Robust Visual Tracking and Vehicle Classification via Sparse Representation

Xue Mei and Haibin Ling, *Member, IEEE*

Abstract—In this paper, we propose a robust visual tracking method by casting tracking as a sparse approximation problem in a particle filter framework. In this framework, occlusion, noise and other challenging issues are addressed seamlessly through a set of trivial templates. Specifically, to find the tracking target in a new frame, each target candidate is sparsely represented in the space spanned by target templates and trivial templates. The sparsity is achieved by solving an ℓ_1-regularized least squares problem. Then the candidate with the smallest projection error is taken as the tracking target. After that, tracking is continued using a Bayesian state inference framework. Two strategies are used to further improve the tracking performance. First, target templates are dynamically updated to capture appearance changes. Second, nonnegativity constraints are enforced to filter out clutters which negatively resemble tracking targets. We test the proposed approach on numerous sequences involving different types of challenges including occlusion and variations in illumination, scale, and pose. The proposed approach demonstrates excellent performance in comparison with previously proposed trackers. We also extend the method for simultaneous tracking and recognition by introducing a static template set, which stores target images from different classes. The recognition result at each frame is propagated to produce the final result for the whole video. The approach is validated on a vehicle tracking and classification task using outdoor infrared video sequences.

Index Terms—Visual tracking, sparse representation, compressive sensing, simultaneous tracking and recognition, particle filter, ℓ_1 minimization.

1 Introduction

Compressive sensing [8], [13], or sparse representation, has played a fundamental role in many research areas. The problem is to exploit the compressibility and sparsity of the true signal and use a lower sampling frequency than the Shannon-Nyquist rate. Sparsity then leads to efficient estimation, compression, dimensionality reduction, and modeling. Within the computer vision community, compressive sensing has been applied to a variety of problems such as face recognition [43], background subtraction [9], [20], media recovery [16], texture segmentation [31], lighting estimation [33], and so on.

Roughly speaking, compressive sensing is a technique for reconstructing a signal (e.g., an image) using the prior knowledge that the reconstruction is sparse or compressible. With this technique, the signal is represented by a sparse set of basis functions. That is, all of the coefficients corresponding to the basis functions vanish except for a few. The idea of compressive sensing has recently been exploited for object recognition, especially for face recognition [43]. In the work, a test face image is linearly represented in terms of all the images in the training set containing numerous subjects. In the representation, the coefficients associated with the images that have the same subject as the test image are usually nonzeros while other coefficients vanish. Thus, the representation vector encodes the identity of the test image. The method has been shown to be robust to various forms of image corruptions and especially to occlusion. In this paper, we exploit sparse representation for robust visual tracking with the intuition that the appearance of a tracked object can be sparsely represented by its appearances in previous frames.

The challenges in designing a robust visual tracking algorithm are caused by the presence of noise, occlusion, varying viewpoints, background clutter, and illumination changes [45]. To overcome these challenges, we develop a robust visual tracking framework by casting the tracking problem as finding a sparse approximation in a template subspace. Motivated by the work in [43], we propose handling occlusion using a set of unit vectors, which are dubbed as *trivial templates*. Each trivial template $\mathbf{i}_j$ has only one non-zero element, 1, at the jth position, $j = 1, 2, \cdots$ (see Fig. 1). The trivial templates can be arranged in a certain order to form an identity matrix. A target candidate is represented as a linear combination of the template set composed of both target templates (obtained from previous frames) and trivial templates. Intuitively, a good target candidate can be efficiently represented by the target templates. This leads to a sparse coefficient vector, since coefficients corresponding to trivial templates (named trivial coefficients) tend to be zeros. In the case of occlusion (and/or other unpleasant issues such as noise corruption or background clutter), a limited number of trivial coefficients will be activated, while the whole coefficient vector remains sparse. A bad target candidate, on the contrary, often leads to a dense representation[1] (e.g., Fig. 3). The sparse representation is achieved through solving an ℓ_1-regularized least squares problem which can be done efficiently through convex optimization. Then the candidate with the smallest target template projection error is chosen as the tracking result. After that, tracking is led by the Bayesian state inference framework in which a particle filter is used for propagating sample distributions over time.

1. Candidates similar to trivial templates have sparse representations, but they are easily filtered out for their large dissimilarities to target templates.

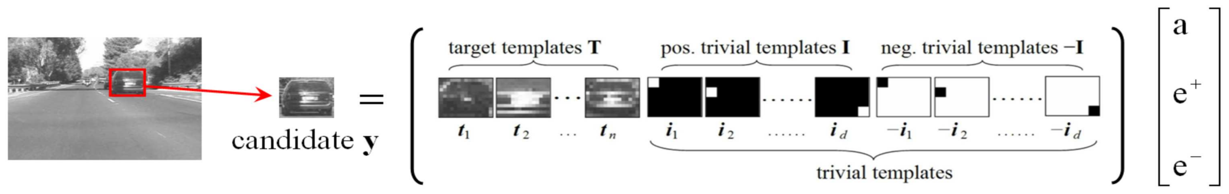

Fig. 1. Representation and templates used in our proposed approach (from the testing sequence, Fig. 10).

To further improve robustness, we enforce nonnegativity constraints and dynamically update the target templates. Non-negativity constraints are especially helpful to eliminate clutter that is similar to target templates with reversed intensity patterns. The constraints are implemented by introducing a new set of negative trivial templates. We rename the trivial templates to positive trivial templates for differentiation. The j^{th} negative trivial template, $-\mathbf{i}_j$, is the reflection of its counterpart, $\mathbf{i}_j$, with only one negative element, -1, at the jth position. Dynamic template update scheme keeps the representative templates throughout the tracking procedure. This is done by adjusting template weights using the coefficients in the sparse representation.

We test the proposed approach on numerous video sequences involving heavy occlusion, large illumination and pose changes. Both normal size and long sequences (more than 2000 frames) are used in the experiments. The proposed approach shows excellent performance in comparison with four previously proposed trackers.

We extend our tracking algorithm for simultaneous tracking and recognition by introducing a set of static target templates and modeling the uncertainties of both tracking and recognition. To differentiate the static target templates which keep constant throughout the tracking, we name the target templates in the tracking algorithm dynamic target templates, which are updated with respect to the template update scheme. By embedding both the static and dynamic target templates in a particle filter framework, the appearance variation of the target is successfully modeled and captured in the video sequence. Therefore, we are able to achieve a reliable tracker and an accurate classifier to handle the difficulties such as pose changes and background noise. We obtain promising results by applying the proposed method on IR-based video sequences for vehicle classification.

The rest of the paper is organized as follows. In the next section related works are summarized. After that, the particle filter algorithm is reviewed in §3. §4 details the tracking algorithm using ℓ_1 minimization and our robust template update scheme. §5 presents the extended work on simultaneous tracking and recognition. Experimental results are reported in §6. Finally, we conclude this paper in §7.

2 Related Work

Tracking is an important topic in computer vision and it has been studied for several decades. In this section we summarize studies that are related to our work; a thorough survey can be found in [45].

Given the wide range and long study of visual tracking, our proposed tracking method borrows many ideas from previous work. First, we use the popular particle filter framework that considers tracking as an estimation of the states for a time series state space model. The particle filter, also known as the sequential Monte Carlo method [14], recursively constructs the posterior probability density function of the state space using Monte Carlo integration. It has been developed in the computer vision community and applied to tracking problems under the name Condensation [21]. It is later extended in many variations such as the Rao-Blackwellized particle filter [25]. There are also recent studies using particle filters similar to our method [19], [47]. For example, [19] proposes an object tracking method based on combination of local classifiers in the presence of partial occlusion. The weight for combining local classifiers is selected adaptively using particle filters.

The proposed method relates to previous work on appearance-based trackers, e.g., [4], [22], [47]. Early examples include using the sum of squared difference (SSD) as a cost function in the tracking problem [4] and using mixture model to model the appearance variation [22]. The work in [47] incorporates an appearance-adaptive model in a particle filter to realize robust visual tracking and classification algorithms. Our modeling using linear template combination is related to both subspace- and template-based tracking approaches [6], [18], [38]. The appearance of the object is represented using an eigenspace [6], affine warps of learned linear subspaces [18], or an adaptive low-dimensional subspace [38].

There are many previous studies treating tracking as a classification problem where learning techniques are naturally involved. Examples include [2], [10], [46], [42], [3], [23]. In [2], a feature vector is constructed for every pixel in the reference image and an adaptive ensemble of classifiers is trained to separate pixels that belong to the object from the ones in the background. In [10], a confidence map is built by finding the most discriminative RGB color combination in each frame. A hybrid approach combining a generative model and a discriminative classifier is used in [46] to capture appearance changes and allow reacquisition of an object after total occlusion. Sparse Bayesian learning is used in [42]. Online multiple instance learning is used in [3] to achieve robustness to occlusions and other image corruptions. In [23], a new tracker is proposed by bootstrapping binary classifiers with structural constraints, and the tracker is shown to be reliable in long sequence tracking.

Tracking can also be combined with either detection [1], [5], [44] or recognition [15], [28], [29], [36], [39], [47]. In [1], a method is proposed for detecting and tracking people in cluttered real-world scenes containing many people and changing background. In [15], an adaptive framework is proposed for learning human identity by using the motion information along the video sequence to improve both face tracking and recognition. In [47], simultaneous track and recognize human faces is studied in a particle filter framework. The recognition determines the identity of the person that is fixed in the whole video sequence. In [29], hidden Markov model is used for video-based face recognition.

Despite sharing many ideas with previous work as discussed above, our proposed method, to the best of our knowledge, is the first one that combines the ℓ_1 regularization and sparse representation for visual tracking. Our work can be viewed as an extension of recent study in visual recognition via sparse representation [43], which has been successfully demonstrated for robust face recognition. Because of this, the method naturally addresses recognition tasks simultaneously with tracking. A preliminary conference version of this paper appears in [34].

3 Particle Filter

The particle filter [14] is a Bayesian sequential importance sampling technique for estimating the posterior distribution of state variables characterizing a dynamic system. It provides a convenient framework for estimating and propagating the posterior probability density function of state variables regardless of the underlying distribution. It consists of essentially two steps: prediction and update. Let $\mathbf{x}_t$ denote the state variable describing the affine motion parameters of an object at time t. The predicting distribution of $\mathbf{x}_t$ given all available observations $\mathbf{y}_{1:t-1} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_{t-1}\}$ up to time $t-1$, denoted by $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$, is recursively computed as

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \ . \tag{1}$$

At time t, the observation $\mathbf{y}_t$ is available and the state vector is updated using the Bayes rule

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})} \ , \tag{2}$$

where $p(\mathbf{y}_t|\mathbf{x}_t)$ denotes the observation likelihood.

