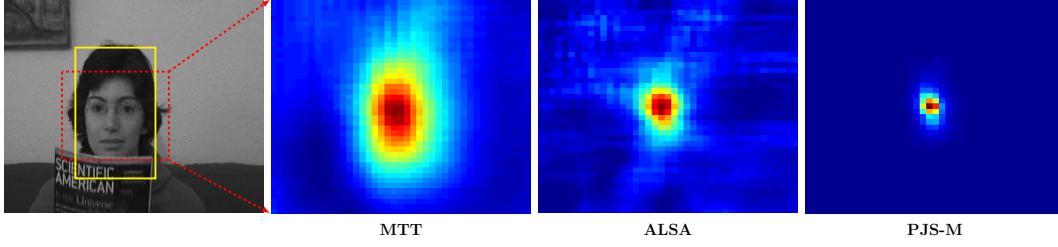


Fig. 4. **Likelihood measure.** Comparison of likelihood maps obtained by MTT [15], ALSA [9], and proposed tracker. These maps show the likelihood of candidates generated by sweeping the ground truth bounding box (yellow rectangle) inside the red rectangular region.

distribution. This distribution is defined so that recent atoms are more likely to be removed.

Based on our patchwise approach, we propose a new occlusion detection scheme. We investigate whether each patch is occluded or not by defining an occlusion probability. A patch is assumed to be occluded if its occlusion probability is larger than $\frac{1}{2}$. Then the occluded patches are excluded from the best candidates, when updating the dictionary. The occlusion probability of a patch $\mathbf{y}^{(i)}$, is defined as:

$$p(o^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, \mathbf{y}^{(i)}, \mathbf{c}^{(i)}) \propto \frac{p(\mathbf{y}^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, o^{(i)}, \mathbf{c}^{(i)})p(o^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, \mathbf{c}^{(i)})}{p(\mathbf{y}^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, \mathbf{c}^{(i)})} \quad (15)$$

where $o^{(i)} = 1$, and $o^{(i)} = 0$ indicate the occluded and non-occluded states, respectively, and $\mathcal{O}^{(i)} = \{o_1^{(i)}, \dots, o_{t-1}^{(i)}\}$ is the history of occlusion states for patch i . Assuming that given the current occlusion state of $\mathbf{y}^{(i)}$, this patch is independent of previous occlusion occurrences, and considering a first order Markov model for occlusion state of patches, we have:

$$p(\mathbf{y}^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, o^{(i)}, \mathbf{c}^{(i)}) = p(\mathbf{y}^{(i)}|\mathbf{D}, o^{(i)}, \mathbf{c}^{(i)}) \quad (16)$$

$$p(o^{(i)}|\mathbf{D}, \mathcal{O}^{(i)}, \mathbf{c}^{(i)}) = p(o^{(i)}|o_{t-1}^{(i)}) \quad (17)$$

To determine the probability of occlusion in (15), it suffices to define the above two distributions. Considering the model in (5), the target patch is represented by the sparse coefficients $\mathbf{c}_{\Lambda_i}^{(i)}$, while severely occluded patches are expected to be represented mainly by $\mathbf{c}_{\Lambda_i^c}^{(i)}$. Since the best candidate is already found, we no longer need to impose the temporal similarity assumption, and \mathbf{c}_{Λ_i} is obtained from (9) with $k = 0$. Hence, to discriminate an occluded patch from a non-occluded patch we define the likelihood of patch i in (16) for the case of $o^{(i)} = 0$ and $o^{(i)} = 1$, respectively:

$$p(\mathbf{y}^{(i)}|\mathbf{D}, \mathbf{c}^{(i)}, o^{(i)} = 0) = \exp(-\|\mathbf{y}^{(i)} - \mathbf{D}_{\Lambda_i} \mathbf{c}_{\Lambda_i}^{(i)}\|_2^2) \quad (18)$$

$$p(\mathbf{y}^{(i)}|\mathbf{D}, \mathbf{c}^{(i)}, o^{(i)} = 1) = \exp(-\|\mathbf{y}^{(i)} - \mathbf{D}_{\Lambda_i^c} \mathbf{c}_{\Lambda_i^c}^{(i)}\|_2^2) \quad (19)$$

To define a prior for the occlusion state of patch i in (17), we consider a simple two state Markov chain for each patch, as shown in Fig 5. The transition probabilities of this model are updated online during tracking, by using Maximum A Posteriori (MAP) estimation. Removing the patch superscript (i) from occlusion states $o^{(i)}$ and restoring the frame subscript for notational convenience, we have:

$$p(o_t|o_{t-1}) = \mu^{o_{t-1}(1-o_t)}(1-\mu)^{o_{t-1}o_t} \eta^{(1-o_{t-1})o_t}(1-\eta)^{(1-o_{t-1})(1-o_t)} \quad (20)$$

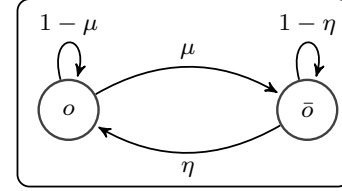


Fig. 5. **Markov chain of occlusion state.** Two-state Markov chain with adaptive transition probabilities used to define occlusion prior of each patch. States o and \bar{o} represent occluded and non-occluded states, respectively.

