Efficient Minimum Error Bounded Particle Resampling L1 Tracker with Occlusion Detection

Xue Mei, *Member, IEEE,* Haibin Ling, *Member, IEEE,* Yi Wu, *Member, IEEE,* Erik Blasch, *Senior Member, IEEE,* and Li Bai, *Senior Member, IEEE*

***Abstract*—Recently, sparse representation has been applied to visual tracking to find the target with the minimum reconstruction error from a target template subspace. Though effective, these L1 trackers require high computational costs due to numerous calculations for ℓ_1 minimization. In addition, the inherent occlusion insensitivity of the ℓ_1 minimization has not been fully characterized. In this paper, we propose an efficient L1 tracker, named *Bounded Particle Resampling (BPR)-L1 tracker*, with a minimum error bound and occlusion detection. First, the minimum error bound is calculated from a linear least squares equation and serves as a guide for particle resampling in a particle filter framework. Most of insignificant samples are removed before solving the computationally expensive ℓ_1 minimization in a two step testing. The first step, named τ *testing*, compares the sample observation likelihood to an ordered set of thresholds to remove insignificant samples without loss of resampling precision. The second step, named *max testing*, identifies the largest sample probability relative to the target to further remove insignificant samples without altering the tracking result of the current frame. Though sacrificing minimal precision during resampling, max testing achieves significant speed up on top of τ testing. The BPR-L1 technique can also be beneficial to other trackers that have minimum error bounds in a particle filter framework, especially for trackers based on sparse representations. After the error-bound calculation, BPR-L1 performs occlusion detection by investigating the trivial coefficients in the ℓ_1 minimization. These coefficients, by design, contain rich information about image corruptions including occlusion. Detected occlusions are then used to enhance the template updating. For evaluation, we conducted experiments on three video applications: biometrics (head movement, hand holding object, singers on stage), pedestrians (urban travel, hallway monitoring), and cars in traffic (wide area motion imagery, ground-mounted perspectives). The proposed BPR-L1 method demonstrates excellent performance as compared with nine state-of-the-art trackers on eleven challenging benchmark sequences.**

***Index Terms*—Visual tracking, sparse representation, compressive sensing, particle filter, ℓ_1 minimization, occlusion detection, minimum error bound.**

I. Introduction

Visual tracking is an important topic for applications such as security and surveillance, vehicle navigation, human computer interaction, and so on. The challenges in designing a robust visual tracking algorithm in a dynamic environment are caused by the presence of noise, occlusions, varying viewpoints, background clutter and illumination changes. A thorough review can be found in [41].

Recently, sparse representation [7], [11] has been successfully applied to visual tracking [14], [17], [22], [25]–[27], [30], [31], [35], [38], [44]–[46]. In these methods, the tracking problem is formulated as finding a sparse representation of the target candidate using templates. The advantage of using the sparse representation lies in the robustness to a wide range of image corruptions, especially occlusion. The results show good performance, however, at a computational expense of the ℓ_1 minimization. Furthermore, the target states are estimated in a particle filter framework and the computational cost grows proportionally as the number of particle samples increases. The large computational cost prevents the tracker from being used in a real time system such as real time surveillance and security operations. Furthermore, the rich information captured in approximation coefficients has not been utilized for occlusion analysis. For example, a gradual occlusion may cause model drifting in the template set.

Inspired by the aforementioned research, we propose an efficient tracking algorithm with a minimum error bound and occlusion detection. Our first contribution is to improve the run time efficiency of the L1 tracker by using an error bound derived from the least squares computation. Specifically, we observe that the computationally expensive reconstruction error in the sparsely constrained ℓ_1 minimization is lower-bounded by the least squares reconstruction error, which can be calculated efficiently. The observation motivates us to design a *Bounded Particle Resampling* (BPR) algorithm, which greatly boosts the speed of the tracking algorithm. Specifically, the probability of a tracking sample is calculated in two stages of which most samples are filtered out before solving the ℓ_1 minimization. In the first stage, the sample is reconstructed by simply projecting it onto the target template subspace. The reconstruction is solved through a linear least squares equation that runs several orders faster than a typical ℓ_1 minimization. In the second stage, only dynamically selected samples that pass the two step testings from previous stage are processed through ℓ_1 minimization. By this two stage reconstruction, the computational cost is greatly reduced and the proposed tracker runs much faster than our previous L1 tracker [30]. The BPR technique can also be beneficial to other trackers that have minimum error bounds in a particle filter framework, especially for trackers based on sparse representations. For example, the BPR technique can be applied to [25], [26], [38] to further improve the speed of their trackers.

Our second contribution is an occlusion detection method that investigates the reconstruction coefficients to improve the template update procedure. The ℓ_1 minimization based reconstruction used in our previous method [30] is known to capture occlusion information, which has been previously used for face recognition [36]. While this property enables

tracking occluded targets, it also induces risks by introducing the occluded target information into the template set and potentially causing tracking failures. To prevent the wrong information from contaminating the template set, we introduce an occlusion detection method. The idea is to first build an occlusion map from the *trivial coefficients*, which indicate pixel-wise image contamination in a given candidate. The occlusion map is then used for occlusion detection and a candidate with detected occlusion will not be used to update the template set.

For evaluation, the proposed BPR-L1 tracker is tested on eleven benchmark sequences involving various challenges such as occlusion and illumination changes. The sequences come from three different application scenarios: biometrics (head movement, hand holding object, singers on stage), pedestrians (urban travel, subway monitoring), and cars in traffic (wide area motion imagery, ground-mounted perspectives). In all experiments, our method shows excellent effective (accuracy) and efficient (timeliness) performance in comparison with previously proposed trackers.

The remainder of the paper is organized as follows. The related work is explained in Section II. Section III presents the particle filter algorithm preliminaries. In Section IV-A, we briefly review the L1 tracker proposed in previous work and its limitations. Section IV-B derives the minimum error bound. The BPR-L1 technique with two step testings is described in Section IV-C. An occlusion detection method is proposed in Section V. Experimental results on both methods are reported in Section VI. Finally, we conclude the paper in Section VII.

II. Related Work

Visual tracking is an important topic in computer vision and it has been studied for decades. The visual tracking problem can be formulated in two different categories: generative and discriminative. Generative tracking methods use an appearance model to represent the target observations. Tracking is formulated as searching the target location that has the most similar appearance to the model. One example is using subspace models. The idea is that the images of the target lie in a low dimensional manifold. The appearance of the object is represented using an eigenspace [5], affine warps of learned linear subspaces [13], an incrementally learned low-dimensional subspace [33], the novel incremental tensor subspace [15], online appearance model [8], etc. Other popular tracking methods include mean shift tracker [10], covariance tracker [32], and frag tracker [1].

Discriminative tracking methods cast the tracking as a binary classification problem. Tracking is formulated as finding the target location that can best separate the target from the background. 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 pixels that belong to the background. The TLD tracker [19] uses P-N learning [18] to exploit the structure of the data and get feedback about the performance of the classifier. It is shown to be reliable in long sequence tracking. In [9], a confidence map is built by finding the most discriminative RGB color combination in each frame.

One intuitive way of improving discriminative and generative methods is to combine them together in a hybrid way. A hybrid approach that combines a generative model and a discriminative classifier captures appearance changes and allows object reacquisition after a total occlusion [43]. A three-level hierarchy combines the discriminative and generative models for tracking under the framework of sequential Bayesian learning in [24].

Many trackers have been proposed to compensate for occlusion induced track failures. Online multiple instance learning is used in [3] to achieve robustness to occlusions as well as other image corruptions. Global mode seeking detects the object after total occlusion and reinitializes the local tracker [42]. In [22], a novel algorithm is proposed to detect occlusion for visual tracking through learning with observation likelihoods and is combined with a L1 tracker [30]. Another example [6] uses image fusion to determine the best appearance model for discrimination and then a generative approach for dynamic target updates.

Particle Filter (PF) has been introduced for visual tracking [16] and has been a popular framework due to excellent performance for nonlinear target motion and flexibility to different object representations. While the use of more particle samples can improve track robustness, the computational load required by the particle filter also increases. Some authors have proposed to speed up the particle filter framework. In [40], the observation likelihood based on multiple features is computed in a coarse-to-fine manner so that the computation can quickly focus on the more promising regions. In [20], an efficient method is introduced for using subspace representations in a particle filter by applying Rao-Blackwellization to integrate out the subspace coefficients in the state vector. Fewer samples are needed since part of the state vector posterior is analytically calculated. In [47], it adjusts the number of particle samples based on the noise variance.