In the particle filter, the posterior $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is approximated by a finite set of N samples $\{\mathbf{x}_t^i\}_{i=1,\cdots,N}$ with importance weights w_t^i. The candidate samples $\mathbf{x}_t^i$ are drawn from an importance distribution $q(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t})$ and the weights of the samples are updated as

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

The samples are resampled to generate a set of equally weighted particles according to their importance weights to avoid degeneration. In the case of the bootstrap filter $q(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{y}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and the weights become the observation likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$.

3.1 Motion Model

In the tracking framework, we apply an affine image warping to model the object motion of two consecutive frames. The tracking process is governed by the two important components: state transition model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ which models the temporal correlation of state transition between two consecutive frames, and observation model $p(\mathbf{y}_t|\mathbf{x}_t)$ which measures the similarity between the appearance observation and the approximation using the target model.

3.1.1 State Transition Model

The state variable $\mathbf{x}_t$ is modeled by the six parameters of the affine transformation parameters $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, t_1, t_2)$ and velocity components (v_1, v_2), where $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$ are the deformation parameters, (t_1, t_2) are the 2D position parameters, and (v_1, v_2) are the velocity of the horizontal and vertical translation parameters (t_1, t_2).

To propagate the particles, sometimes the velocity of the motion model is also taken into account [5], [7]. The deformation parameters in $\mathbf{x}_t$ are modeled independently by a Gaussian distribution around the previous state $\mathbf{x}_{t-1}$. We use an average velocity for the translation parameters to model the object motion.

$$\begin{aligned}(\alpha_1, \alpha_2, \alpha_3, \alpha_4, t_1, t_2)_t &= (\alpha_1, \alpha_2, \alpha_3, \alpha_4, t_1, t_2)_{t-1} \\ &\quad + (0,0,0,0,v_1,v_2)_{t-1} \cdot \Delta t + \epsilon_t \ , \qquad (4) \\ (v_1, v_2)_t &= \frac{1}{n} \sum_{j=1,\ldots,n} (v_1, v_2)_{t-j} \ . \qquad (5)\end{aligned}$$

The process noise ϵ_t for each state variable is independently drawn from zero-mean normal distributions. The variances of affine parameters are different and do not change over time. Δt depends on the frame-rate of the sequence. n is the number of previous velocities used for averaging. The larger the variances of the parameters, the more particles are needed to model the distribution. However, the cost of the computation to evaluate all the possible particles will be high as well. Some work [27], [38], [47] exploits the balance between the efficiency (less particles, less precision) and effectiveness (more particles, better precision) in visual tracking.

3.1.2 Observation Model

By applying an affine transformation using $\mathbf{x}_t$ as parameters, we crop the region of interest $\mathbf{y}_t$ from the image and normalize it to be the same size as the target templates in the gallery. The observation model $p(\mathbf{y}_t|\mathbf{x}_t)$ reflects the similarity between a target candidate and the target templates and the probability is formulated from the error approximated by the target templates using ℓ_1 minimization (see §4.1 for details). The likelihood of the projected sample is governed by a Gaussian distribution as follows:

$$p(\mathbf{y}_t|\mathbf{x}_t) = \prod_{j=1,\ldots,d} \mathbb{N}(\widetilde{\mathbf{y}}_t(j); 0, \sigma^2) \ , \tag{6}$$

where $\mathbb{N}(.)$ is the Gaussian distribution and σ^2 is the variance, d is the number of pixels in the appearance model, and $\widetilde{\mathbf{y}}_t(j)$ denotes the j-th pixel approximation residual of observation $\mathbf{y}_t$ by the target template set using ℓ_1 minimization (§4.1). We use the same variance σ^2 for all the pixels in the appearance model. Note the approximation residuals are assumed to be independent of each other. It is a simple assumption that the image pixel is corrupted by independent Gaussian noise. Such an assumption is often made (e.g., [47]) due to its simplicity. The assumption works well in our experiments. However, in reality the noises over different pixels are seldom independent. For example, illumination change can make similar "noise" in the shadowed region. Therefore, more accurate models could be used for further improvement of our method.

4 ℓ_1 Minimization Tracking

4.1 Sparse Representation of a Tracking Target

The global appearance of an object under different illumination and viewpoint conditions is known to lie approximately in a low dimensional subspace [28], [46]. In tracking, such a subspace can be treated as spanned by a set of templates, which we denote as $\mathbf{T} = [\mathbf{t}_1 \dots \mathbf{t}_n] \in \mathbb{R}^{d\times n}$ $(d >> n)$ containing n target templates where $\mathbf{t}_i \in \mathbb{R}^d$ (we stack template image columns to form a 1D vector). A tracking result, $\mathbf{y} \in \mathbb{R}^d$, is an image patch in the current frame and normalized to have the same size as the template. Therefore, it approximately lies in the linear span of $\mathbf{T}$,

$$\mathbf{y} \approx \mathbf{T}\mathbf{a} = a_1\mathbf{t}_1 + a_2\mathbf{t}_2 + \cdots + a_n\mathbf{t}_n \ , \tag{7}$$

where $\mathbf{a} = (a_1, a_2, \cdots, a_n)^\top \in \mathbb{R}^n$ is called a *target coefficient vector*.

In many visual tracking scenarios, targets are often corrupted by noise or partially occluded. The occlusion creates unpredictable errors. It may affect any part of the target and appear at any size on the target. The occlusion can be either a connected region of pixels or a number of randomly scattered pixels, though the former is more likely to be occluded. In this paper, we treat equally all pixels in the target image patch for simplicity. Since the number of the trivial templates is the value of the multiplication of the width and height of the template, it remains constant during the tracking. To incorporate the effect of occlusion and noise, Eqn. (7) is rewritten as

$$\mathbf{y} = \mathbf{T}\mathbf{a} + \epsilon \ , \tag{8}$$

where ϵ is the error vector. The nonzero entries of ϵ indicate the pixels in $\mathbf{y}$ that are corrupted or occluded. The locations of corruption can differ for different tracking images and are unknown to the computer. Following the scheme in [43], we can use trivial templates $\mathbf{I} = [\mathbf{i}_1, \mathbf{i}_2, ..., \mathbf{i}_d] \in \mathbb{R}^{d\times d}$ to capture the occlusion as

$$\mathbf{y} = \begin{bmatrix}\mathbf{T}, & \mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{a} \\ \mathbf{e}\end{bmatrix} \ , \tag{9}$$

where a *trivial template* $\mathbf{i}_i \in \mathbb{R}^d$ is a vector with only one nonzero entry (i.e. $\mathbf{I}$ is an identity matrix), and $\mathbf{e} = (e_1, e_2, \cdots, e_d)^\top \in \mathbb{R}^d$ is called a *trivial coefficient vector*.

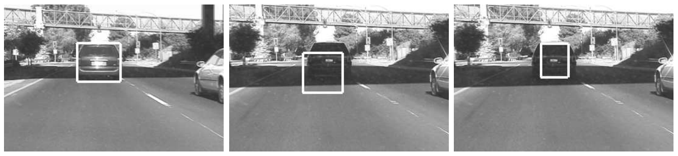
Fig. 2. Left: target template. Middle: tracking result without non-negativity constraint. Right: tracking result with non-negativity constraint.

4.2 Nonnegativity Constraints

In principle, the coefficients in $\mathbf{a}$ can be any real numbers if the target templates are taken without restrictions. However, we argue that in tracking, a tracking target can almost always be represented by the target templates dominated by nonnegative coefficients. Here by "dominated" we mean that the templates that are most similar to the tracking target are positively related to the target. This is true when we start tracking from the second frame. The target is selected in the first frame manually or by a detection method. The target in the second frame will look more like the target in the first frame such that the coefficient is positive when the target in the first frame is used to approximate the target in the second frame. In new frames, the appearance of targets may change, but new templates will be brought in (may replace old templates) and the coefficients will still be positive for the most similar target templates in the following frames.

Another important reason for including nonnegative coefficients comes from their ability to filter out clutter that is similar to target templates at reversed intensity patterns, which often happens when shadows are involved. We give an example in Fig. 2. In Fig. 2, the left image shows the first frame in which the target template is created in the bounding box. The middle image shows the tracking result without nonnegativity constraints, where the tracking result is far from the correct location. By checking the coefficients in $\mathbf{a}$, we found that the failure is because that the intensity pattern in the tracking result (dark in top and light in bottom) is roughly reversed compared to the target template (dark in bottom and light in top). This problem can be avoided by enforcing the nonnegativity constraints, as shown in the right image.

Enforcing nonnegativity constraints on the target coefficient vector $\mathbf{a}$ is straightforward. However, it is unreasonable to put such constraints directly on the trivial coefficient vector $\mathbf{e}$. For this reason, we propose extending the trivial templates by including *negative trivial templates* as well. Consequently, model (9) is now written as

$$\mathbf{y} = \begin{bmatrix}\mathbf{T}, & \mathbf{I}, & -\mathbf{I}\end{bmatrix} \begin{bmatrix}\mathbf{a} \\ \mathbf{e}^+ \\ \mathbf{e}^-\end{bmatrix} \hat{=} \mathbf{B}\mathbf{c} \ , \quad \text{s.t.} \ \ \mathbf{c} \succcurlyeq 0 \ , \tag{10}$$

where $\mathbf{e}^+, \mathbf{e}^- \in \mathbb{R}^d$ are called *positive* and *negative* trivial coefficient vectors respectively; $\mathbf{B} = [\mathbf{T}, \mathbf{I}, -\mathbf{I}] \in \mathbb{R}^{d\times(n+2d)}$; $\mathbf{c}^\top = [\mathbf{a}, \mathbf{e}^+, \mathbf{e}^-] \in \mathbb{R}^{n+2d}$ is a non-negative coefficient vector; and "$\succcurlyeq$" denotes elementary-wise "$\geq$". Example templates are illustrated in Fig. 1.

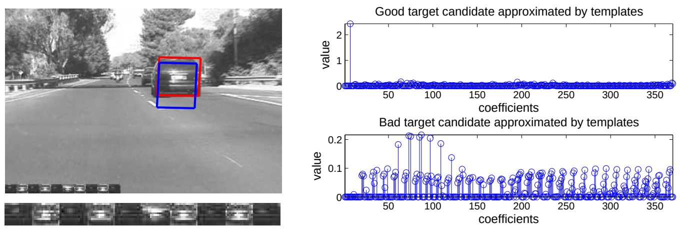

Fig. 3. Top left: good and bad target candidates. Bottom left: ten templates in the template set. Top right: approximation of the good target candidate by the template set. Bottom right: approximation of the bad target candidate.