Since the transition probabilities are between 0 and 1, an appropriate prior for these parameters is the beta distribution, namely $\mu \sim \beta(a, b)$, and $\eta \sim \beta(c, d)$. By changing the parameters of these distributions we can fully reflect any prior knowledge about the transition probabilities that determine patch occlusion. With knowledge of occlusion states in previous frames and prior distribution of the transition probabilities, μ, η can be found using MAP estimation as follows:

$$\arg \max_{\mu, \eta} p(\mu, \eta | o_1, o_2, \dots, o_t) = \quad (21)$$

$$\arg \max_{\mu, \eta} \beta(\mu|a, b)\beta(\eta|c, d) \prod_{i=1}^t p(o_i|o_{i-1}, \mu, \eta)$$

Substituting (20) and solving this minimization, the MAP estimation of μ and η can be found as:

$$\hat{\mu}_{\text{MAP}} = \frac{a - 1 + n_{o\bar{o}}}{a - 1 + n_{o\bar{o}} + b - 1 + n_{oo}} \quad (22)$$

$$\hat{\eta}_{\text{MAP}} = \frac{c - 1 + n_{\bar{o}o}}{c - 1 + n_{\bar{o}o} + d - 1 + n_{\bar{o}\bar{o}}} \quad (23)$$

where for example $n_{o\bar{o}}$ counts the number of transitions between occlusion state $o_{i-1} = 1$, and non-occluded state $o_i = 0$ in the occlusion history \mathcal{O} . A pseudo-code of the dictionary update procedure is represented in Algorithm 2.

V. EXPERIMENTS

In this section, we present qualitative and quantitative experimental results to evaluate the performance of the proposed trackers. Qualitative comparisons involve demonstrating each tracker's performance on a number of sample frames. Since this measure is limited to observing performance for a few hand picked frames, it is essential to evaluate different tracking algorithms with quantitative measures such as Center Location Error (CLE), PASCAL overlap score (VOC), and success rate. The experimental results are organized as follows. In Section

Algorithm 2 Dictionary update

Input: Dictionary D , newly found target image \mathbf{y} , state transition counters, occlusion history of patches $\{\mathcal{O}^{(i)}\}_{i=1}^m$.

Output: Updated dictionary D , updated state transition counters, updated occlusion history of patches $\{\mathcal{O}^{(i)}\}_{i=1}^m$.

- 1: Randomly select an old dictionary target image \mathbf{y}_{old} , with higher probability in selection of recent targets.
 - 2: **for** $i = 1$ to m **do**
 - 3: Find sparse representation of the target patch:
 $\mathbf{c}^{(i)} \leftarrow \arg \min_{\mathbf{c}} \frac{1}{2} \|\mathbf{y}^{(i)} - D\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_1$
 - 4: Find occlusion likelihood of the target patch according to (18) and (19).
 - 5: Update the transition probabilities of Markov chain according to (22) and (23).
 - 6: Evaluate the prior probability of occlusion according to (20).
 - 7: Calculate occlusion probability according to (15).
 - 8: Update dictionary with non-occluded patches of the target:
 - 9: **if** $p(o^{(i)} = 0 | D, \mathcal{O}^{(i)}, \mathbf{y}^{(i)}, \mathbf{c}^{(i)}) \geq 0.5$ **then**
 - 10: $\mathbf{y}_{\text{old}}^{(i)} \leftarrow \mathbf{y}^{(i)}, o^{(i)} \leftarrow 0$
 - 11: **else**
 - 12: $o^{(i)} \leftarrow 1$
 - 13: **end if**
 - 14: Update counters using occlusion states $o^{(i)}$ and $o_{t-1}^{(i)}$.
 - 15: $\mathcal{O}^{(i)} \leftarrow \mathcal{O}^{(i)} \cup \{o^{(i)}\}$
 - 16: **end for**
-

V-A an overview of the different datasets used for experiments, and the trackers used for comparison, is provided. Some important implementation details are discussed in Section V-B. The comparison of different trackers according to qualitative and quantitative measures are presented in Sections V-C and V-D, respectively.

A. Base Trackers and Dataset

For evaluation we have selected a set of 10 publicly available datasets that cover the common challenges prevalent in visual tracking. The datasets are commonly known as *board*, *crossing*, *david*, *dollar*, *faceocc2*, *skating1*, *stone*, *sylv*, *trellis*, and *walking*, that are available along with their ground truth at [19]. A list of the challenges posed by these datasets is provided in Table I. We have annotated the datasets according to these challenges as listed in Table II, followed by their different attributes, namely, target size, resolution, and number of frames. A histogram that shows the number of datasets that pose each challenge is shown in Fig. 6. This histogram approximately has an uniform distribution, indicating the diversity of the selected datasets.

We compare our results with four well known trackers, namely OAB [24], MIL [25], Frag [18], and IVT [38]. We also report results of two recent sparse trackers, denoted as APG [8], and MTT [12]. For these trackers, we have used the source codes either available from the corresponding author’s website or provided by the tracking benchmark [19]. The proposed trackers are denoted by PJS-M¹ and PJS-S². The two trackers differ in that PJS-M solves the convex joint sparse minimization of (9), while PJS-S solves the non-convex formulation of (8). We use the implementation of

¹Patchwise Joint Sparse-M-FOCCUS

²Patchwise Joint Sparse-SOMP

TABLE I
LIST OF CHALLENGES USED AS ATTRIBUTES FOR DATASET ANNOTATION IN TABLE II.