Sparse representation has recently been introduced for tracking in [30] and later exploited in [14], [22], [25]–[27], [38]. In [30], a tracking candidate is sparsely represented as a linear combination of target templates and trivial templates that only have one nonzero element. The sparse representation problem is solved through an ℓ_1 minimization problem with non-negativity constraints to solve the inverse intensity pattern problem during tracking. In [26] the group sparsity is integrated and very high dimensional image features are used for improving tracking robustness. In [38], a novel blur-driven tracker framework for tracking motion-blurred targets is proposed. It introduces a blur template subspace and track the target without performing deblurring. In [25], a real-time tracker is proposed by adopting dimensionality reduction and a customized Orthogonal Matching Pursuit (OMP) algorithm to accelerate the tracking. In [4], a very fast numerical solver is developed to solve the resulting ℓ_1 norm related minimization problem with guaranteed quadratic convergence based on the accelerated proximal gradient approach. In [27], a tracking algorithm with a static sparse dictionary and dynamic online updated basis distribution is developed. Our work is inspired by these studies, but we use an ℓ_2 error bound to improve efficiency and introduce an occlusion map for a reliable

template update. In [39], several fast ℓ_1 minimization methods have been reviewed and compared. The focus of this paper is to improve the speed by exploiting the structure of the particle filter and combining with the ℓ_2 bound. This technique is different from the efforts of finding fast ℓ_1 minimization algorithms such as [4], [25] and in fact the two can be combined. Therefore, we did not evaluate through all different ℓ_1 minimization algorithms.

Our work shares philosophies with works where the error bound is used to guide visual tracking. For example, in [28] a boosting error bound in a co-training framework is used to guide the novel tracker construction. However, the application of using an error bound in a sparse tracker has not been well explored. Furthermore, our goal is to use the error bound for speed up *without* sacrificing accuracy (τ testing) or *without* altering the tracking result (max testing). The τ testing is first described in a preliminary conference version of this work in [31]. In comparison, this paper introduces the new *max testing* for further improvement and involves more thorough evaluations.

III. Particle Filter

The L1 tracker proposed in [30] is formulated as finding a sparse representation in the template subspace. The representation is then used with the particle filter framework [16] for visual tracking. Specifically, for frame at time t, we denote $\mathbf{x}_t$ as the state variable describing the location and shape of a target. The tracking problem can be formulated as an estimation of the state probability $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, where $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_t\}$ represents the observations from previous t frames. The tracking proceeds using a two-stage Bayesian sequential estimation, which recursively updates the filtering distribution as

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

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

where $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ indicates the state transition probability, and $p(\mathbf{z}_t|\mathbf{x}_t)$ gives the observation likelihood of state $\mathbf{x}_t$. Direct calculation of the above distribution is practically intractable. Alternatively, the posterior $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is approximated by a set of N particle samples $\{\mathbf{x}_t^i\}_{i=1}^N$ with importance weights π_t^i. The samples are updated and resampled at each frame.

In the L1 tracker, the state variable $\mathbf{x}_t$ contains six parameters of the affine transformation. The state transition of $\mathbf{x}_t$ are modeled independently by a Gaussian distribution around the previous state $\mathbf{x}_{t-1}$, and N candidate samples are generated based on the state transition model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The observation model $p(\mathbf{z}_t|\mathbf{x}_t)$ reflects the similarity between a target candidate and the target templates, which is formulated using approximation error in the sparse representation.

For tracking at time t, the candidate with the maximum observation likelihood is chosen as the tracking result. The likelihood determines sample weights and importance resampling thresholds in the particle filter. A summary of the particle filter for L1 tracker is given in Algorithm 1.

Algorithm 1 Particle filter for L1 tracker

1: At $t = 0$, initialize samples $\mathbf{x}_0^i$, for $i = 1, 2, \cdots, N$
2: **for** $t = 1$ to number of frames **do**
3: **for** $i = 1$ to number of samples **do**
4: Draw sample $\mathbf{x}_t^i$ with respect to $p(\mathbf{x}_t|\mathbf{x}_{t-1})$
5: Prepare the observation $\mathbf{y}_t^i$ from $\mathbf{x}_t^i$
6: Calculate the observation probability $p(\mathbf{y}_t^i|\mathbf{x}_t^i)$
7: Resample with respect to $p(\mathbf{y}_t^i|\mathbf{x}_t^i)$, the number of times that $\mathbf{x}_t^i$ appears in the new set is $N * p(\mathbf{y}_t^i|\mathbf{x}_t^i)$
8: **end for**
9: **end for**

IV. Efficient L1 Tracker with Bounded Particle Resampling

In this section we will first review the original L1 tracker [30] that combines the sparse representation and the particle filter framework. Then we will present the least squares based minimum error bound, which can be more efficiently computed than the error calculated used in the L1 tracker through ℓ_1 minimization. After that, we propose the BPR-L1, using the BPR procedure to increase efficiency.

A. L1 Tracker with Sparse Representation

To model the observation likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$, a region corresponding to state $\mathbf{x}_t$ is first cropped from frame $\mathbf{z}_t$[1]. The region is then normalized and reshaped to a 1D vector $\mathbf{y}$, which is used as a target candidate.

The sparse representation of $\mathbf{y}$ is formulated as a minimum error reconstruction through a regularized ℓ_1 minimization with nonnegativity constraints

$$\min_{\mathbf{c}} \| \mathbf{Bc} - \mathbf{y} \|_2^2 + \lambda \| \mathbf{c} \|_1 \quad , \quad \text{s.t.} \quad \mathbf{c} \succcurlyeq 0 , \quad (3)$$

where $\mathbf{B} = [\mathbf{T}, \mathbf{I}, -\mathbf{I}]$ is an over-complete dictionary composed of the target template set $\mathbf{T}$, the positive and negative trivial template sets $\mathbf{I}$ and $-\mathbf{I}$. Each column in $\mathbf{T}$ is a target template generated by reshaping pixels of a candidate region into a column vector; and each column in the trivial template sets is a unit vector that has only one nonzero element. $\mathbf{c} = [\mathbf{a}^\top, \mathbf{e}^{+\top}, \mathbf{e}^{-\top}]^\top$ is composed of target coefficients $\mathbf{a}$ and positive and negative trivial coefficients $\mathbf{e}^+$, $\mathbf{e}^-$ respectively.

Finally, the observation likelihood is derived from the reconstruction error of $\mathbf{y}$ as

$$p(\mathbf{z}_t|\mathbf{x}_t) = \frac{1}{\Gamma} \exp\{-\alpha \| \mathbf{Ta} - \mathbf{y} \|^2\} , \quad (4)$$

where $\mathbf{a}$ is obtained by solving the ℓ_1 minimization (3), α is a constant controlling the shape of the Gaussian kernel, and Γ is a normalization factor.

B. Minimum Error Bound

In this section, we will show that the reconstruction error from the target templates with an ℓ_2 norm is bounded by a minimum error that can be calculated much faster than solving an ℓ_1 minimization.

[1]The frame at time t is treated as the observation $\mathbf{z}_t$.

1) Least squares error bound: The observation likelihood defined in (4) builds on the reconstruction error $\| \mathbf{Ta} - \mathbf{y} \|^2$ measured in the ℓ_2 norm. Since $\mathbf{a}$ is calculated by the ℓ_1 minimization (3), there is a natural lower bound for the reconstruction error

$$\| \mathbf{Ta} - \mathbf{y} \|^2 \geq \| \mathbf{T\hat{a}} - \mathbf{y} \|^2 \quad , \tag{5}$$

where

$$\hat{\mathbf{a}} = \arg\min_{\mathbf{b}} \| \mathbf{Tb} - \mathbf{y} \|^2 \tag{6}$$

is the linear least squares approximation of $\mathbf{y}$ in the subspace spanned by $\mathbf{T}$. One can also view $\hat{\mathbf{a}}$ as a degenerate case of $\mathbf{a}$ when $\lambda = 0$ in (3). Similarly, for the observation likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$, we derive its upper bound $q(\mathbf{z}_t|\mathbf{x}_t)$ using the least squares approximation error

$$q(\mathbf{z}_t|\mathbf{x}_t) = \frac{1}{\Gamma} \exp\{-\alpha \| \mathbf{T\hat{a}} - \mathbf{y} \|^2\} \ , \tag{7}$$

where α and Γ are the same as in (4). We immediately have

$$p(\mathbf{z}_t|\mathbf{x}_t) \leq q(\mathbf{z}_t|\mathbf{x}_t) \ . \tag{8}$$

2) Efficiency analysis: The linear system in (6) can be solved by Cholesky factorization or QR factorization. For dense matrices, the cost of the Cholesky factorization method is $dn^2 + (1/3)n^3$, while the cost of the QR factorization method is $2dn^2$, where d is the image dimension and n is the number of templates. The QR factorization method is slower by a factor of at most 2 if $d \gg n$, which is the case for our problem. For small and medium-size problems, the factor of two does not outweigh the difference in accuracy, and the QR factorization is the recommended method.

The original L1 tracker uses the preconditioned conjugate gradients (PCG) method [21] to solve the ℓ_1 minimization. The PCG algorithm computes 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 steps 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.

From the complexity analysis, we can see the solution to the least squares problem in (6) is two orders faster than the ℓ_1 solution. For example, if we are using template size of 15×12, then $d = 15 \times 12 = 180$. The number of templates is $n = 10$. The cost of Cholesky factorization method is $dn^2 + (1/3)n^3 = 180 \times 100 + (1/3) \times 1000 \approx 18000$. While the cost of a PCG step is $O(d^2 + dn) = O(32400)$, and there will be a few hundred such steps.