Algorithm 1 Template Update

```
1: $\mathbf{y}$ is the newly chosen tracking target.
2: $\mathbf{a}$ is the solution to (11).
3: $\mathbf{w}$ is current weights, such that $w_i \leftarrow \|\mathbf{t}_i\|_2$.
4: $\tau$ is a predefined threshold.
5: Update weights according to the coefficients of the target templates. $w_i \leftarrow w_i * exp(a_i)$.
6: if ( $sim(\mathbf{y}, \mathbf{t}_m) < \tau$ ), where $sim$ is a similarity function. $\mathbf{t}_m$ has the largest coefficient $a_m$, where $m = \arg\max_{1 \le i \le n} a_i$ then
7:     $i_0 \leftarrow \arg\min_{1 \le i \le n} w_i$
8:     $\mathbf{t}_{i_0} \leftarrow \mathbf{y}$,    /*replace an old template*/.
9:     $w_{i_0} \leftarrow \text{median}(\mathbf{w})$,    /*replace an old weight*/.
10: end if
11: Normalize $\mathbf{w}$ such that $sum(\mathbf{w}) = 1$.
12: Adjust $\mathbf{w}$ such that $max(\mathbf{w}) = 0.3$ to prevent skewing.
13: Normalize $\mathbf{t}_i$ such that $\|\mathbf{t}_i\|_2 = w_i$.
```

4.3 Achieving Sparseness through ℓ_1 Minimization

The system in (10) is underdetermined and does not have a unique solution for $\mathbf{c}$. The error caused by occlusion and noise typically corrupts a fraction of the image pixels. Therefore, for a good target candidate, there are only a limited number of nonzero coefficients in $\mathbf{e}^+$ and $\mathbf{e}^-$ that account for the noise and partial occlusion. Consequently, we want to have a sparse solution to (10). We exploit the compressibility in the transform domain by solving the problem as an ℓ_1-regularized least squares problem, which is known to typically yield sparse solutions [43]

$$\min \|\mathbf{Bc} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{c}\|_1 \quad , \quad \text{s.t.} \quad \mathbf{c} \succcurlyeq 0 \quad , \tag{11}$$

where $\|.\|_1$ and $\|.\|_2$ denote the ℓ_1 and ℓ_2 norms respectively.

Fig. 3 shows the coefficients approximated by the template set for the good and bad target candidates. The good and bad target candidates are shown in the red and blue bounding boxes on the top left image. The 10 target templates with size of 12×15 are shown on the left corner, while the enlarged version is shown on the bottom left image. The images on the right show the good and bad target candidate approximated by the template set, respectively. The first 10 coefficients correspond to the 10 target templates used in the tracking and the remaining 360 coefficients correspond to the trivial templates. In the top right image, the seventh coefficient is relatively large compared to the remaining coefficients. Thus, the seventh target template represents the good candidate well, and the trivial templates are a small factor in approximating the good candidate. In the bottom right image, the coefficients are densely populated and trivial templates account for most of the approximation for a bad candidate in the left image.

Our implementation solves the ℓ_1-regularized least squares problem via an interior-point method based on [26]. The method uses the preconditioned conjugate gradients (PCG) algorithm to compute the search direction, and the run time is determined by the product of the total number of PCG steps required over all iterations and the cost of a PCG step. The total number of PCG iterations required by the truncated Newton interior-point method depends on the value of the regularization parameter λ. In the experiments, we found that the total number of PCG to compute a solution is a few hundred. The computationally most expensive operation for a PCG step is a matrix-vector product which has $O(d(2d+n)) = O(d^2 + dn)$ computing complexity, where d is the number of pixels of the target template, and n is the number of target templates. We use the code from [12] for the minimization task.

We then find the tracking result by finding the smallest residual after projecting on the target template subspace. Specifically, at frame t, let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n_p}\}$ be the n_p state candidates and $\mathcal{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{n_p}\}$ be the corresponding target candidates; the tracking result $\widehat{\mathbf{y}}$ is chosen by

$$\widehat{\mathbf{y}} = \text{argmax}_{\mathbf{y} \in \mathcal{Y}} \prod_{j=1,\dots,d} \mathbb{N}((\mathbf{y} - \mathbf{Ta})(j); 0, \sigma^2) \ , \tag{12}$$

where $\mathbf{a}$ is achieved by the ℓ_1 minimization (11), and $(\mathbf{y} - \mathbf{Ta})(j)$ denotes the j-th element in $\mathbf{y} - \mathbf{Ta}$.

4.4 Template Update

Template tracking was suggested in the computer vision literature in [30], dating back to 1981. The object is tracked through the video by extracting a template from the first frame and finding the object of interest in successive frames. In [17], a fixed template is matched with the observation to minimize a cost function in the form of sum of squared distance (SSD). A fixed appearance template is not sufficient to handle recent changes in the video, while on the other hand, a rapidly changing model [40] which uses the best patch of interest in the previous frame is susceptible to drift. Approaches have been proposed to overcome the drift problem [22], [24], [32] in different ways.

Intuitively, object appearance remains the same only for a certain period of time, but eventually the template is no longer an accurate model of the object appearance. If we do not update the template, the template cannot capture the appearance variation due to illumination or pose changes. If we update the template too often, small errors are introduced each time the template is updated. The errors are accumulated and the tracker drifts from the target. We tackle this problem by dynamically updating the target template set $\mathbf{T}$.

To dynamically update the target template set $\mathbf{T}$, we need to introduce importance weights for the templates in order to prefer more stable ones, and identify rarely used templates at the same time. One important feature of ℓ_1 minimization is that it favors templates with large norms because of the regularization part $\|\mathbf{c}\|_1$. The larger the norm of $\mathbf{t}_i$ is, the smaller coefficient a_i is needed in the approximation $\|\mathbf{y} - \mathbf{B}\mathbf{c}\|_2$. We exploit this characteristic by assigning $\|\mathbf{t}_i\|_2$ to the weight w_i of the template $\mathbf{t}_i$. The weight is updated during the tracking, and so does the norm of the template. Therefore, the larger the weight is, the more important the template is. At initialization, the first target template is manually selected from the first frame and zero-mean-unit-norm normalization is applied. The remaining target templates are created by perturbating a few pixels in four possible directions at the corner points of the first template in the first frame. Thus we create all the target templates (10 for our experiments) at the first frame. The target template set $\mathbf{T}$ is then updated with respect to the coefficients of the tracking result.

The template update scheme is summarized in Alg. 1. It includes three main operations: template replacement, template updating, and weight updating. If the tracking result $\mathbf{y}$ is not similar to the current template set $\mathbf{T}$, it will replace the least important template in $\mathbf{T}$ and be initialized to have the median weight of the current templates. Having a median weight of the current templates effectively prevents the newly added template from having a dominant role in approximating the tracking candidate in the following frames. Thus it attenuates the drifting problem. The weight of each template increases when the appearance of the tracking result and template is close enough and decreases otherwise. The similarity function $sim(.,.)$ is defined as $cos(\theta)$, where θ is the angle between the two input (normalized) vectors. τ can be any value between 0 and 1 depending on the experiment. In addition, we adjust the maximum template weight to be 0.3 to prevent the skewing, in that situation there can be a template with an extremely large weight (e.g., 0.9). Such a template dominates the template set and can cause problems in frames that follow.

The above template update schemes aim at reducing drift. These schemes show excellent performance in our experiments. However, as in many trackers, drift can still happen in theory and in practice. For example, when the target is occluded by a similar object for many frames. We expect further future study along this direction.

5 Simultaneous Tracking and Recognition

In this section, we extend our ℓ_1 tracker and propose a simultaneous tracking and recognition method using ℓ_1 minimization in a probabilistic framework. When we locate the moving target in the image, we want to identify it based on the template images in the gallery. Typically, tracking is solved before recognition. When the object is located in the frame, it is cropped from the frame and transformed using an appropriate transformation. This tracking and recognition are performed sequentially and separately to resolve the uncertainty in the video sequences. The recognition after tracking strategy poses some difficulties such as selecting good frames and estimation of parameters for registration. Previous studies have shown the effectiveness of simultaneous tracking and recognition on certain cases [15], [47], [36], [29], [39], [35]. We propose extend the above sparse tracking for the same purpose and apply it for vehicle classification in IR-based video sequences.

5.1 Framework

Given a video sequence, our task now is to simultaneously track the target of interest and recognize it. Suppose there are k object classes, then the recognition is to assign the tracking target to one of them. To use the same particle filter framework, we first introduce an identity variable o_t which ranges from class 1 to class k. Then, in the particle filter framework, the problem of simultaneous tracking and recognition is to infer the motion state variable $\mathbf{x}_t$ and identity variable o_t given the observations $\mathbf{y}_{1:t}$ from frame 1 to t. We rewrite (2) as follows

$$p(\mathbf{x}_t, o_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t, o_t)p(\mathbf{x}_t, o_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})} \ , \tag{13}$$

where $p(\mathbf{x}_t, o_t|\mathbf{y}_{1:t-1})$ can be recursively computed as follows:

$$\begin{aligned} &p(\mathbf{x}_t, o_t|\mathbf{y}_{1:t-1}) \\ &= \int p(\mathbf{x}_t, o_t|\mathbf{x}_{t-1}, o_{t-1})p(\mathbf{x}_{t-1}, o_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \ . \end{aligned} \tag{14}$$

We assume the $\mathbf{x}_t$ and o_t are independent of each other and $p(o_t|o_{t-1})$ is a uniform distribution. Eqn. (14) can be simplified as

$$\begin{aligned} &p(\mathbf{x}_t, o_t|\mathbf{y}_{1:t-1}) \\ &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(o_t|o_{t-1})p(\mathbf{x}_{t-1}, o_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \\ &\propto \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}, o_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \ . \end{aligned} \tag{15}$$

With the above extension, our task boils down to finding the most likely candidate from the joint state-class candidate set $\mathcal{X} \times \{1, ..., k\}$ and estimation of $p(\mathbf{y}_t|\mathbf{x}_t, o_t)$. This leads to the following extension of sparse representation in §4.1.

It is worth noting that our task is to find the object identity using the *whole* sequence. This is done through accumulating the identity estimation over frames, i.e., the inference of o_t. In general $p(o_t|o_{t-1})$ does not follow a uniform distribution. However, this assumption greatly simplifies the inference and shows excellent performance in our experiments. Intuitively, though the inference on each individual frame can be inaccurate and unstable, the fused results across the whole sequence are usually rather reliable. The phenomenon is observed clearly in our experiments in §6.2.