Attribute	Description	Attribute	Description
IV	Illumination variation	PO	Partial occlusion
SV	Scale variation	SO	Sever occlusion
PC	Pose change	CM	Camera motion
IR	In-plane rotation	MB	Motion blur
OR	Out-of-plane rotation	ST	Similar targets

TABLE II
DATASET CHARACTERISTICS: RESOLUTION, LENGTH, TARGET SIZE, AND CHALLENGES LISTED IN TABLE I

Dataset	Challenges	Target size	Resolution	Length
board	SV,IR,OR,FM,MB	195 × 153	640 × 480	696
david	IV,SV,PC,OR,PO,MB	68 × 122	320 × 240	460
dollar	PC,ST	62 × 98	320 × 240	325
faceocc2	IR,PC,PO	82 × 98	320 × 240	594
skating1	IV,SV,PC,OR,SO,ST,MB	35 × 100	640 × 360	400
stone	SO,ST	41 × 20	320 × 240	591
sylv	IV,PC,IR,OR	51 × 61	320 × 240	800
trellis	IV,SV,PC,IR,OR,CM	68 × 101	320 × 240	569
walking2	SV,PC,PO,SO,ST	31 × 115	384 × 288	500

SOMP available by the SPAMS [55], [56] package to solve (8), and our implementation of regularized M-FOCCUS for (9). All source codes for our experiments are available at <http://ssp.dml.ir/research/pjs>.

B. Implementation Details

For methods that use pixel intensities as features (IVT, APG, MTT, PJS-M, and PJS-S) the target object was resized to 32 × 32 pixels. For OAB, MIL, and Frag that extract features from raw images, the parameters were set as provided by the authors. The patch size was set to 8 × 8 pixels, resulting in 16 non-overlapping patches for our method. Hence, the dictionary is composed of 16 patch templates each of which contains 10 vectorized grayscale patches. The dictionary is initialized by patches from the given target object and nine slightly (1 to 2 pixels) shifted versions. The nine shifted versions are replaced by the found target in the next nine frames, after which the dictionary is updated according to the scheme described in Section IV-D. The dictionary update parameters of (22) and (23) were set equal to $a = d = 4$, and $b = c = 8$. For the joint sparse recovery minimizations of (8) and (9) we use $L = 4$, $\gamma = 0.001$, and $k = 4$. For methods that require a particle filter motion model, the number of particles was set to 600 and their variance was set to $\sigma = \text{diag}([6, 6, 0.02, 0.002, 0.002, 0])$. The first two diagonal elements of σ correspond to spatial variance in horizontal and vertical directions, and the others correspond to rotation, scale, and skew. These parameters were fixed across all datasets and trackers.

Motion models, especially those which are based on a particle filter, usually involve randomness. To fairly evaluate the trackers and reduce the effects of randomness, we run each experiment 10 times and report the average results. In order to provide reproducible results, the random seed for the 10 runs was set between 0 to 9 in MATLAB.

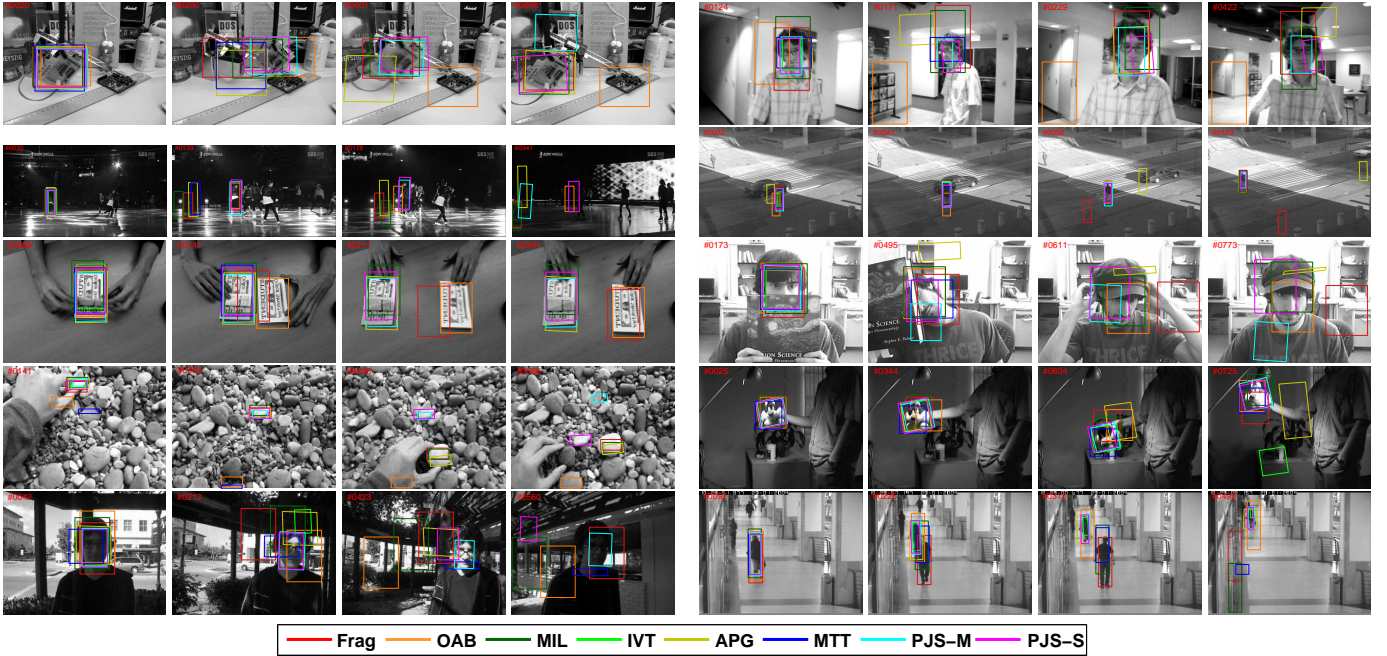


Fig. 7. **Sample frame illustration.** Tracking results of different trackers on 10 datasets: board, david, skating1, crossing, dollar, faceocc2, stone, sylv, trellis, and walking2, from left to right and top to bottom respectively.