C. Bounded Particle Resampling

In the previous section, we showed that the computation is much more intensive to compute the observation probability $p(\mathbf{z}_t|\mathbf{x}_t)$ than to compute its upper bound $q(\mathbf{z}_t|\mathbf{x}_t)$. It is therefore attractive to use the upper bound for samples that are not promising enough and only conduct likelihood computations for the promising samples. Nevertheless, to search for the candidate with the maximum likelihood, we still need the observation probabilities for resampling. Fortunately, in many cases only a small portion of the samples will be preserved after resampling and an efficient algorithm can be designed to avoid computing all observation probabilities. We propose a two step testing to remove most insignificant samples from computing the ℓ_1 minimization. The first step is τ *testing*. The likelihood of the samples is compared with τ which is the summation of all previous calculated observation probabilities and the ones smaller than τ are removed without loss of precision during resampling. The second step is *max testing*. The tracking result for the current frame is defined as the sample that has the maximum observation probability p. The likelihood bound of the samples is compared with the current calculated maximum observation probability. A tracking result is found if the likelihood bound is smaller than the maximum observation probability. The samples below the maximum observation probability will not affect the current-frame tracking result, but only affect the resampling. Since the remaining samples do not affect the current frame, we approximate the remaining samples observation probability by interpolation. We divide the remaining samples into several groups. For each group, we calculate the observation probability for the first and last sample from the ℓ_1 minimization, and the probability of the middle samples is approximated by interpolation. The insignificant samples are further removed without altering the tracking result. Although sacrificing minimal precision during resampling, we achieve significant speed up from *max testing* in addition to the efficiency obtained from the thresholding in the first step.

1) τ Testing: We denote $\mathcal{X}_t = \{\mathbf{x}_t^1, \mathbf{x}_t^2, \cdots, \mathbf{x}_t^N\}$ as the sample set at time t, and denote $p_i = p(\mathbf{z}_t^i|\mathbf{x}_t^i)$ as the observation probability and its corresponding upper likelihood bound, $q_i = q(\mathbf{z}_t^i|\mathbf{x}_t^i)$, defined in previous subsection. At the end of each frame, the samples are resampled with respect to their observation probabilities. We have the following observation.

Observation 1. *If the sample appears at least once in the resampled set, its observation probability must satisfy the following condition*

$$p_i \geq \frac{1}{2N} \sum_{j=1}^{N} p_j \ . \tag{9}$$

The observation is straightforward because the number of samples remains unchanged before and after resampling. In order for the sample to appear in the resampled set at least once, the normalized probability $p_i / \sum_{j=1}^{N} p_j$ has to be greater than $\frac{1}{2N}$. The "2" in the denominator is due to rounding.

Motivated by the observation, we develop a two-stage bounded resampling algorithm to calculate the probability of the tracking candidates. The first stage is straightforward: we compute the probability bounds q_i for all samples and sort them in descending order. Without loss of generality, in the following we assume the samples are already sorted, i.e., $q_1 \geq q_2 \geq \cdots \geq q_N$.

In the τ testing, our task is to calculate the observation probability p_i for samples that will survive resampling. The observation probability can be done efficiently even for large number of samples by using a dynamically updated threshold τ

to exclude to-be-discarded samples. In particular, τ is defined according to the following theorem:

Theorem 1. *If the i^{th} sample $\boldsymbol{x}_i$ appears at least once after resampling, its likelihood bound q_i is no less than a threshold τ_i defined as*

$$\tau_i = \frac{1}{2N-1}\sum_{j=1}^{i-1} p_j \ . \tag{10}$$

Proof: From Observation 1, we have

$$2Np_i \geq \sum_{j=1}^{N} p_j \geq \sum_{j=1}^{i} p_j \ . \tag{11}$$

Subtract p_i on both sides and divide by $2N-1$, we have

$$p_i \geq \frac{1}{2N-1}\sum_{j=1}^{i-1} p_j = \tau_i \ . \tag{12}$$

Using the fact that q_i is an upper bound of p_i, we have $q_i \geq p_i \geq \tau_i$. ■

From the definition, we see that the thresholds are non-decreasing, i.e., $0 = \tau_1 \leq \tau_2 \leq \cdots \leq \tau_N$, and there is

$$\tau_{i+1} = \tau_i + \frac{p_i}{2N-1} \ , \tag{13}$$

which can be used for an efficient threshold update.

With the above thresholds, in the τ testing, we start with the first sample that has the largest likelihood bound q_1, and calculate the probability p_1 according to (4) and update the corresponding threshold τ_2 according to (13). Then we continue for samples 2, 3, ... For the i^{th} sample, if the likelihood bound $q_i \geq \tau_i$, we compute the observation probability p_i and update threshold τ_{i+1}. Otherwise if $q_i < \tau_i$, which according to Theorem 1, implies that $\mathbf{x}_i, \mathbf{x}_{i+1}, \cdots, \mathbf{x}_N$ will be discarded during resampling. Then, we directly set $p_i = p_{i+1} = \cdots = p_N = 0$. The probabilities $p_1, p_2, \cdots, p_N$ are then used for resampling set $\mathcal{X}$.

The above τ testing procedure does not sacrifice resampling precision, which is guaranteed by Theorem 1. τ testing avoids the expensive computation on samples with low likelihoods. The amount of time saved is mainly determined by the dissimilarity between the tracking foreground and its surrounding background. Intuitively, the larger the foreground/background difference, the more speedup from the BPR procedure. Furthermore, the τ testing framework encourages using more particles with larger sampling variance in comparison with the previously proposed L1 tracker [30]; this in turn helps improve tracking accuracy in addition to computational efficiency.

Empirical studies. Fig. 1 shows the curves of p, q, and τ when 600 particles are used. The values are calculated from one testing frame. After about 100 particle samples, when q becomes smaller than τ that is defined in Theorem 1 and fails the τ testing, the probability p of the rest samples are assigned to 0 without calculating the computational expensive ℓ_1 minimization. Hence $\log(p)$ drops suddenly since $\log(p)$ becomes negative infinity when p is 0. For this frame, only 20% of the samples need to solve the ℓ_1 minimization, and we achieve about 5 times speed up for this frame.

Fig. 2 shows the run time and the number of particle samples for which the ℓ_1 minimization is performed. We can see that the run time is proportional to the number of the samples and is dominated by the second stage probability calculation. For most of the frames in the sequence, only 20% of the particle samples are used in the ℓ_1 minimization update. Notice in Fig. 2, from frame 190 to 230, the man comes out of the shop and the woman is partially occluded. We clearly see that the number of particles calculating for the ℓ_1 minimization increases dramatically when the target is occluded. When the target is occluded, none of the samples can model the target appearance well enough so the probabilities are distributed between the particles. When the probabilities are evenly distributed, more particles are needed, which results in a longer run time for the frame. The more probabilities concentrate on the first few samples, the less particles are needed for the ℓ_1 minimization calculation.

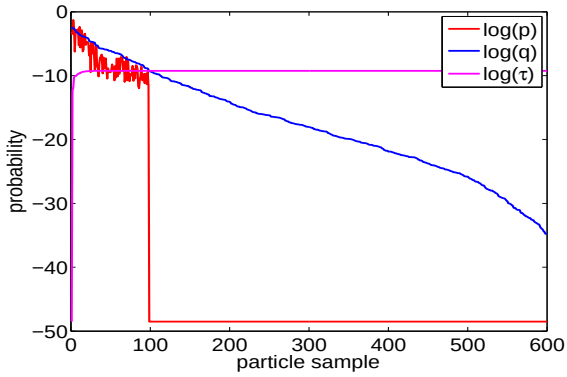

Fig. 1. The curves of p, q, and τ. The logarithm is applied to the data for display purpose.

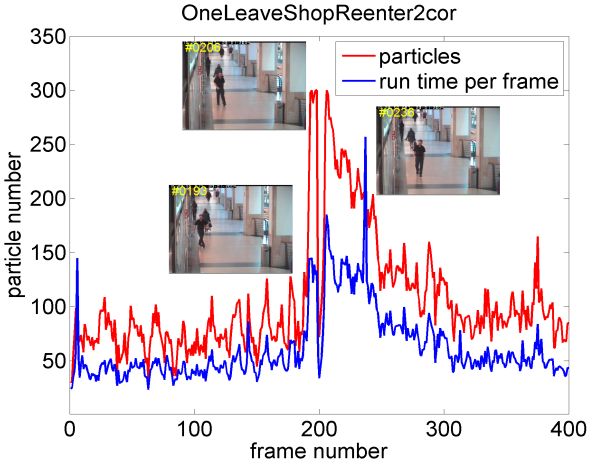

Fig. 2. The run time and number of particle samples for sequence *OneLeaveShopReenter2cor* using τ testing.