5.2 Representation

To extend the sparse representation (10) for recognition, we include, in addition to $\mathbf{T}$, a group of template sets $\mathbf{T}^{(s)} = [\mathbf{T}^{(s)}_1, \mathbf{T}^{(s)}_2, \cdots, \mathbf{T}^{(s)}_k]$, such that each $\mathbf{T}^{(s)}_i$ contains the templates collected from class i. Compared to the template set $\mathbf{T}$ that undergoes *dynamic updating* during the tracking procedure, $\mathbf{T}^{(s)}$ remains unchanged all the time, i.e., *static*. The static template set has the target images at different pose and illumination, and can effectively prevents the tracker from drifting when the target undergoes drastic pose and illumination changes. With these template sets, a target candidate $\mathbf{y}$ has the following representation,

$$\mathbf{y} = \left[\mathbf{T}^{(s)}, \mathbf{T}, \mathbf{I}, -\mathbf{I}\right] \begin{bmatrix} \mathbf{a}^{(s)} \\ \mathbf{a} \\ \mathbf{e}^{+} \\ \mathbf{e}^{-} \end{bmatrix} \hat{=} \mathbf{Bc} \quad , \quad \text{s.t.} \ \ \mathbf{c} \succcurlyeq 0 \quad , \qquad (16)$$

where $\mathbf{a}^{(s)}$ is the concatenation of $\mathbf{a}_1^{(s)}, \mathbf{a}_2^{(s)}, \cdots, \mathbf{a}_k^{(s)}$ such that each $\mathbf{a}_i^{(s)}$ indicates the linear coefficients of templates from class i. To achieve the sparse representation for $\mathbf{c}$, we use the same ℓ_1 regularized least square as (11).

Now for frame t, given state set $\mathcal{X} = \{\mathbf{x}_t^1, \mathbf{x}_t^2, \ldots, \mathbf{x}_t^{n_p}\}$ and target set $\mathcal{Y} = \{\mathbf{y}_t^1, \mathbf{y}_t^2, \ldots, \mathbf{y}_t^{n_p}\}$, the tracking result $\widehat{\mathbf{y}}$ and its class label $\widehat{o}$ can be found by

$$(\widehat{\mathbf{y}}, \widehat{o}) = \arg\max_{\mathbf{y} \in \mathcal{Y}, 1 \le o \le k} \prod_{j=1}^{d} \mathbb{N}((\mathbf{y} - \mathbf{T}_o^{(s)} \mathbf{a}_o^{(s)} - \mathbf{Ta})(j); 0, \sigma^2), \ (17)$$

Since all the coefficients from the static templates are nonzero, they tend to have large values if their corresponding object has the same class as the target [43]. The recognition score of $p(\mathbf{y}|\mathbf{x}, o)$ is then defined accordingly by the sum of the coefficients of the static templates for class o, and normalized so that the sum of the scores across all the classes is 1.

$$p(\mathbf{y}|\mathbf{x}, o) \propto \sum\nolimits_{j=1,\ldots,n_o^{(s)}} a_{o,j}^{(s)} \qquad (18)$$

where $n_o^{(s)}$ is the number of static templates for class o.

The estimated sample $(\widehat{\mathbf{y}}, \widehat{o})$ implicitly encodes the tracking result (best target position) and recognition result (o^{th} object class). The static template set $\mathbf{T}^{(s)}$ consists of the templates from k classes which are obtained from the training video sequences.

As illustrated in Alg. 2, the weights $\mathbf{w}$ are only assigned to the dynamic template set $\mathbf{T}$ which are dynamically updated over time during the tracking. No weights are assigned to the static template set $\mathbf{T}^{(s)}$ since the static templates are used for recognition and not updated over time. The template update scheme is defined the same as Alg. 1. The norm of the dynamic template set is updated over time according to the template update scheme, while the norm of the static template set is initialized to be $1/n_d$, and kept constant during the whole tracking process.

Algorithm 2 Simultaneous tracking and recognition

```
1:  **Input**: a matrix of dynamic, static and trivial templates,
    $\mathbf{B} = [\mathbf{T}^{(s)}, \mathbf{T}, \mathbf{I}, -\mathbf{I}]$
2:  Uniformly initialize the weight $\mathbf{w}$ to $1/n_d$ for the dynamic
    templates, where $n_d$ is the number of dynamic templates
3:  Uniformly initialize the recognition probability
      $p_o(0) \leftarrow 1/k$, $o = 1, ..., k$.
4:  The norm of the templates in $\mathbf{T}$ and $\mathbf{T}^{(s)}$ is set to be $1/n_d$
5:  for $t = 1$ to number of frames do
6:     for $i = 1$ to number of samples do
7:        Draw $\epsilon_i$ from a Gaussian distribution
8:        Construct the sample state $\mathbf{x}_t^i \leftarrow \mathbf{x}_{t-1}^i + \nu_{t-1} + \epsilon_i$ in
          set $\mathcal{X}$, where $\nu_{t-1}$ is the velocity vector
9:        Compute the transformed candidate $\mathbf{y}_t^i$ from $\mathbf{x}_t^i$
10:       Solve the $\ell_1$ minimization problem (16) for each $\mathbf{y}_t^i$
11:       Update the sample weight $w_t^i \leftarrow p(\mathbf{y}_t^i|\mathbf{x}_t^i)$
12:    end for
13:    Find the current target $\widehat{\mathbf{y}}$ and its class $\widehat{o}$ by (17)
14:    Calculate the recognition score $p(\widehat{\mathbf{y}}|o)$ for each class $o$
15:    Compute the accumulated recognition probability
         $p_o(t) \leftarrow p_o(t-1) \times p(\widehat{\mathbf{y}}|o)$, $o = 1, ..., k$
16:    Normalize $p_o(t)$ across $o$ such that $\sum_{o=1}^{k} p_o(t) = 1$
17:    Update templates
18: end for
```

6 Experiments

6.1 Tracking Experiments

We implemented the proposed approach in MATLAB and evaluated the performance on numerous video sequences. The videos were recorded in indoor and outdoor environments in different formats (color, grayscale, and IR) where the targets underwent lighting and scale changes, out-of-plane rotation, and occlusion. The template sizes are 12×15, 15×12, 10×18, or 18×10 depending on the width to height ratio for the target image in the first frame. The number of previous velocities used for averaging is 3, and the variances of affine parameters ϵ is set to $(0.06, 0.01, 0.01, 0.06, 4, 4)$ for all experiments except the two long sequences in Fig. 8 in which it is set to be $(0.06, 0.01, 0.01, 0.06, 8, 8)$. The variance σ of observation probability is 0.0001. The similarity function threshold τ is set to be 0.5. The number of particles used for our method and AAPF is 300 for all experiments except for the two long sequences where it is 800. The number of target templates 10 is a trade-off between computational efficiency and effectiveness of modeling fast target appearance changes. In all cases, the initial position of the target was selected manually.

6.1.1 Experimental Results

The first test sequence shows a moving animal doll and presents challenging pose, lighting, and scale changes. It is from http://www.cs.toronto.edu/~dross/ivt/. Fig. 4 shows the tracking results using our proposed method, where the first row of each panel shows the tracked target which is enclosed by a parallelogram. Five images on the second row from left to right are tracked target image patch, reconstructed and residue image using target templates, reconstructed and residue image using both target and trivial templates, respectively. The third and fourth rows show the ten target templates which are updated dynamically over time during the tracking. We can see that more and more template images are replaced with the tracking results as tracking proceeds. The newly added templates are added since the old templates are no longer able to capture the variation of the moving target and provide the updated appearance of the target so the tracker will not drift. The frame indices are 24, 48, 158, 273, 383, 465, 515, 606 from left to right. In the first 200 frames (first four frames on the first row), the templates are not changed since the target undergoes out-of-plane rotation and translation.

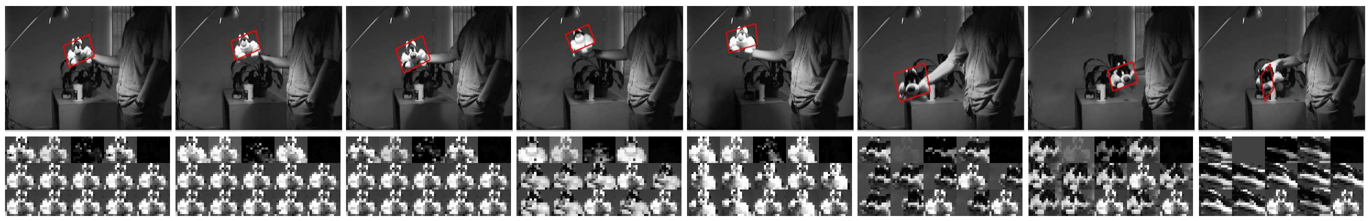

Fig. 4. An animal doll moves with significant lighting, scale, and pose changes. The first row of each panel shows the tracked target which is enclosed with a parallelogram. The second row shows (from left to right) the tracked target image patch, reconstructed and residue images using target templates, reconstructed and residue images using both target and trivial templates. The third and fourth rows show the ten target templates. Same configurations are used in Figures 5, 6, and 7.

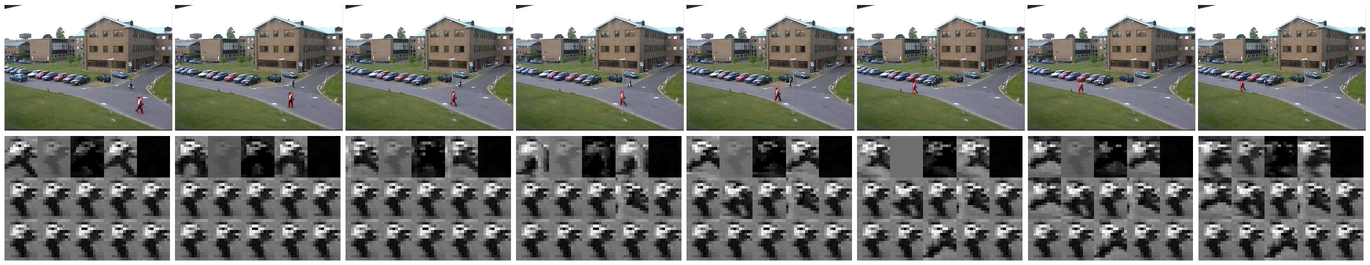

Fig. 5. A person walks passing the pole and high grasses with significant body movements and occlusion.

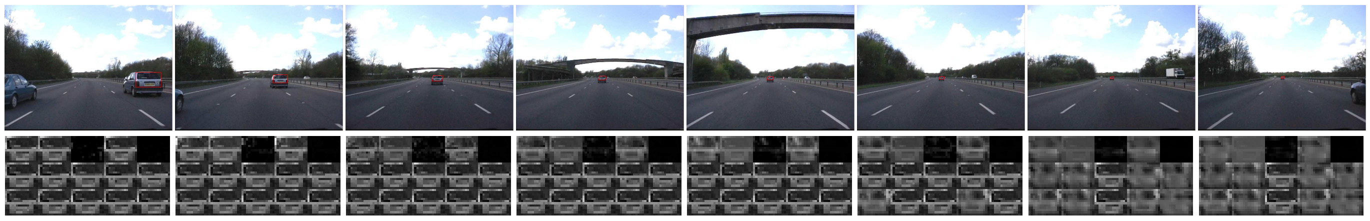

Fig. 6. A car moves fast with significant scale changes.