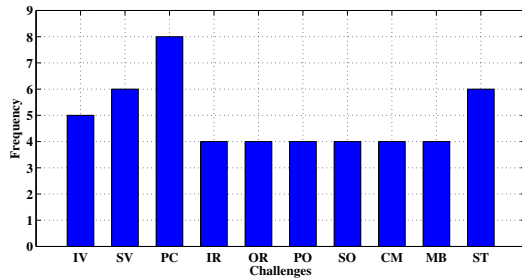


Fig. 6. **Challenge histogram.** Frequency of challenges listed in Table I.

C. Qualitative Comparison

Figure 7 shows sample frames and the bounding boxes found by each tracker in each dataset. The illustrated results show that the proposed trackers have been successful in challenging scenarios where previous trackers have often failed. To better reflect on the robustness of the trackers, the next subsection is dedicated to quantitative comparisons. In this section we qualitatively describe how the trackers have dealt with some of the challenges posed by the selected datasets.

1) **board**: In this video, an electronic board undergoes different visual changes such as scale variation, motion blur, and in-plane and out-of-plane rotation. This is quite a challenging dataset, and all trackers except PJS-S and PJS-M have lost track of the target prior to frame 100. In frame 400, the target passes its previous location, at which point Frag and MIL were able to track the target again, but lost it again after 60 or so frames. Only PJS-S was able to track the target throughout the sequence.

2) **david**: This dataset shows a young man’s face as he and the camera move, resulting in challenges such as pose change, illumination variation, partial occlusion, and out-of-

plane rotation. Prior to frame 124, which has illumination variation and minor pose change, only OAB drifted from the target. In subsequent frames, before the target rotates about 90° , APG, Frag, and MIL also lost the target. After the target returned to its initial pose, all trackers except PJS-S drifted.

3) **skating1**: In this video sequence, the target object and camera move, and the target undergoes severe illumination change, pose variation, and occlusion by a similar object. Around frame 73, the target goes through rapid pose change, as a result only OAB, PJS-M, and PJS-S continued to track the target. Finally, after a severe illumination change, only PJS-S tracked the target until the sequence ended.

4) **crossing**: In this sequence a man (target) crosses a street and a car passes by, momentarily changing the target’s background. Almost all trackers were able to follow the target up to frame 41, when the car appears. After this, due to low resolution and decreased discrimination between the car and the target, Frag, IVT, APG, and MTT lost the target. A close look at the frames shows that PJS-M and PJS-S produce better fitting bounding boxes and are more robust against scale variation and low resolution.

5) **dollar**: The main challenge in this sequence is the existence of an object similar to the target, which is a ticket pass. The ticket is folded in frame 50, and then a similar unfolded ticket is introduced. All trackers except Frag and OAB were able to follow the ticket till the end of sequence.

6) **faceocc2**: In this video, the target undergoes pose change, occlusion, and rotation. Before frame 336, the target is not severely occluded and all trackers except the two proposed trackers exhibited a minor drift. In subsequent frames, this drift increases for all the competing methods, except for the two proposed trackers. For IVT, although the center location error is low, the overlap ratio decreased drastically. At the end, APG, IVT, PJS-S, and PJS-M had less drift compared to the

other trackers.

7) *stone*: The main challenge here is the existence of many objects that are similar to the target (which may also occlude the target). In frame 385, a similar object occludes the target, at which point Frag, MIL, and APG lost the target. In frame 510, the target passes near another similar object, where only PJS-S was able to overcome this challenge.

8) *sylv*: In this sequence, the camera is fixed and a stuffed cat (target) moves in front of a light source, resulting in illumination change, and in and out-of-plane rotation. During the final frames, and after large out-of-plane rotations, only PJS-S and PJS-M were able to track the target with the largest overlap.

9) *trellis*: This dataset is similar to *david* but exhibits larger pose change and sever illumination variations. After frame 212, a large change in illumination caused all trackers except PJS-S and PJS-M to drift or lose the target.

10) *walking2*: The main challenge of this sequence is sever occlusion by a similar object. In frame 200, after occlusion by the similar object, Frag, MIL, and MTT followed the occlusion, and APG and OAB drifted. This drift continued for the remainder of the sequence, and only PJS-S and PJS-M succeed to track the target with accurate alignment. This sequence also shows the ability of our methods to handle large scale changes.

D. Quantitative Comparison

The Center Location Error (CLE), and PASCAL overlap score (VOC) are typical quantitative measures which are usually plotted as a function of frame number. We have also calculated the mean values of these measures and presented them in tabular format for different trackers and datasets. The center location error measures the Euclidian distance between the centers of tracker, and ground truth bounding boxes. The frame and target size or orientation are not taken into account by CLE. Furthermore, after the target is lost, large values of CLE become meaningless and corrupt the mean CLE value. To resolve these deficiencies, the PASCAL overlap score is used. This score is defined as the ratio of intersection to union of tracker and ground truth bounding boxes: $\frac{|S_{gt} \cap S_{tr}|}{|S_{gt} \cup S_{tr}|}$, where $|R|$ denotes the number of pixels in region R . This measure which is normalized within $[0, 1]$, is independent of video and target sizes or orientation. The success rate measure which is equal to the ratio of successfully tracked frames in a video sequence, may be used to express the overall performance of a tracker. Moreover, a frame is successfully tracked if its overlap score is greater than a predefined threshold.