2) Max Testing: If at certain point, the likelihood bound q_i is smaller than the maximum of the observation probability p_{max}, where $p_{max} = \max_{j=1,\ldots,i-1}\{p_j\}$, then the tracking result is found. The observation probability p_j, where $j = i, i+1, \cdots, n$, only affects the resampling and does not impact the current-frame tracking result. Inspired by this, we propose a *max testing* operation to further reduce the insignificant samples without altering the tracking result. This *max testing* has the flexibility to be combined with the τ testing to further

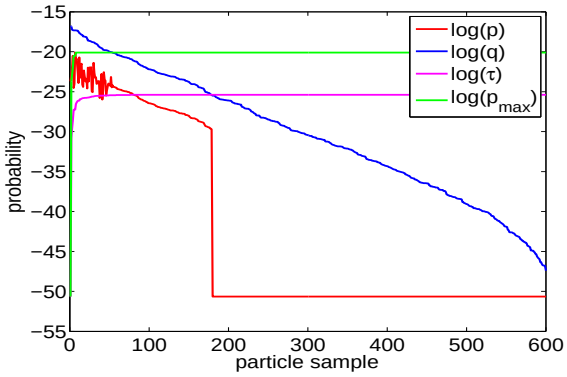

Fig. 3. The curves of p, q, τ, and p_{max}. The logarithm is applied to the data for display purpose.

speed up the L1 tracker. The number of ℓ_1 minimization calculations is reduced to a fixed number depending on the number of groups that the particles are divided into. First, the rest particles from i to N are divided into k groups. Second, for each group, we calculate the observation probability of the samples whose likelihood bounds are the maximum and minimum in the group and greater than τ. For the samples whose likelihood lies in between, the observation probability is the linear interpolation of the two. The maximum number of ℓ_1 minimization is a fixed number of $2k$. The proposed two stage bounded resampling is summarized in Algorithm 2.

Empirical studies. Fig. 3 shows the curves of p, q, τ, and p_{max} calculated from one frame when 600 particles are used. After 55 particle samples, q is becoming smaller than p_{max}, and the probability of the remaining samples is the linear interpolation between the first and last observation probability of each group. After 179 particle samples, q is becoming smaller than τ, the probability p of the rest samples are assigned to 0 without calculating the computational expensive ℓ_1 minimization. For this frame, only about 10% of the samples need to solve the ℓ_1 minimization, and we achieve about 10 times speed up for this frame.

Fig. 4 shows the run time and the number of particle samples for which the ℓ_1 minimization is performed using both τ and max testing. We can see that the run time is proportional to the number of the samples and is dominated by the second stage probability calculation as in Fig. 2. It has similar curve as in Fig. 2 as well. For most of the frames in the sequence, only 7% of the particle samples calculate the ℓ_1 minimization. The average particle samples per frame for this sequence is about 30, and 300 particles are used for the experiment. We achieve about 10 times speed up for this sequence.

V. Occlusion Detection

The template set needs to be updated to capture the appearance variations of the target during tracking. In [30], the tracking result is added to the template set if none of the template is similar to the tracking result. Therefore, the tracker is vulnerable to failures when the tracking result with a large occlusion is added to the template set. To prevent an improper addition to the template set, we propose a method to detect the large occlusion in the tracking result before it is added to the template set. Fig. 5 illustrates the occlusion detection algorithm flow.

For occlusion detection, we investigate the responses in the trivial templates when solving the ℓ_1 minimization (3). The trivial templates are activated when the pixel intensity can not

Algorithm 2 Two-stage Bounded Resampling

Input: sample set $\mathcal{X}_{t-1} = \{\mathbf{x}_{t-1}^i\}_{i=1}^N$
Output: sample set $\mathcal{X}_t = \{\mathbf{x}_t^i\}_{i=1}^N$

/*Stage 1*/
for $i = 1$ to N **do**
  Draw sample $\mathbf{x}_t^i$ from $\mathbf{x}_{t-1}^i$
  Prepare the sample appearance $\mathbf{y}_t^i$ from $\mathbf{x}_t^i$
  Solve the linear least squares problem (6) for $\mathbf{y}_t^i$
  Compute q_i according to (7)
end for
Sort samples in descending order according to q_i

/*Stage 2*/
$i \leftarrow 1$, $\tau_1 \leftarrow 0$
while $i \leq N$ **do**
  /*τ Testing*/
  if $q_i < \tau_i$ **then**
    break
  end if
  /*max Testing*/
  if $q_i < \max\{p_j\}_{j=1}^{i-1}$ **then**
    $r \leftarrow (N - i + 1)/k$
    for $l = 1$ to k **do**
      $i_m = \text{argmin}_j\{q_j \geq \tau_i\}_{j=i}^{i+r-1}$
      Solve the ℓ_1 minimization (3) for $\mathbf{y}_t^i$ and $\mathbf{y}_t^{i_m}$
      Compute p_i and p_{i_m} according to (4)
      if q_i == q_{i_m} **then**
        $p_j \leftarrow (p_i + p_{i_m})/2,\ j = i, \cdots, i_m$
      else
        $p_j \leftarrow p_i + \frac{(p_i - p_{i_m})(q_j - q_i)}{(q_i - q_{i_m})},\ j = i, \cdots, i_m$
      end if
      if $i_m < i + r - 1$ **then**
        $i \leftarrow i_m$
        $\tau_i \leftarrow \frac{1}{2N-1}\sum_{j=1}^{i-1} p_j$
        break
      else
        $i \leftarrow i + r$
        $\tau_i \leftarrow \frac{1}{2N-1}\sum_{j=1}^{i-1} p_j$
      end if
    end for
  else
    Solve the ℓ_1 minimization (3) for $\mathbf{y}_t^i$
    Compute p_i according to (4)
    $\tau_{i+1} \leftarrow \tau_i + p_i/(2N - 1)$
    $i \leftarrow i + 1$
  end if
end while
$p_j \leftarrow 0,\ \forall j \geq i$
/*Resampling*/
$\mathcal{X}_t \leftarrow$ Resample $\{\mathbf{x}_t^i\}_{i=1}^N$ with respect to $\{p_i\}_{i=1}^N$

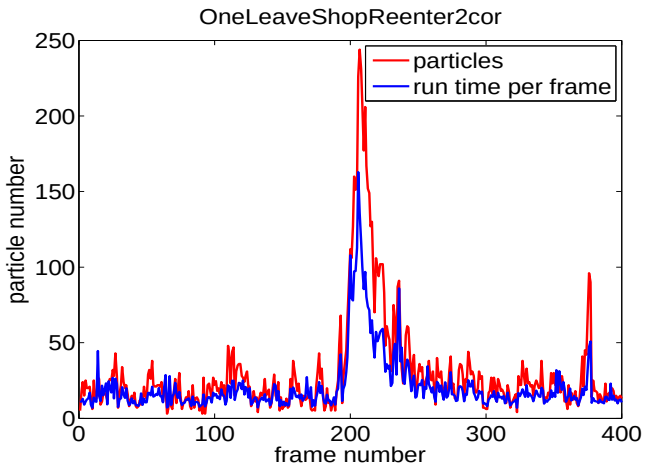

Fig. 4. The run time and number of particle samples for sequence OneLeaveShopReenter2cor using both τ and max testing.

Fig. 5. The flow chart of the occlusion detection algorithm.

be well approximated using the target templates. Therefore, we explore the trivial template coefficients for the occlusion detection. We convert the 1D trivial coefficient vector to a 2D trivial image by reversing the way that the target template is vectorized. Each pixel in trivial image is mapped to the pixel in the same location in the template image. We threshold the trivial image and obtain another 2D binary image that we call an *occlusion map*. The white pixel in the occlusion map indicates that the pixel is occluded and the black pixel indicates no occlusion. We assume that an occluder is large in size and its intensity is different enough to be separated from small random noise. Therefore, the occlusion is a large connected region in the occlusion map. The occlusion detection is then reduced to finding a white area that is large enough to be classified as an occlusion. After applying morphological operations to the occlusion map to remove the small areas and fill the small holes between regions, we count the number of pixels in the largest region. If the area is larger than a predefined threshold, say 30% of the area in the occlusion map, we conclude that there is an occlusion in the tracking result, and the template set should not be updated.

Normally when an occlusion is detected, it will persist for a certain period of time. For example, when the target is occluded by an object, and the object is moving away from the target, the occlusion becomes smaller before it disappears. In our occlusion detection method, we avoid updating the template set for the next five frames after an occlusion is detected. Future work would optimize the frame selection based on the scenario.

For illustration, Fig. 6 shows tracking results of the *faceocc2* sequence with frame 710 (left column) and 742 (right column). The top row shows the tracking result without occlusion handling, while the bottom row with occlusion handling. In frame 710, the target, face, undergoes heavy occlusion by a book. Without occlusion detection, the templates are contaminated and the tracker drifts apart in frame 742. The proposed tracker, by contrast, is able to follow the target after the book is moved away from the face.

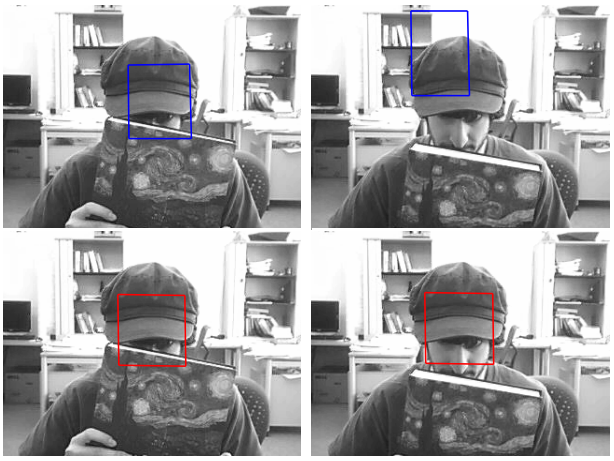

Fig. 6. Example tracking results of the *faceocc2* sequence without occlusion detection (top) and with occlusion detection (bottom).