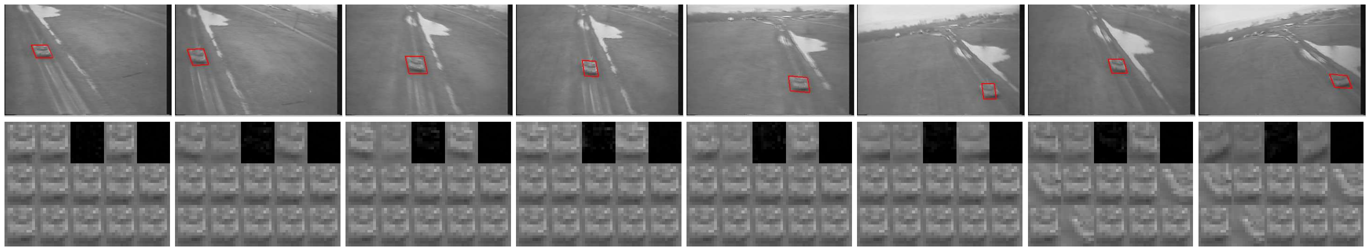

Fig. 7. A moving tank with significant motion blur in the image and background clutter.

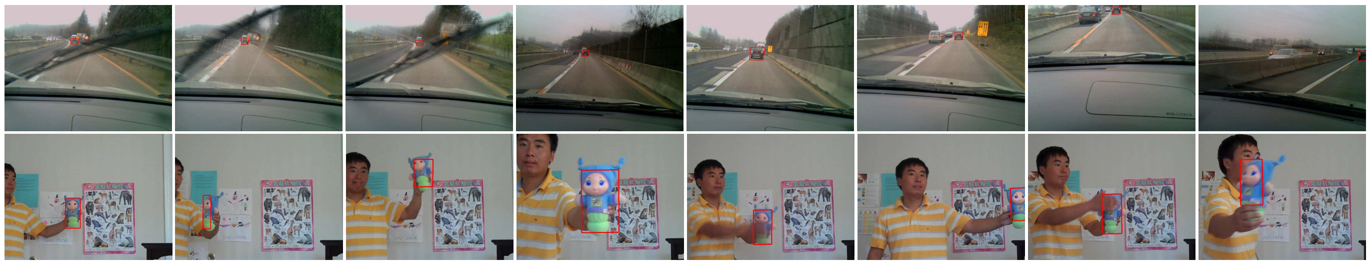

Fig. 8. Tracking results on two long sequences. Top: a moving vehicle on the freeway, with video taken in a car from back. There is significant occlusion in the sequence. Bottom: a doll is moving in the hand. There are pose and scale changes and occlusions in the sequence.

The appearance does not change much given the motion is modeled by affine transformation. After 200 frames, the target is moving farther and closer to the indoor light, which causes the appearance of the target changes dramatically. Therefore, more and more tracking results are added to capture the appearance changes of the target. Our tracker tracks the target well throughout the whole probe sequence.

The second test sequence is obtained from PETS 2001 benchmark dataset http://www.cvg.cs.rdg.ac.uk/PETS2001/. Some samples of the tracking results are shown in Fig. 5. The frame indices are 1442, 1479, 1495, 1498, 1535, 1628, 1640, 1668 from left to right. It shows a person walking from the right bottom corner of the image to the left. The person is small in the image and undergoes deformable motion when he walks past a pole (frame 1495, 1498, 1535) and long grasses (frame 1628, 1640, 1668). Since the person is thin in the image (the width of the target is small), the pole causes significant occlusion on the target. The template set gradually adds the tracking results as the person walks with hands swinging and legs spreading to maintain the variation of the template set. Our tracker tracks well even through the significant motion and occlusion. Note that a corrupted template can be included in the template set (e.g., the fourth template of the fourth frame in Fig. 5). However, the tracker still tracks the person properly because there are still other templates that are not contaminated.

The third test sequence contains a car moving very fast from close to far away. It has been used in [47] as "car" sequence. Some samples of the tracking results are shown in Fig. 6. It shows significant scale changes and fast motion. When the car starts pulling away, it becomes smaller and smaller, and harder to track since the target image is becoming smaller and gradually merges into the background. Even people have a hard time figuring out the precise location of the target. The frame indices are 547, 596, 634, 684, 723, 810, 984, 1061 from left to right. Our tracker follows the car well throughout the sequence. The scale changes on the width and height go up to as much as 10 and 6 times.

The fourth test sequence shows a tank moving on the ground and is captured by a moving camera. It has been used in [47] as "tank" sequence. Some samples of the tracking results are shown in Fig. 7. The frame indices are 641, 671, 689, 693, 716, 782, 884, 927 from left to right. There is significant motion blur in the image and background clutter. The tank looks very similar to the environment around it, and the tracker very easily gets stuck to the ground and drifts away. The sudden movement of the camera poses great challenges to the tracker since the movement of the target is very unpredictable and the target can go any direction from the current location. The random sampled particles allow the tracker to easily find the target's next movement. Therefore, our tracker locks on the target very well throughout the whole probe sequence.

Long term sequence tracking has recently drawn many researchers' attention [23] due to the challenges and practical applications. We test the proposed method on two long sequences. The first long sequence is a long-term sequence which runs more than 2000 frames used in [23]. The tracking results are shown in the top row of Fig. 8. The frame indices are 275, 276, 369, 774, 1059, 1508, 1925, and 2717. From frame 275 to 369, the windshield fluid is ejected on the windshield which causes blur on the target, and the windshield wiper is wiping through the windshield causing occlusion on the target. There is scale change in frame 1059 and the target moves out of the frame after frame 2717.

The second long sequence tested is taken by a hand held camcorder. Some samples of the tracking results are shown in the bottom row of Fig. 8. The frame indices are 170, 333, 750, 928, 1713, 2356, 2636, and 2938. It shows the tracking capability of our method under scale and pose changes and occlusions.

6.1.2 *Qualitative Comparison*

In our experiments, we compare the tracking results of our proposed method with the standard mean shift (MS) tracker [11], the covariance (CV) tracker [37], the appearance adaptive particle filter (AAPF) tracker [47], and the ensemble (ES) tracker [2] on five sequences. MS is based on the implemented function in OpenCV. We use the software of the author for testing AAPF. CV and ES are implemented according to the author's paper. The implementations are all parametrized according to the original papers with some tuning. The first two videos consist of 8-bit gray scale images while the last three are composed of 24-bit color images.

The first test sequence "pktest02" is an infrared (IR) image sequence from the VIVID benchmark dataset [41]. Some samples of the final tracking results are demonstrated in Fig. 9. Six representative frames of the video sequence are shown, with indices 1117, 1157, 1173, 1254, 1273, and 1386. The target-to-background contrast is very low and the noise level is high for these IR frames. From Fig. 9, we see that our tracker is capable of tracking the object all the time even with severe occlusions by the trees on the roadside. In comparison, MS and ES trackers lock onto the car behind the target starting from the fourth index frame (true target location is shown in a green rectangle). They keep tracking it in the rest of the sequences and are unable to recover it. CV tracker fails to track the target in the fourth index frame, similar to MS tracker, but it is able to recover the failure and track the target properly from the fifth index frame and throughout the rest of the sequence. In comparison, our proposed method avoids this problem and is effective under low contrast and in noisy situations. AAPF achieves similar results to our method.

The second test sequence "car4" is obtained from http://www.cs.toronto.edu/~dross/ivt/. The vehicle undergoes drastic illumination changes as it passes beneath a bridge and under trees. Some samples of the final tracking results are demonstrated in Fig. 10. The frame indices are 181, 196, 233, 280, 308, and 315. MS tracker loses the target very quickly and goes out of range from the fourth index frame. CV tracker loses the target after tracking it for a while and gets stuck on the background. ES tracker tracks the car reasonably well given small scale changes in the sequence. Our tracker and AAPF are able to track the target well through the drastic illumination changes.

The third test sequence "oneleaveshop" is obtained from http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/. In this video, the background color is similar to the color of the woman's trousers, and the man's shirt and pants have a similar color to the woman's coat. In addition, the woman undergoes partial occlusion. Some result frames are given in

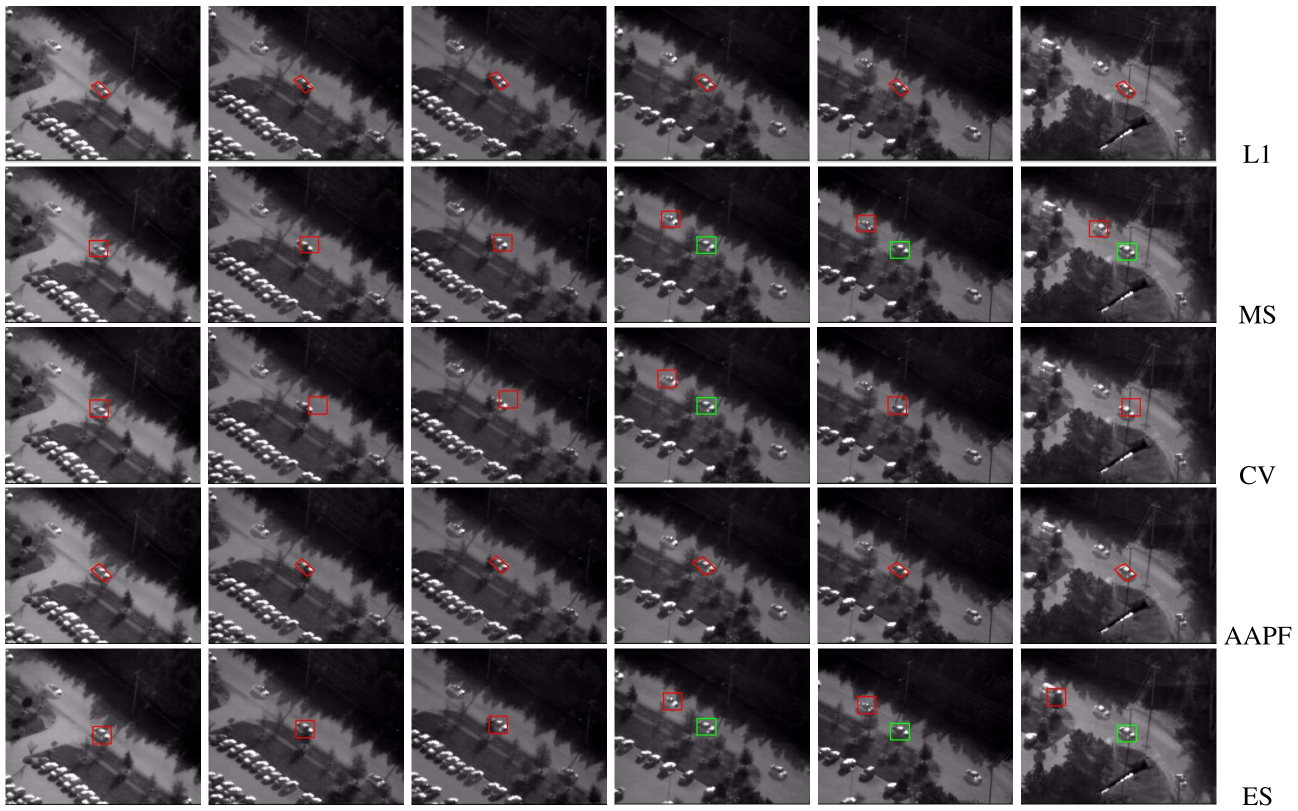

Fig. 9. Tracking results of the sequence “pktest02” with severe occlusion; tracker names are listed on the right side (same for the rest of the figures). For better illustration, we draw a green rectangle around the true target when a tracker fails.