Figures 8 and 9 show the CLE measure and overlap score vs frame number, for the average results of all trackers, respectively. As illustrated, PJS-S and PJS-M are among the trackers with lowest CLE and highest VOC. As we mentioned earlier, CLE alone is not a fair measure for comparison. For example, in the walking sequence PJS-S and IVT have the lowest CLE but when comparing the VOC, PJS-S and PJS-M gain the best results. From these figures, we can observe the average performance of the trackers for the entire video sequence, and evaluate them with regard to the challenges

TABLE III
FROM TOP TO BOTTOM: CLE, OVERLAP RATIO, AND SUCCESS RATE AVERAGED ON 10 RUNS FOR DIFFERENT TRACKERS ON 10 DATASETS. BEST TWO RESULTS ARE HIGHLIGHTED.

	Frag	OAB	MIL	IVT	APG	MTT	PJS-M	PJS-S
board	212	118	57	151	191	149	37	31
crossing	58	5	3	4	86	38	6	2
david	19	114	17	8	62	18	11	14
dollar	57	68	16	15	19	6	7	7
faceocc2	42	12	14	10	29	10	14	13
skating1	112	28	151	273	134	336	46	23
stone	31	95	33	10	32	72	47	19
sylv	7	10	7	19	15	12	9	5
trellis	64	74	53	59	76	55	7	9
walking2	59	10	56	4	9	16	8	4
Average	66	54	41	55	65	71	19	13

	Frag	OAB	MIL	IVT	APG	MTT	PJS-M	PJS-S
board	0.15	0.28	0.52	0.19	0.11	0.19	0.63	0.65
crossing	0.31	0.67	0.73	0.27	0.37	0.46	0.70	0.74
david	0.42	0.17	0.42	0.43	0.29	0.45	0.59	0.54
dollar	0.36	0.33	0.62	0.64	0.71	0.80	0.80	0.80
faceocc2	0.47	0.67	0.66	0.56	0.46	0.68	0.65	0.64
skating1	0.24	0.36	0.13	0.06	0.07	0.07	0.46	0.50
stone	0.32	0.10	0.22	0.35	0.32	0.20	0.25	0.45
sylv	0.72	0.65	0.70	0.63	0.61	0.54	0.67	0.73
trellis	0.27	0.17	0.29	0.23	0.19	0.24	0.66	0.63
walking2	0.26	0.44	0.30	0.60	0.59	0.54	0.68	0.75
Average	0.35	0.38	0.46	0.40	0.37	0.42	0.61	0.64

	Frag	OAB	MIL	IVT	APG	MTT	PJS-M	PJS-S
board	14	20	35	11	5	11	57	70
crossing	37	69	86	21	45	51	88	95
david	10	18	5	33	33	29	51	46
dollar	39	38	48	52	87	99	100	99
faceocc2	45	67	62	52	50	70	67	63
skating1	15	20	9	4	5	5	40	41
stone	13	12	10	14	13	11	21	26
sylv	81	73	81	81	74	19	79	94
trellis	22	5	17	22	13	12	79	72
walking2	19	22	29	58	52	51	78	92
Average	30	34	38	35	38	36	66	70

posed in different frames. In this regard, the results confirm the observations of Section V-C. Table III provides a numerical comparison of mean CLE, overlap ratio, and success rate for an overlap ratio threshold equal to 0.6, over 10 runs for each pair of dataset and tracker. The average results over all datasets in the last row of the tables shows that the proposed trackers are the best among the competing trackers. Also, in all sequences except *faceocc2*, the proposed trackers achieve the first or second place. The main reason is the patchwise joint-sparse nature of our method, which causes the bounding boxes to accurately align with the target. The average success rate in all datasets for the two best trackers, PJS-S and PJS-M, is 69% and 65%, respectively. Using one specific threshold for tracker evaluation may not be adequately representative of its performance; Therefore similar to [19] we have used a success plot. The success plot shows the ratio of successful frames as the threshold varies from 0 to 1. The average success plots are shown in Fig. 10 for all datasets and trackers. These plots show that the proposed trackers, as previously shown by using

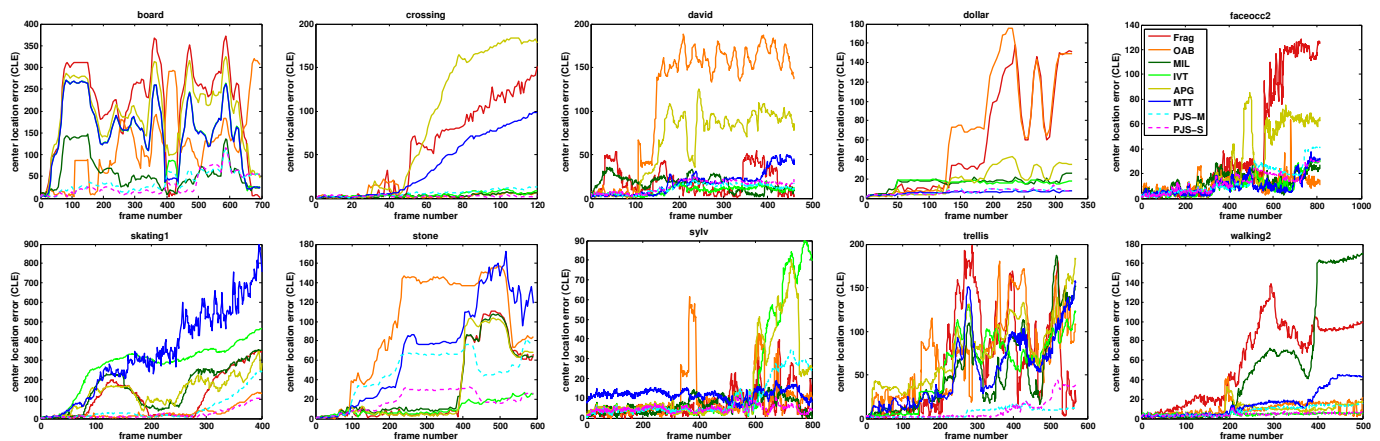


Fig. 8. **Center location error plots.** Distance between tracker and ground truth bounding boxes vs frame number. Results were averaged over 10 runs.