VI. Experiments

We implemented the proposed approach in MATLAB with the SPArse Modeling (SPAM) Software package[2] [29] and evaluated the performance on eleven publicly available video sequences.[3] We use these sequences to test various challenges including background changes (illumination), target changes (pose and scale), occlusion variations (none, partial, and significant), and cluttered environmental conditions (bleaching, multiple targets, and bright spots/rays).

The α (Eq. 4) controlling the shape of the Gaussian kernel and the regularization parameter λ (Eq. 3) are initialized to 40 and 0.01 for all the experiments. The number of groups k in *max testing* is set to 3. Our proposed tracker is compared with nine state-of-the-art trackers include Incremental Visual Tracking (IVT) [33], Multiple Instance Learning (MIL) [3], Visual Tracking Decomposition (VTD) [23], Generalized Kernel Tracking (GKT) [34], L1 tracker (L1) [30], Covariance Tensor Learning (CTL) [37], Online AdaBoost (OAB) [12], Tracking-Learning-Detection (TLD) [19], and local sparse appearance model (SPT) [27]. The comparative tracker results were obtained by running the source code or binaries provided by their authors using the same initial positions in the first frame.

Our method is implemented in MATLAB and can process 5 frames per second on an Intel Core2Duo 2.66GHz standard PC with 8 GB memory. We expect our method to process 10 to 15 frames per second with optimized implementation. We use the same number of particles for trackers involving particle filtering for fair comparison.

[2]http://www.di.ens.fr/willow/SPAMS/downloads.html

[3]Sequences 1–3 were from http://www.cs.toronto.edu/~dross/ivt/. Sequences 4–7 were from the PETS 2001 dataset http://www.cvg.cs.rdg.ac.uk/ PETS2001/, http://groups.inf.ed.ac.uk/vision /CAVIAR/CAVIARDATA1/, http://vision.stanford.edu/~birch/headtracker /seq/, http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml, and http://www.dabi.temple.edu/~hbling/code_data.htm

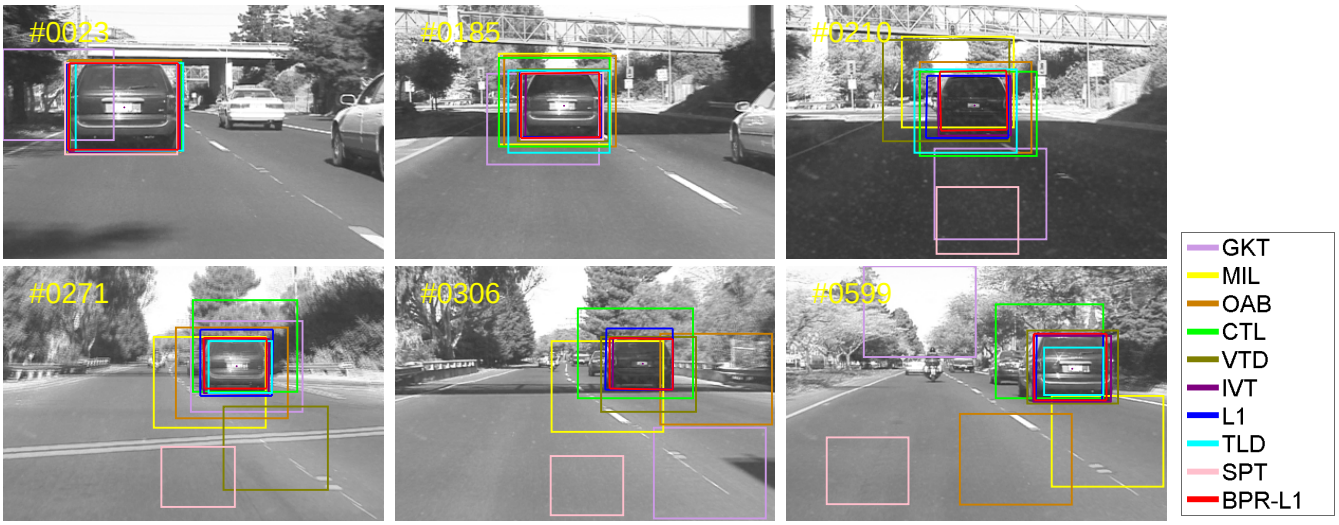

Fig. 7. Tracking results of the *car4* sequence.

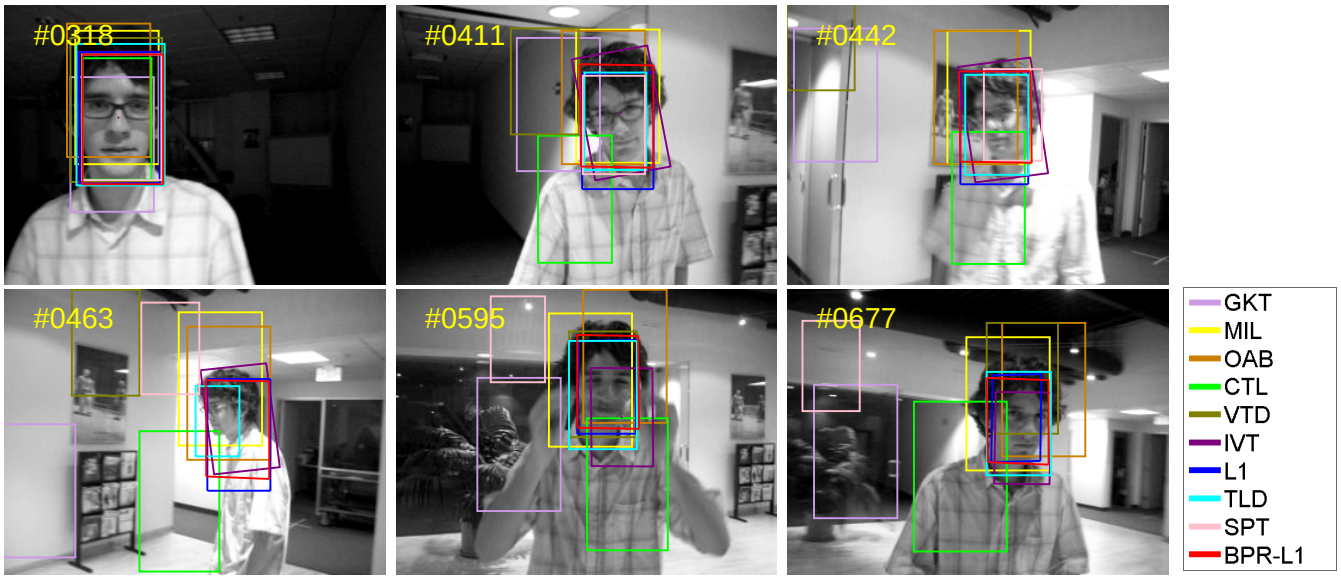

Fig. 8. Tracking results of the *David Indoor* sequence.

A. *Qualitative Comparison*

The first sequence, *car4*, shows a vehicle undergoes drastic illumination change as it passes beneath a bridge and under trees. Tracking results on several frames are presented in Fig. 7. The BPR-L1 tracker, L1 tracker, IVT, TLD and CTL are able to track the target well even through the drastic illumination changes, while the other trackers lose the target due to the illumination change after it goes under the bridge.

The second sequence, *David Indoor*, was captured in an indoor environment. Results on several frames are presented in Fig. 8. The BPR-L1 tracker, L1 tracker, IVT, TLD, OAB and MIL track the target faithfully throughout the sequences. The other trackers fail to track the target when there are both pose and illumination changes.

Results on the third sequence, *sylvester*, are shown in Fig. 9. It shows a moving animal doll and presents challenging lighting, pose, and scale changes. The L1 tracker, IVT, and VTD eventually fail in frame 736 as a result of a combination of drastic pose and illumination change. The BPR-L1, GKT, MIL, OAB, TLD and CTL trackers are able to track the target for this long sequence, though GKT and SPT are a little off the target in frames 521, and 546.

In the fourth sequence, *PETS01D1Human1*, a person is walking from right bottom corner to the left of the image (Fig. 10). The target is very small compared to the image and therefore does not have much discrimination against the background. Most trackers, including BPR-L1, track the target successfully. IVT and SPT lock to the background and are not able to recover from early failures.