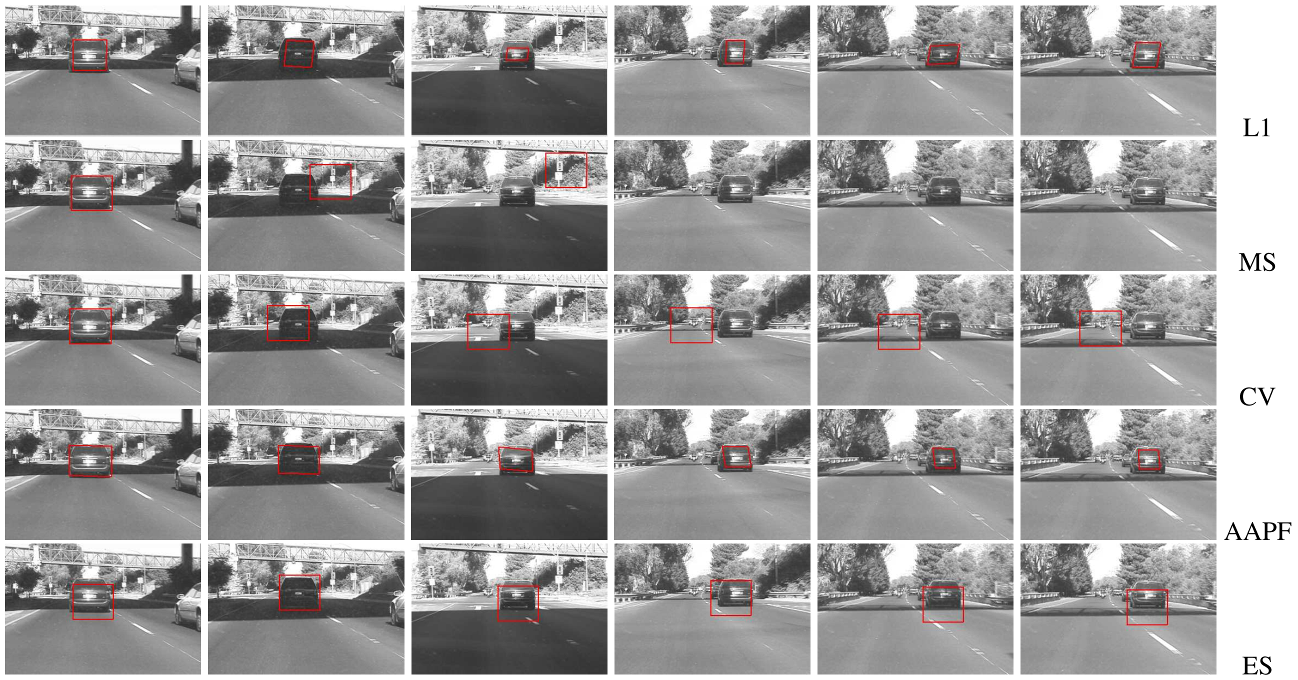

Fig. 10. Tracking results of the sequence “car4” over representative frames with drastic illumination changes.

Fig. 11. The frame indices are 137, 183, 207, 225, 245, and 271. It can be observed that MS, CV, and AAPF trackers start tracking the man when the woman is partially occluded at the third index frame, and are not able to recover the failure after that. ES tracker drifts away from the target very quickly and goes out of the bounds of the image. Compared with other trackers, our tracker is more robust to the occlusion, which makes the target model not easily degraded by the outliers.

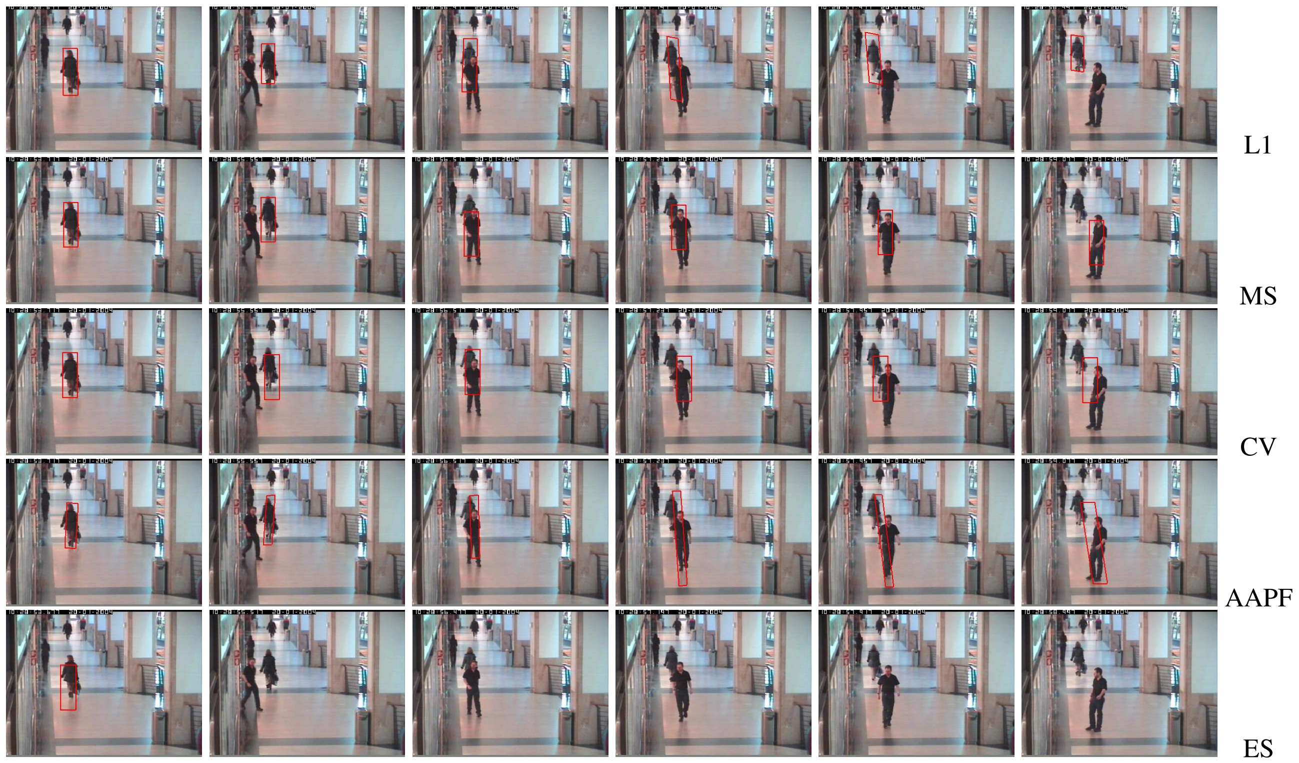

Fig. 11. Tracking results of the sequence “oneleaveshop” over frames with partial occlusion and background clutter.

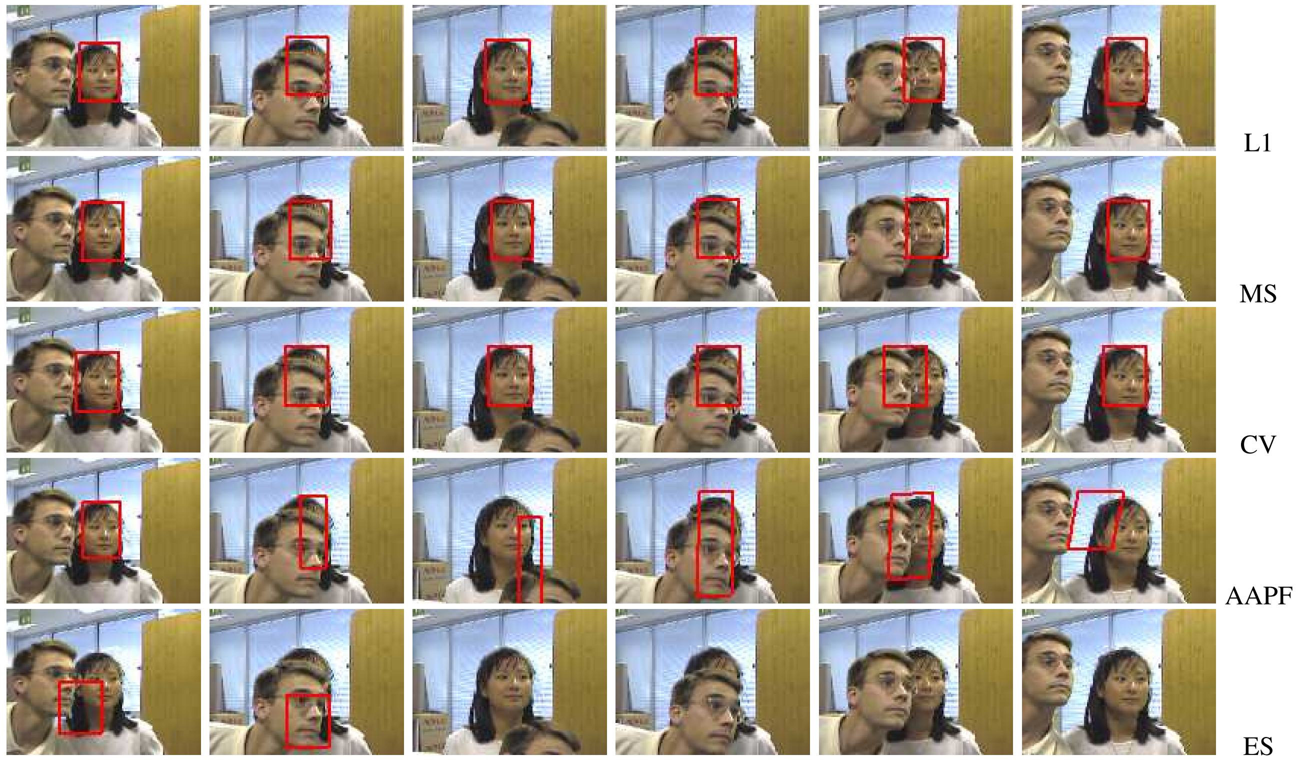

Fig. 12. Tracking results of the sequence “birch_seq_mb” over representative frames with severe occlusion.