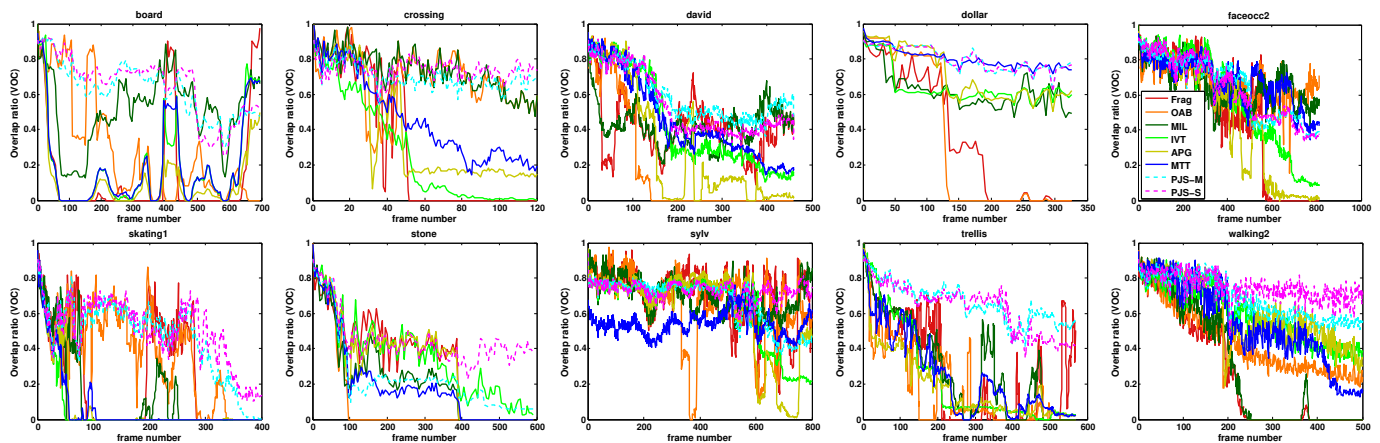


Fig. 9. **Overlap ratio plots.** Overlap ratio of tracker and ground truth bounding boxes vs frame number. Results were averaged over 10 runs.

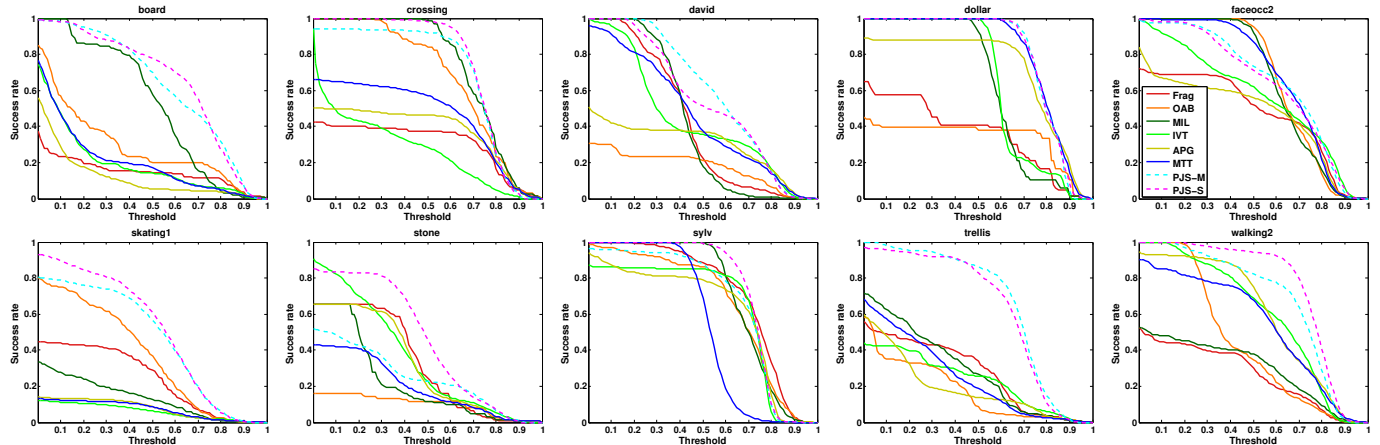


Fig. 10. **Success plot diagrams.** Success rates for thresholds in $[0, 1]$. Results were averaged over 10 runs.

other measures, achieved the best success rate for all datasets, except *faceocc2*.

To further investigate the effect of joint sparse coding on the proposed tracker’s (PJS-S) performance, the success plot for different values of group size, $k = 0, 2, 4, 6$ is depicted in Fig. 11 for *board*, *skating1*, and *walking2* datasets. As expected, increasing the group size will increase tracker performance,

as long as the temporal similarity assumption is not violated. In *board* and *walking2* group size $k = 6$ and $k = 4$ obtained the best results. On the other hand, in *skating1* due to severe occlusions and fast movements, group sizes higher than $k = 4$ invalidate our assumption and consequently the performance has decreased for $k = 6$.