The fifth sequence, *OneLeaveShopReenter2cor*, includes people walking in and out of a crowded hallway. 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

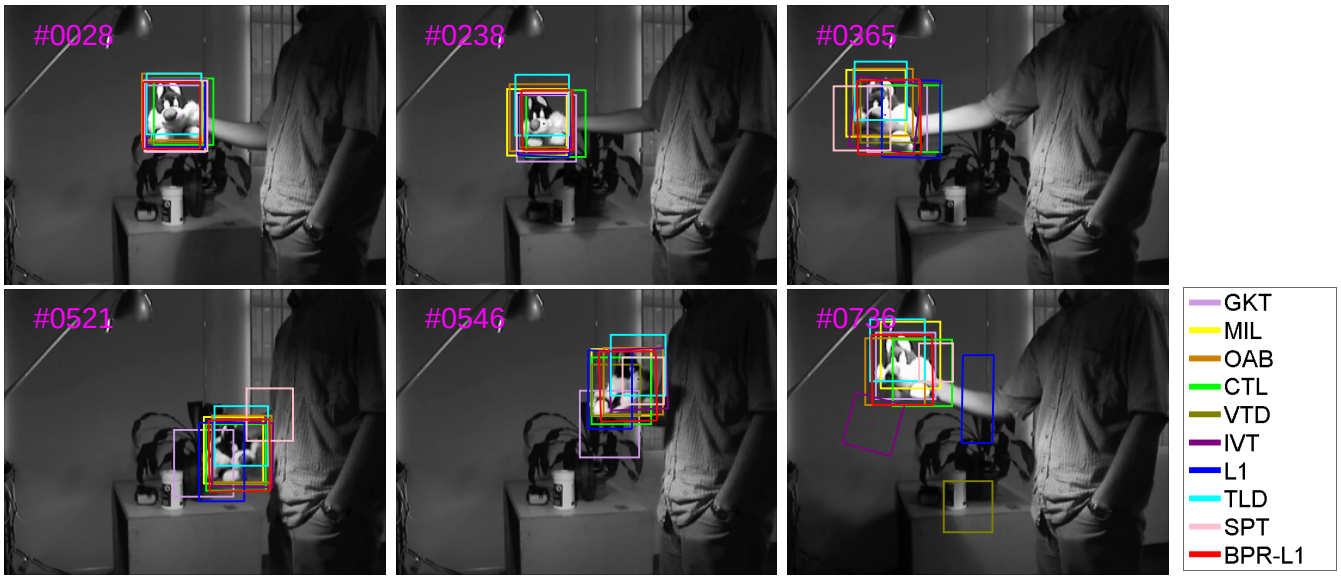

Fig. 9. Tracking results of the *sylvester* sequence.

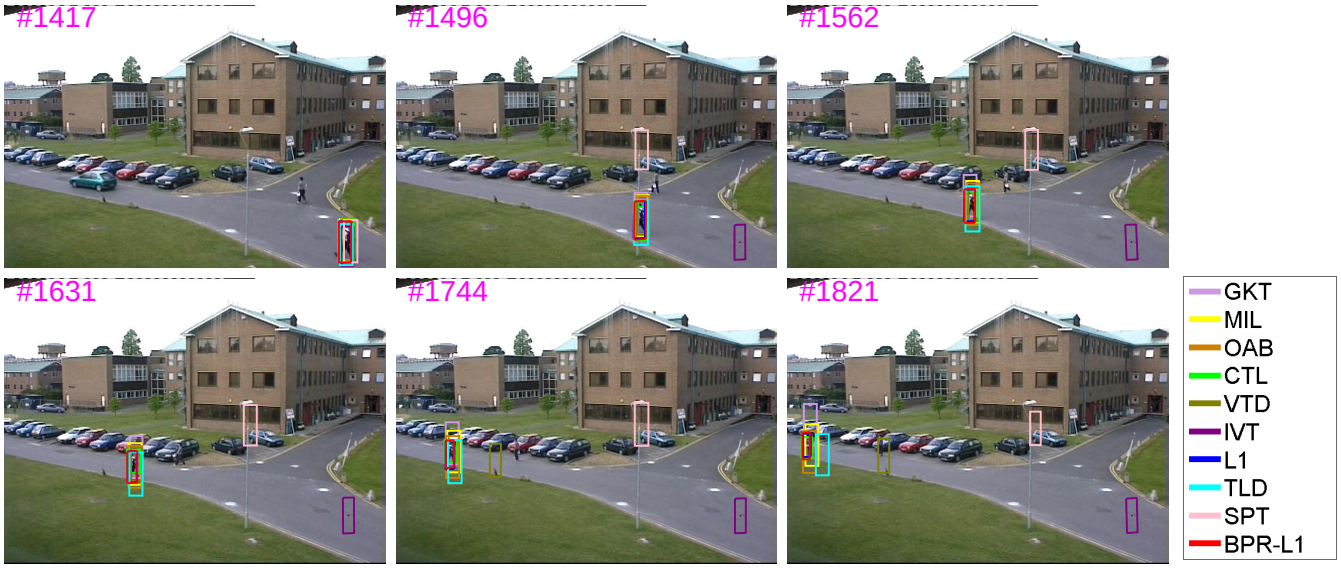

Fig. 10. Tracking results of the *PETS01D1Human1* sequence.

occlusion. Some result frames are given in Fig. 11. The BPR-L1, L1 and TLD trackers are able to track the target during the entire sequence. Many other trackers lock on the man when he occludes the woman after he comes out of the shop.

Results of the sixth sequence, *girl*, are shown in Fig. 12. In this sequence, we show the robustness of our algorithm in handling occlusion and large pose changes. All the trackers track the target in the entire sequence except for the MIL, which loses the target in the frame 466.

Results on the seventh sequence, *Occluded Face*, are shown in Fig. 13. Many trackers start drifting from the target when the man's face is almost fully occluded by the book. The BPR-L1 tracker explicitly handles occlusions, and update target templates accordingly. Therefore, it handles well appearance change in this sequence and continues tracking the target when the occlusion disappears.

Fig. 14 presents the most challenging scenario in that the sequence, *shaking*, has moving targets, variations in illumination with bright spots and rays, as well as occlusions; however retains similarity to the face tracking scenarios as above. Consistent track maintenance is achieved by the MIL and BPR-L1 trackers only.

In the ninth sequence, *singer* shown in Fig. 15, the singer wears white clothes which are similar to the background and matches the clutter from the light. CTL loses track because it focuses on the color discrimination that is not evident in the image. The L1 tracker loses the singer target due to the frequent template update. The GKT, MIL, OAB, VTD, IVT, SPT and BPR-L1 all track the target; however, the BPR-L1 captures the target size more accurately.

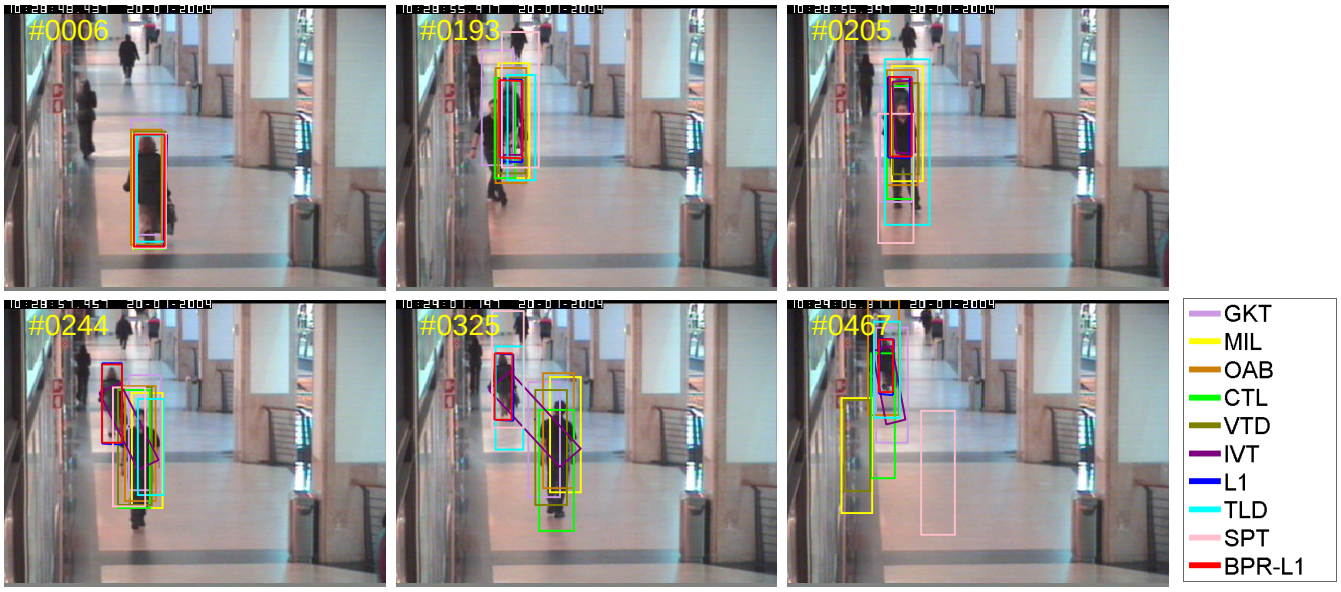

Fig. 11. Tracking results of the *OneLeaveShopReenter2cor* sequence.