The fourth test sequence “birch_seq_mb” is obtained from http://vision.stanford.edu/˜birch/headtracker/seq/. We show some samples of the tracking results for the trackers in Fig. 12. The six representative frame indices are 422, 436, 448, 459, 466, and 474. The man’s face is passing in front of the woman’s face. Again, our method obtains good tracking results. The MS and CV trackers also obtain good results. The AAPF tracker drifts when the severe occlusion happens in the second index frame. The ES tracker loses the target when the man’s head occludes the girl’s face.

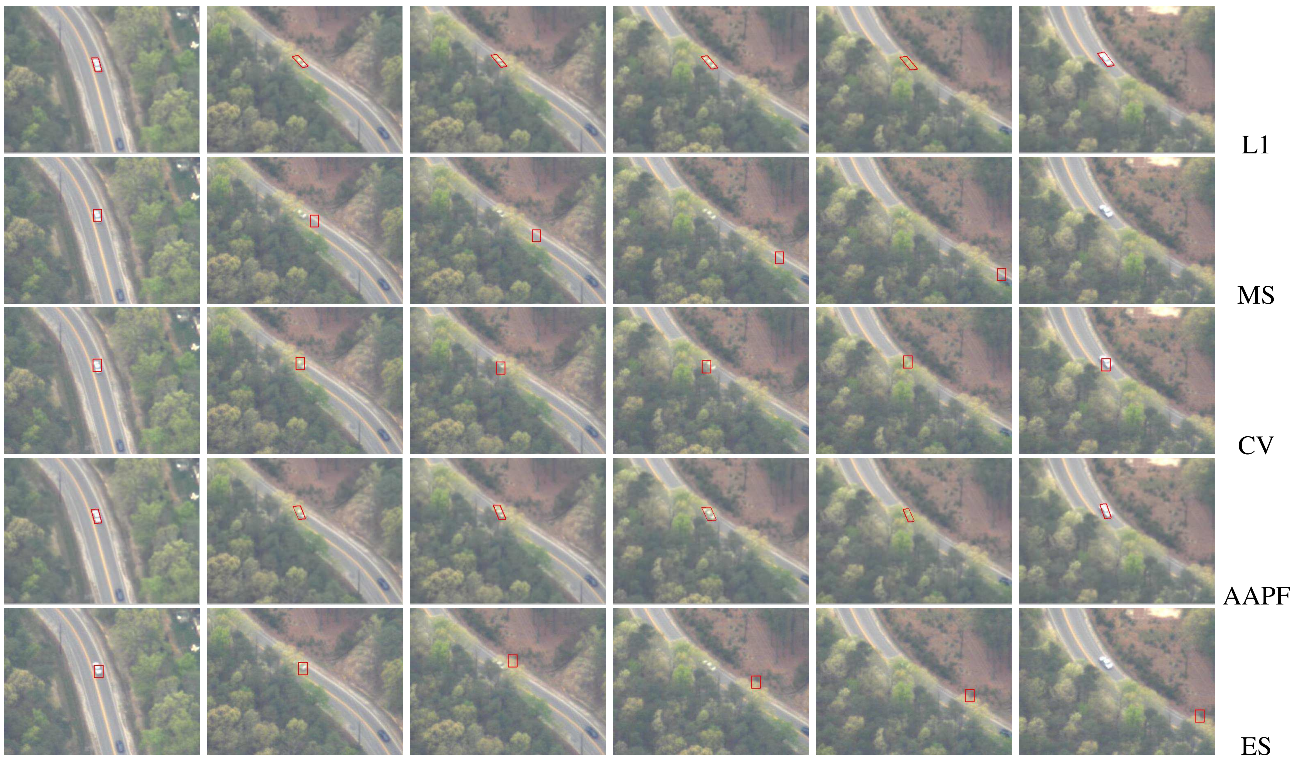

Fig. 13. Tracking results of the sequence “V4V1_7_7” over frames with occlusion and large pose variation.

The fifth test sequence “V4V1_7_7” is an airborne car video [41]. The car is running on a curved road and passing beneath the trees. It undergoes heavy occlusions and large pose changes. We show some samples of the tracking results for the trackers in Fig. 13. The six representative frame indices are 215, 331, 348, 375, 393, and 421. MS tracker loses the target very quickly and goes out of range in the sixth frame. ES tracker drifts away from the target when it goes under the trees along the road and is heavily occluded. Although CV and AAPF can track the target, they do not locate the target well. Our tracker tracks the target very well throughout the whole sequence.

6.1.3 *Quantitative comparison*

To quantitatively compare robustness under challenging conditions, we manually labeled the ground truth of the sequences “pktest02”, “car4”, “oneleavesshop”, “birch_seq_mb”, and “V4V1_7_7” from the previous section for 300, 300, 150, 93, and 300 frames, respectively. The evaluation criteria of the tracking error are based on the relative position errors (in pixel) between the center of the tracking result and that of the ground truth. Ideally, the position differences should be around zero.

As shown in Fig. 14, the position differences of the results in our ℓ_1 tracker are much smaller than those of the other trackers. It demonstrates the advantage of our ℓ_1 tracker.

Our method is implemented in MATLAB, and takes about 2 seconds to process one frame in the above experimental setup. We expect our method to process 2 to 5 frames per second if the implementation is optimized.

Fig. 15. Static templates for four types of vehicles.

6.2 Tracking and Recognition Experiments

In this section, we apply the simultaneous tracking and recognition to the IR-based vehicle classification task. We test our method on the video sequences used in [35]. The static templates are obtained from some training sequences. Four different types of vehicles are used in the experiment. Fig. 15 shows the five static templates for each vehicle, named “bmp”, “m60”, “brdm”, and “wetting”, respectively.

Fig. 16 shows the tracking and classification results of the vehicle “wetting”. Fig. 16 (a)-(f) show the tracking results (top) and normalized recognition score at the current frame (bottom) for the index frames 552, 624, 675, 713, 740, 802, respectively. Fig. 16 (g) shows the normalized recognition score at each frame for 320 frames and (h) shows the accumulated recognition score for each vehicle from beginning to current frame (only first 20 frames are shown since the recognition converges afterwards). In Fig. 16 (b), the current frame is mistakenly classified as “m60”, but we are taking the accumulative score as the final classification result. The recognition score for “wetting” is gradually increasing to 1 as the process goes. Therefore, we obtain correct classification results on this vehicle. Similar to Fig. 16, Fig. 17 shows the tracking and classification results of the vehicle “bmp” for frames 78144, 78185, 78202, 78276, 78347, 78430, respectively. In Fig. 17 (a) and (c), the current frame is mistakenly classified as “wetting” and “brdm”.

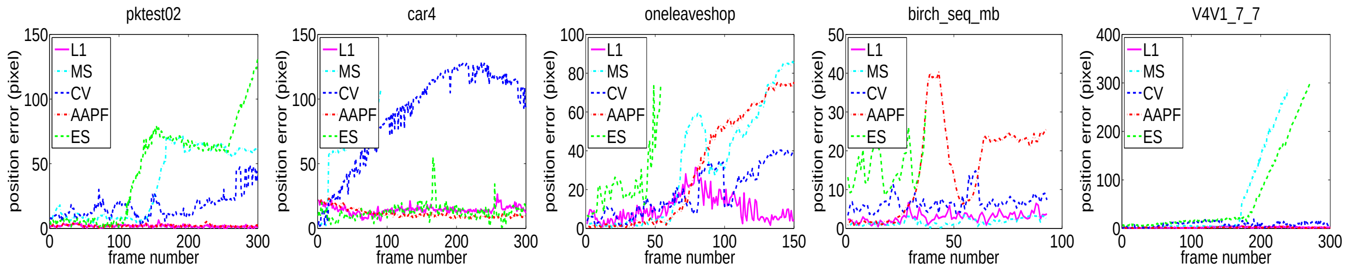

Fig. 14. Quantitative comparison of the trackers in terms of position errors (in pixel).

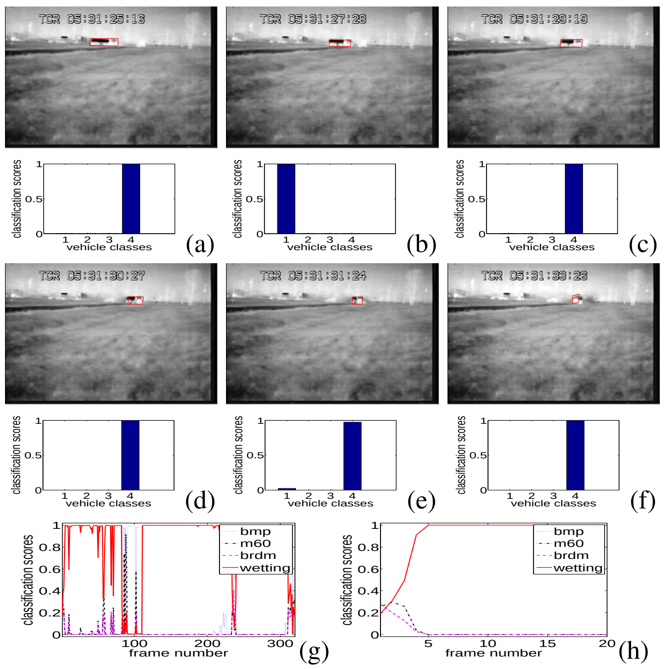

Fig. 16. Tracking and classification results of the vehicle "wetting". (a)-(f): tracking results (top) and corresponding normalized recognition scores (bottom). (g): probabilities of each vehicle at each frame. (h): accumulated probabilities for each vehicle from beginning to current frame (only first 20 frames are shown due to convergence).