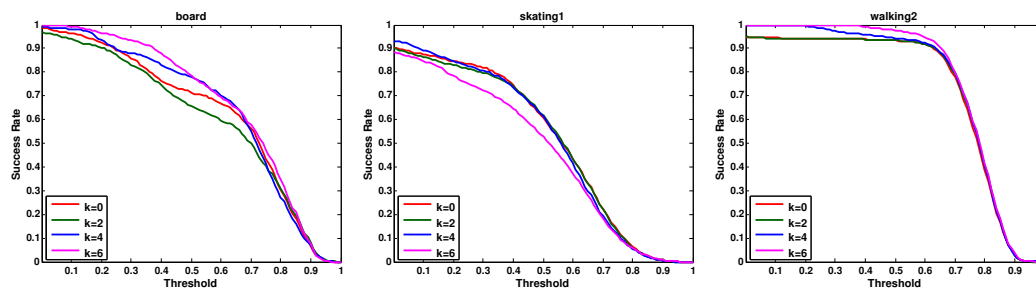


Fig. 11. **Group size effect.** Effect of group size on tracking performance of PJS-S. The results were averaged over 10 runs.

VI. CONCLUSION

In this paper, we exploited the temporal similarity assumption by using joint sparsity to enforce the targets in consecutive frames belong to a common subspace. The target image was viewed as a collection of non-overlapping patches. This resulted in a likelihood measure which is calculated as a sum over patch-reconstruction errors. Furthermore, an occlusion detection scheme was introduced by taking advantage of patch-reconstruction errors, and a prior probability of occlusion derived from an adaptive Markov chain of occlusion states. Extensive experimental results showed the effectiveness of the proposed approach on a variety of video sequences. In particular, our observations show that interpreting the target as a collection of smaller patches reduced the effects of occlusion by isolating its effects to the occluded patches. This is further important for updating the dictionary, because the occluded patches can be excluded from this procedure. In addition, we have observed that imposing the temporal similarity assumption generally improved the tracker's performance against appearance change, although this improvement was less profound for video sequences with rapid changes in appearance. Many directions may be followed for future work. The temporal similarity assumption may be imposed in other manners, the tracker's speed can be improved to make it suitable for real-time applications, and finally a weighted sum of patch-reconstruction errors may be able to improve the tracker's performance by assigning less weight to marginal patches that often contain the background and are more prone to occlusion.

REFERENCES

- [1] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, 2009.
- [2] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, 2010.
- [3] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa, "Compressive sensing for background subtraction," in *Proc. IEEE European Conf. on Comput. Vision (ECCV'08)*, 2008, pp. 155–168.
- [4] M. Aharon, A. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.
- [5] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 791–804, 2012.
- [6] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2259–2272, 2011.
- [7] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient ℓ_1 tracker with occlusion detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'11)*, 2011, pp. 1257–1264.
- [8] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'12)*, 2012, pp. 1830–1837.
- [9] X. Jia, H. Lu, and M. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'12)*, 2012, pp. 1822–1829.
- [10] B. Liu, J. Huang, L. Yang, and C. Kulikowski, "Robust tracking using local sparse appearance model and k-selection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'11)*, 2011, pp. 1313–1320.
- [11] W. Zhong and H. Lu, "Robust object tracking via sparsity-based collaborative model," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'12)*, 2012, pp. 1838–1845.
- [12] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'12)*, 2012, pp. 2042–2049.
- [13] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. IEEE European Conf. on Comput. Vision (ECCV'12)*, 2012, pp. 864–877.
- [14] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Low-rank sparse learning for robust visual tracking," in *Proc. IEEE European Conf. on Comput. Vision (ECCV'12)*, 2012, pp. 470–484.
- [15] —, "Robust visual tracking via structured multi-task sparse learning," *Int. J. of Comput. Vision*, vol. 101, no. 2, pp. 367–383, 2013.
- [16] D. Wang, H.-C. Lu, and M.-H. Yang, "Online object tracking with sparse prototypes," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 314–325, 2013.
- [17] Q. Wang, F. Chen, J. Yang, W. Xu, and M.-H. Yang, "Transferring visual prior for online object tracking," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3296–3305, 2012.
- [18] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'06)*, 2006, pp. 798–805.
- [19] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'13)*, 2013.
- [20] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm Computing Surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.
- [21] S. Zhang, H. Yao, X. Sun, and X. Lu, "Sparse coding based visual tracking: Review and experimental comparison," *Pattern Recognition*, vol. 46, no. 7, pp. 1772–1788, 2013.
- [22] S. Avidan, "Support vector tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 8, pp. 1064–1072, 2004.
- [23] —, "Ensemble tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 261–71, 2007.
- [24] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. British Machine Vision Conf.*, 2006, pp. 6.1–6.10.
- [25] B. Babenko, M. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, 2011.
- [26] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, 2012.
- [27] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, 2002.