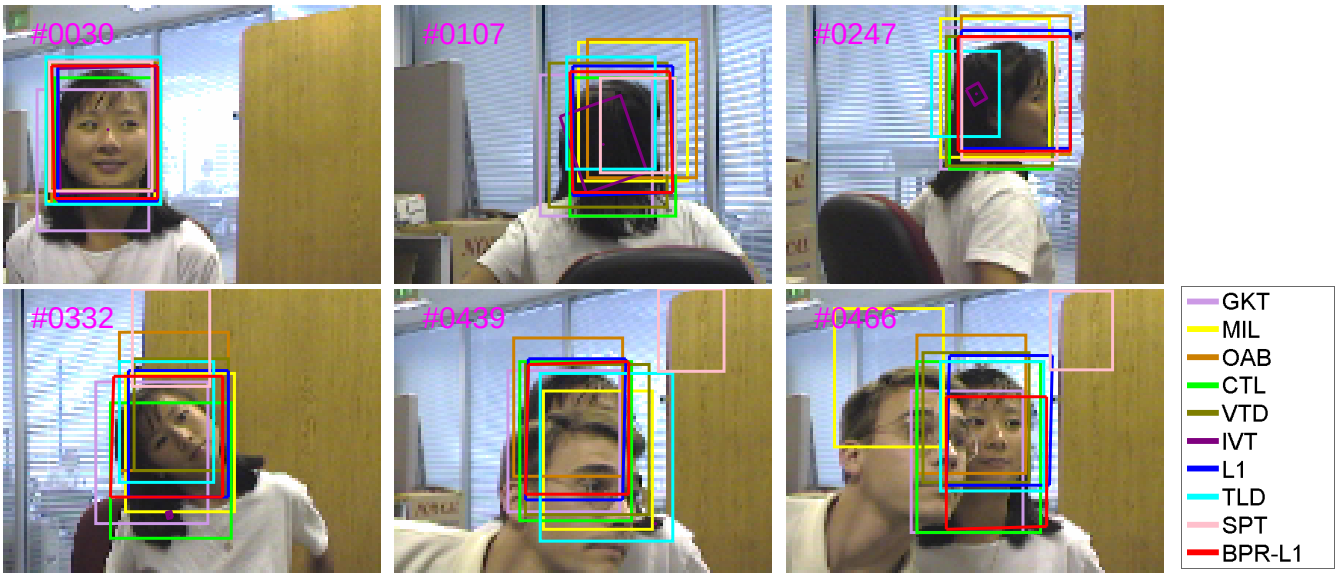

Fig. 12. Tracking results of the *girl* sequence.

Fig. 16 presents the tenth sequence, *pktest02*. The multi-car traffic scene has many objects in the image that are similar to the tracking target, thus present numerous opportunities for target switching. It confuses GKT and OAB confused throughout the sequence and does the same to MIL, TLD and IVT after a while. Another reason for failure may be attributed to the corruption of the target appearance from the occlusion and shadow of trees. The CTL, SPT, L1 and BPR-L1 successfully tracks the target through the sequence. The last experiment is on a long sequence (total 2600 frames), *doll*, taken by a hand-held camcorder [30] and shown in the Fig. 17. The doll undergoes scale, out-of-plane rotation, and occlusion. The VTD, GKT, and MIL lose the target after around 1000, 1400, and 1600 frames. The SPT does not have a consistent tracking results. The OAB gradually drift away from the target and lost the target at the end of the sequence. The IVT lose the target from frame 2400. Only CTL, L1, TLD, and BPR-L1 can track the target throughout the whole sequence.

B. Quantitative Comparison and Discussion

To quantitatively compare robustness under challenging conditions, we manually labeled the ground truth of the eleven sequences. The tracking accuracy is measured by the relative position errors (in pixels) between the center of the tracking result and that of the ground truth. Ideally, the position differences should be around zero. The results are summarized in Fig. 18.

Two issues are important in the quantitative results. First, when a tracker loses the target, it is difficult to reacquire the target and the error grows rapidly. Second, the proposed

Fig. 13. Tracking results of the *Occluded Face* sequence.

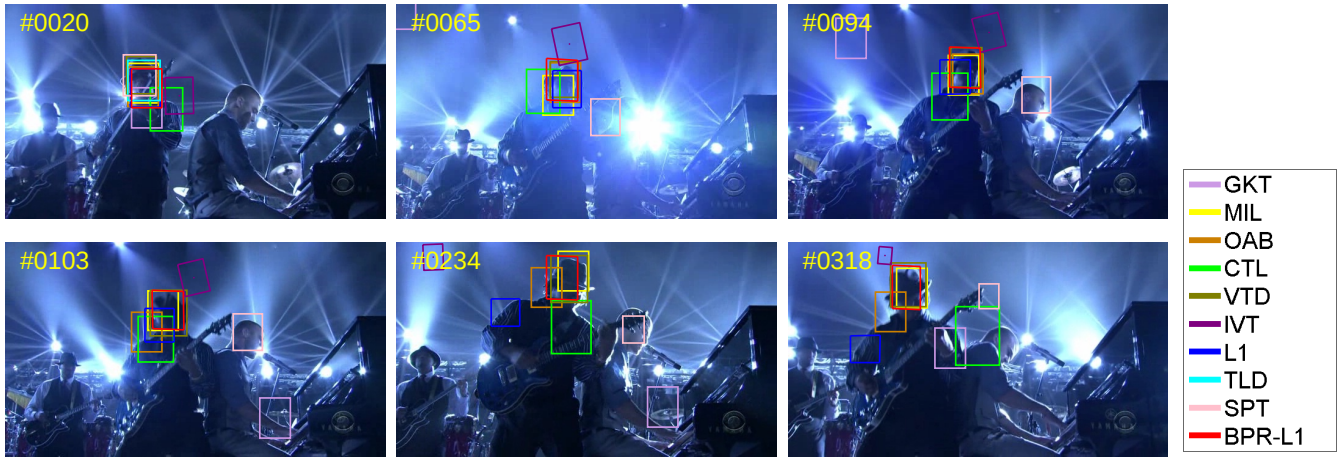

Fig. 14. Tracking results of the *shaking* sequence.

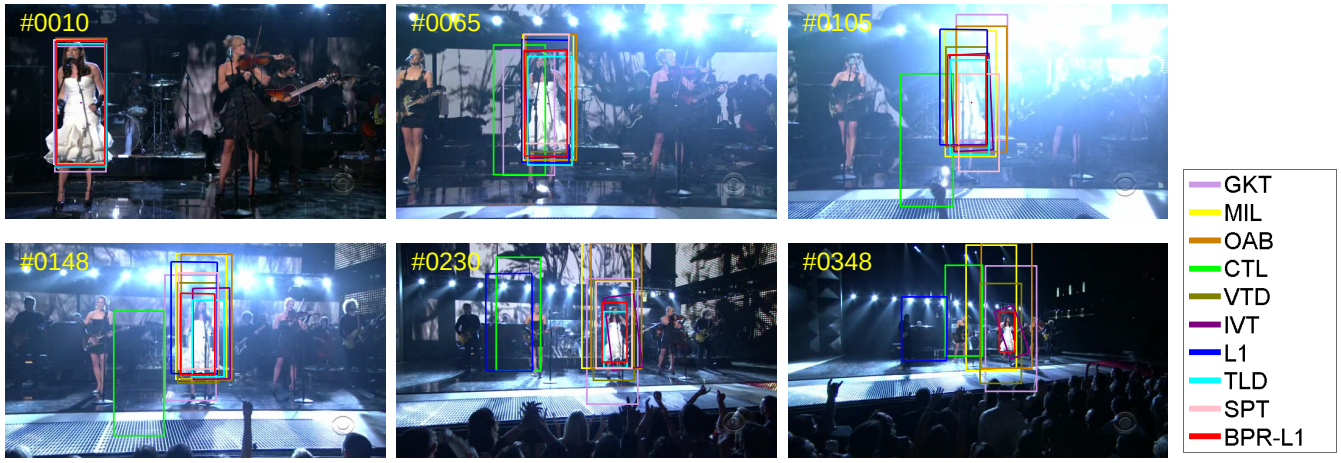

Fig. 15. Tracking results of the *singer1* sequence.

BPR-L1 tracker in general produces low tracking error, though there are cases where BPR-L1 does not perform the best (e.g. occluded face, frames 620-750).

In general, we attributes the excellent performance of BPR-

Fig. 16. Tracking results of the *pktest02* sequence.