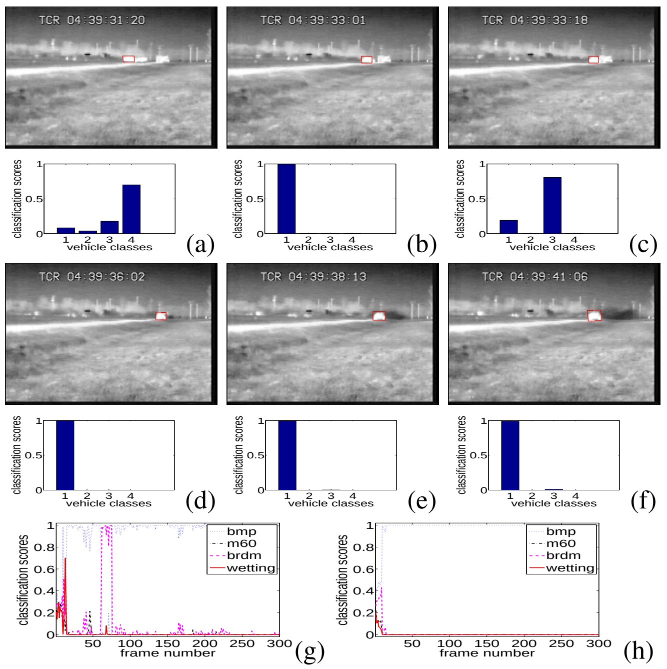

Fig. 17. Tracking and classification results of the vehicle "bmp". Subtitles for (a)-(g) are same as in Fig. 16.

6.3 Discussion

The experiments demonstrate the robust tracking performance of our algorithm. However, our tracker can fail when there is very large pose change or the target moves completely out of the frame and reappears. Fig. 18 shows two failure cases of our tracker. In the top row, the woman's head is rotating out-of-plane for 180 degrees and the target varies from frontal face in left image to back of her head in middle image. Tracker drifts away from the target after the rotation is complete and the woman's frontal face reappears in the right image. In the bottom row of Fig. 18, the target moves completely out of the frame in the middle image. The tracker is tracking the background without knowing it. When the target reappears in the right image, the tracker cannot recover after losing track of the target.

Another interesting situation is when the tracking target is occluded by another visually similar object. The performance depends on the degree of the similarity between the target and the occluder, and on the robustness of motion estimation. Theoretically, it is possible that the tracker will lock to the occluding object. We expect future investigation to address this issue.

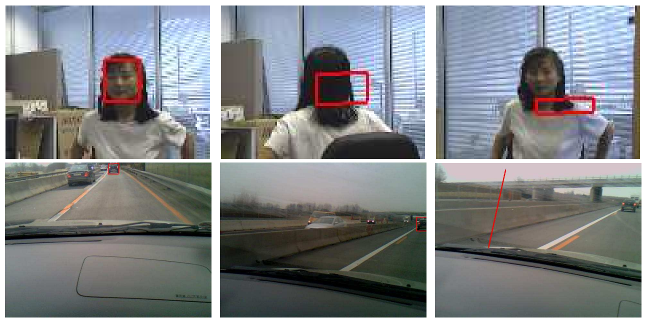

Fig. 18. Two failed tracking cases.

7 Conclusion

In this paper, we propose using sparse representation for robust visual tracking. We model tracking as a sparse approximation problem and solve it through an ℓ_1-regularized least squares approach. For further robustness, we introduce nonnegativity constraints and dynamic template update in our approach. In thorough experiments involving numerous challenging sequences and four other state-of-the-art trackers, our approach demonstrates very promising performance. We also extend our work to simultaneous tracking and recognition and apply it to IR-based vehicle classification. The experimental results demonstrate clearly the effectiveness of our proposed method.

Acknowledgment We thank Drs. David Jacobs and Albert Montillo for proofreading the manuscript, and anonymous reviewers for valuable suggestions. This work is supported in part by NSF Grants IIS-0916624 and IIS-1049032.

References

[1] M. Andriluka, S. Roth, and B. Schiele. "People-tracking-by-detection and people-detection-by-tracking", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1–8, 2008.

[2] S. Avidan. "Ensemble Tracking", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 494–501, 2005.

[3] B. Babenko, M. Yang, and S. Belongie. "Visual Tracking with Online Multiple Instance Learning", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2009.

[4] S. Baker and I. Matthews. "Lucas-kanade 20 years on: A unifying framework", *Int'l Journal of Computer Vision*, 56:221–255, 2004.

[5] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, L. V. Gool. "Robust tracking-by-detection using a detector confidence particle filter", in *Proc. Int'l Conf. on Computer Vision*, 2009.

[6] M. J. Black and A. D. Jepson. "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation", *Int'l Journal of Computer Vision*, 26:63-84, 1998.

[7] Y. Cai, N. Freitas and J. Little. "Robust Visual Tracking for Multiple Targets", in *Proc. European Conf. on Computer Vision*, 107–118, 2006.

[8] E. Candès, J. Romberg, and T. Tao. "Stable signal recovery from incomplete and inaccurate measurements", *Comm. on Pure and Applied Math*, 59(8):1207-1223, 2006.

[9] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa. "Compressive Sensing for Background Subtraction", in *Proc. European Conf. on Computer Vision*, 2008.

[10] R. T. Collins and Y. Liu. "On-Line Selection of Discriminative Tracking Features", in *Proc. Int'l Conf. on Computer Vision*, 346–352, 2003.

[11] D. Comaniciu, V. Ramesh, and P. Meer. "Kernel-based object tracking", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 25:564–577, 2003.

[12] l1_ls: Simple Matlab Solver for l1-regularized Least Squares Problems, [online] *http://www.stanford.edu/~boyd/l1_ls/*.

[13] D. Donoho. "Compressed Sensing", *IEEE Trans. Information Theory*, 52(4):1289-1306, 2006.

[14] A. Doucet, N. de Freitas, and N. Gordon. "Sequential Monte Carlo Methods in Practice". *Springer-Verlag*, 2001, New York.

[15] G.J. Edwards, C.J. Taylor, and T.F. Cootes. "Improving Identification Performance by Integrating Evidence from Sequences", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1:486-491, 1999.

[16] J. Gu, S. Nayar, E. Grinspun, P. Belhumeur, and R. Ramamoorthi. "Compressive Structured Light for Recovering Inhomogeneous Participating Media", in *Proc. European Conf. on Computer Vision*, 2008.

[17] G. D. Hager and P. N. Belhumeur. "Efficient region tracking with parametric models of geometry and illumination", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 20:1025-1039, 1998.

[18] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman. "Visual tracking using learned subspaces", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 782-789, 2004.

[19] K. Hotta. "Adaptive weighting of local classifiers by particle filters for robust tracking", in *Pattern Recognition*, 42(5):619-628, 2009.

[20] J. Huang, X. Huang, and D. Metaxas. "Learning With Dynamic Group Sparsity", in *Proc. Int'l Conf. on Computer Vision*, 2009.

[21] M. Isard and A. Blake. "Condensation - conditional density propagation for visual tracking", *Int'l Journal of Computer Vision*, 29:5-28, 1998.

[22] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. "Robust online appearance models for visual tracking", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 25:1296-1311, 2003.

[23] Z. Kalal, J. Matas, and K. Mikolajczyk. "P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2010.

[24] T. Kaneko and O. Hori. "Feature Selection for Reliable Tracking using Template Matching", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 796-802, 2003.

[25] Z. Khan, T. Balch, and F. Dellaert. "A Rao-Blackwellized particle filter for EigenTracking", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 980-986, 2004.

[26] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. "A method for large-scale ℓ_1-regularized least squares", *IEEE Journal on Selected Topics in Signal Processing*, 1(4):606-617, 2007.

[27] M. Kristan, S. Kovacic, A. Leonardis, and J. Pers. "A Two-Stage Dynamic Model for Visual Tracking", *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, PP(99):1-16, 2010.

[28] K.-C. Lee, and D. Kriegman. "Online Learning of Probabilistic Appearance Manifolds for Video-based Recognition and Tracking", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 852-859, 2005.

[29] X. Liu, and T. Chen."Video-Based Face Recognition Using Adaptive Hidden Markov Models", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1:340-345, 2003.

[30] B. Lucas and T. Kanade. "An iterative image registration technique with an application to stereo vision", *Int'l Joint Conf. on Artificial Intel.*, 674-679, 1981.

[31] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. "Discriminative learned dictionaries for local image analysis", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.

[32] I. Matthews, T. Ishikawa, and S. Baker. "The template update problem", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 810-815, 2004.

[33] X. Mei, H. Ling, and D.W. Jacobs. "Sparse Representation of Cast Shadows via ℓ_1-Regularized Least Squares", in *Proc. Int'l Conf. on Computer Vision*, 2009.

[34] X. Mei and H. Ling. "Robust Visual Tracking using ℓ_1 Minimization", in *Proc. Int'l Conf. on Computer Vision*, 2009.

[35] X. Mei, S. K. Zhou and H. Wu. "Integrated Detection, Tracking and Recognition for IR Video-Based Vehicle Classification", *IEEE Int'l Conf. on Acoustics, Speech and Signal Processing*, 745-748, 2006.

[36] J. Mooser, Q. Wang, S. You, and U. Neumann. "Fast Simultaneous Tracking and Recognition Using Incremental Keypoint Matching", *Int'l Symp. on 3D Data Processing, Visualization and Transmission*, 2008.

[37] F. Porikli, O. Tuzel and P. Meer. "Covariance tracking using model update based on lie algebra", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 728-735, 2006.

[38] D. A. Ross, J. Lim, R. Lin and M. Yang. "Incremental learning for robust visual tracking", *Int'l Journal of Computer Vision*, 77:125-141, 2008.

[39] J. Sakagaito, and T. Wada. "Nearest first traversing graph for simultaneous object tracking and recognition", in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1-7, 2007.

[40] H. Sidenbladh, M. J. Black, and D. J. Fleet. "Stochastic tracking of 3d human figures using 2d image motion", *European Conf. on Computer Vision*, 2:702-718, Copenhagen, Denmark, 2002.

[41] VIVID database. [online] https://www.sdms.afrl.af.mil/request/data_request.php#vivid

[42] O. Williams, A. Blake, and R. Cipolla. "Sparse Bayesian Learning for Efficient visual Tracking", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 27:1292-1304, 2005.

[43] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. "Robust Face Recognition via Sparse Representation", *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 31(1):210-227, 2009.

[44] B. Wu and R. Nevatia. "Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors", in *Int'l Journal of Computer Vision*, 75(2):247-266, 2007.

[45] A. Yilmaz, O. Javed, and M. Shah. 'Object tracking: A survey". *ACM Comput. Survey*, 38(4), 2006.

[46] Q. Yu, T. B. Dinh and G. Medioni. "Online tracking and reacquistion using co-trained generative and discriminative trackers", in *Proc. European Conf. on Computer Vision*, 678-691, 2008.

[47] S. K. Zhou, R. Chellappa, and B. Moghaddam. "Visual tracking and recognition using appearance-adaptive models in particle filters", *IEEE Trans. Image Processing*, 11:1491-1506, 2004.