- [28] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'06)*, vol. 1, 2006, pp. 260–267.
- [29] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. IEEE European Conf. on Comput. Vision (ECCV'08)*, 2008, pp. 234–247.
- [30] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof, "On-line semi-supervised multiple-instance boosting," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'10)*, 2010, pp. 1879–1879.
- [31] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 10, pp. 1025–1039, 1998.
- [32] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'00)*, 2000, pp. 142–149.
- [33] —, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, 2003.
- [34] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'91)*, 1991, pp. 586–591.
- [35] R. Basri and D. Jacobs, "Lambertian reflectance and linear subspaces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 2, pp. 218–233, 2003.
- [36] A. Wagner, J. Wright, A. Ganesh, Z. Zhou, H. Mobahi, and Y. Ma, "Toward a practical face recognition system: Robust alignment and illumination by sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 372–386, 2012.
- [37] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Int. J. of Comput. Vision*, vol. 26, no. 1, pp. 63–84, 1998.
- [38] D. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *Int. J. of Comput. Vision*, vol. 77, no. 1, pp. 125–141, 2008.
- [39] P. Tseng, "On accelerated proximal gradient methods for convex-concave optimization," *SIAM J. on Opt.*, 2008.
- [40] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR'10)*, 2010, pp. 3360–3367.
- [41] Z. Chen, "Bayesian filtering: From kalman filters to particle filters, and beyond," *Statistics*, pp. 1–69, 2003.
- [42] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online. nonlinear/non-gaussian bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, 2002.
- [43] X. Zhang, W. Li, W. Hu, H. Ling, and S. Maybank, "Block covariance based l1 tracker with a subtle template dictionary," *Pattern Recognition*, vol. 46, no. 7, pp. 1750–1761, 2013.
- [44] Y. Wu, H. Ling, E. Blasch, L. Bai, and G. Chen, "Visual tracking based on log-euclidean riemannian sparse representation," *Advances in Visual Computing*, pp. 738–747, 2011.
- [45] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of syst. of equations to sparse modeling of signals and images," *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.
- [46] L. Zelnik-Manor, K. Rosenblum, and Y. C. Eldar, "Dictionary optimization for block-sparse representations," *IEEE Trans. Signal Process.*, vol. 60, no. 5, pp. 2386–2395, 2012.
- [47] R. Baraniuk, V. Cevher, and M. Wakin, "Low-dimensional models for dimensionality reduction and signal recovery: A geometric perspective," *Proc. IEEE*, vol. 98, no. 6, pp. 959–971, june 2010.
- [48] J. Tropp, "Algorithms for simultaneous sparse approximation. part ii: Convex relaxation," *Signal Processing*, vol. 86, no. 3, pp. 589–602, 2006.
- [49] S. Cotter, B. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2477–2488, Jul. 2005.
- [50] D. Malioutov, M. Cetin, and A. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 3010–3022, Aug. 2005.
- [51] H. Lu, X. Long, and J. Lv, "A fast algorithm for recovery of jointly sparse vectors based on the alternating direction methods," *J. of Machine Learning Research*, vol. 15, pp. 461–469, 2011.
- [52] A. Rakotomamonjy, "Surveying and comparing simultaneous sparse approximation (or group-lasso) algorithms," *Signal processing*, vol. 91, no. 7, pp. 1505–1526, 2011.
- [53] J. Tropp, "Algorithms for simultaneous sparse approximation. part i: Greedy pursuit," *Signal Processing*, vol. 86, no. 3, pp. 572–588, 2006.
- [54] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1 -problems in compressive sensing," *SIAM J. on scientific computing*, vol. 33, no. 1, pp. 250–278, 2011.
- [55] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. 26th Annual Int. Conf. on Machine Learning*, 2009, pp. 689–696.
- [56] —, "Online learning for matrix factorization and sparse coding," *The J. of Machine Learning Research*, vol. 11, pp. 19–60, 2010.



Ali Zarezade received his B.Sc. and M.Sc in Aerospace Engineering from Amirkabir and Sharif University of Technology, Tehran, Iran, in 2007 and 2009, respectively. He also received an M.Sc. in Artificial Intelligence from Sharif University of Technology, Tehran, Iran in 2013, and is currently working toward the Ph.D. degree in the Department of Computer Engineering at Sharif University of Technology.

His current research interests include the application of machine learning and sparse representation in computer vision and visual tracking.



Hamid R. Rabiee (SM'07) received his B.S. (1987) and M.S. (1989) degrees (with great distinction) in Electrical Engineering from CSULB, USA, his EEE in Electrical and Computer Engineering from USC, USA, and his Ph.D. (1996) in Electrical and Computer Engineering from Purdue University, West Lafayette, USA.

From 1993 to 1996 he was a Member of Technical Staff at AT&T Bell Laboratories. From 1996 to 1999 he worked as a Senior Software Engineer at Intel Corporation. He was also with PSU, OGI, and OSU Universities as an adjunct professor of Electrical and Computer Engineering from 1996 to 2000. Since September 2000, he has joined Sharif University of Technology (SUT), Tehran, Iran. He is the founder of Sharif University Advanced Information and Communication Technology Research Center (AICT), Advanced Technologies Incubator (SATI), Digital Media Laboratory (DML) and Mobile Value Added Services Laboratory (VASL). He is currently a Professor of Computer Engineering at Sharif University of Technology and the Director of AICT, DML and VASL. He has been the initiator and director of national and international level projects in the context of UNDP International Open Source Network (IOSN) and Iran National ICT Development Plan.

Prof. Rabiee has received numerous awards and honors for his industrial, scientific and academic contributions. He has acted as chairman in a number of national and international conferences, and holds three patents.



Ali Soltani-Farani received his M.Sc. in Computer Architecture from Sharif University of Technology, Tehran, Iran, in 2010 and a B.Sc. in Hardware Engineering from the same university in 2008. He is currently working toward the Ph.D. degree in the Department of Computer Engineering at Sharif University of Technology.

His current research interests include structured sparse representations, dictionary learning, and their application to signal and image processing.



Ahmad Khajenezhad received his M.Sc. in Artificial Intelligence from Sharif University of Technology, Tehran, Iran, in 2012 and a B.Sc. in Software Engineering from the same university in 2010. He is currently working toward the Ph.D. degree in the Department of Computer Engineering at Sharif University of Technology.

His current research interests include the application of machine learning and sparse representation in computer vision and visual tracking.