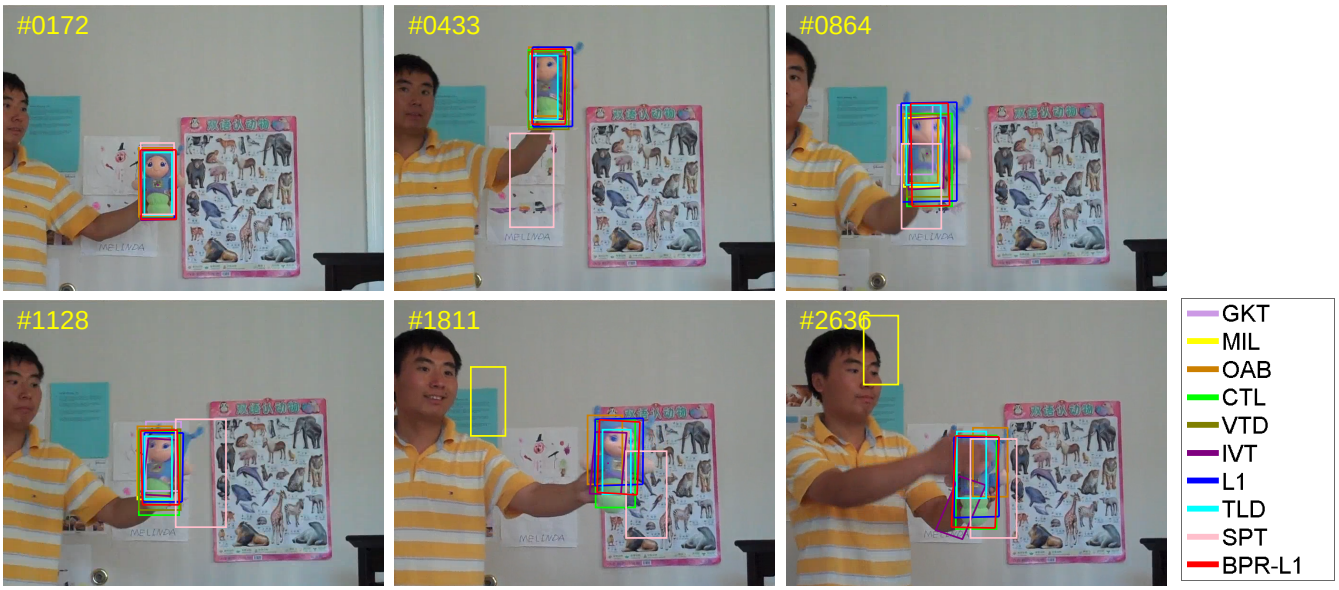

Fig. 17. Tracking results of the *doll* sequence.

L1 to: (1) the speed up allows more particles to be used so as to better approximate the posterior probability in the sequential Bayesian inference, and (2) the occlusion detection and consequent handling reduces the chances of drifting.

One limitation of our method lies in the holistic appearance model for tracking targets. The model takes pixel position into account and may have problems when dealing with non-rigid objects in which relative pixel positions have changed. In addition, in some cases the model may suffer from misalignment, especially when images/templates of large sizes are used.

VII. Conclusions

We demonstrated an efficient Bounded-Particle Re-sampling L1 minimization (BPR-L1) tracker with minimum error bound and occlusion detection. The BPR-L1 tracker employs a two-stage sample probability scheme, where most samples with small probabilities from first stage are filtered out without solving the computationally expensive ℓ_1 minimization. The occlusion detection coupled with a template update scheme effectively prevents the occluded target from contaminating the target template set. Preventing an incorrect update to the target template set reduces tracking failures. Our proposed BPR-L1 method is computationally more efficient than the previous L1 tracker, and demonstrates the effectiveness in handling a number of challenging sequences such as variations in illumination, pose, occlusions, and dense targets. We compared the BPR-L1 tracker with nine other state-of-the-art trackers including our original L1 tracker on eleven sequences

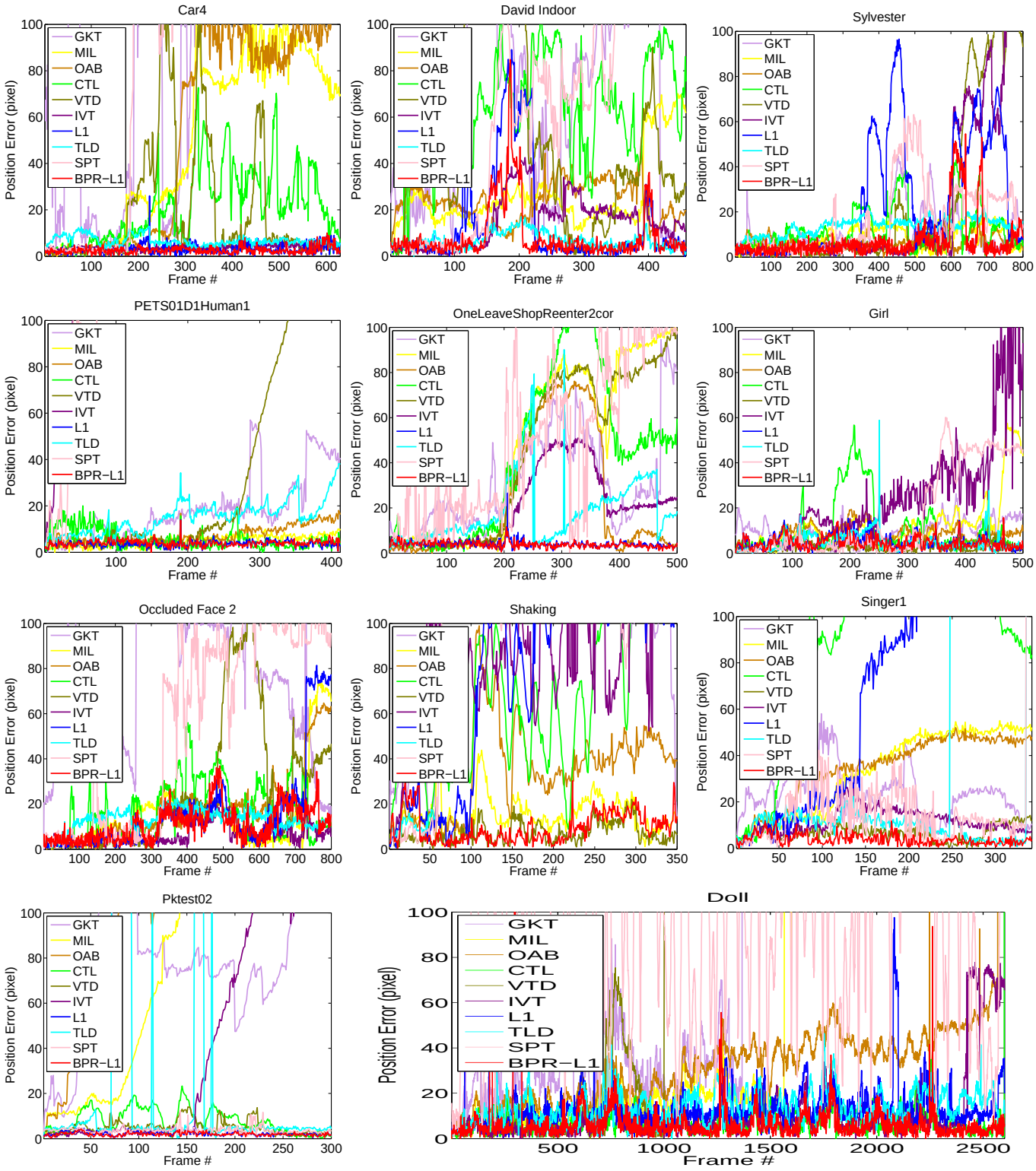

Fig. 18. Quantitative comparison of the trackers in terms of position errors (in pixel).

to validate robustness. Overall, BPR-L1 demonstrated very promising tracking accuracy and ran significantly faster than the original L1 tracker.

Acknowledgment We thank the editor and anonymous reviewers for valuable suggestions, which help significantly improve the manuscript. This work was supported in part by NSF Grants IIS-0916624 and IIS-1218156.

REFERENCES

[1] A. Adam, E. Rivlin, and I. Shimshoni. "Robust fragmentsbased tracking using the integral histogram", *Proc. IEEE Conf. on Computer Vision and*

Pattern Recognition, 798-805, 2006.
[2] S. Avidan. "Ensemble Tracking", *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", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 983-990, 2009.
[4] C. Bao, Y. Wu, H. Ling, and H. Ji. "Real Time Robust L1 Tracker Using Accelerated Proximal Gradient Approach", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1830-1837, 2012.
[5] M. J. Black and A. D. Jepson. "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation", *Int. J. Computer Vision*, 26:63-84, 1998.
[6] E. Blasch and B. Kahler. "Multiresolution EO/IR Target Tracking and Identification", *Int. Conf. on Info Fusion*, 275-282, 2005.
[7] 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.
[8] D. Chen and J. Yang. "Robust Object Tracking Via Online Dynamic Spatial Bias Appearance Models." *IEEE Trans. Pattern Anal. Mach. Intell.* 29(12): 2157-2169, 2007.
[9] R. T. Collins, and Y. Liu. "On-Line Selection of Discriminative Tracking Features", *Proc. of Int. Conf. on Computer Vision*, 346-352, 2003.
[10] D. Comaniciu, and V. Ramesh, and P. Meer. "Kernel-based object tracking", *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):564-577, 2003.
[11] D. Donoho. "Compressed Sensing", *IEEE Trans. Inf. Theory*, 52(4):1289-1306, 2006.
[12] H. Grabner, M. Grabner, and H. Bischof. "Real-time tracking via online boosting". *Proc. of British Machine Vision Conf.*, 47-56, 2006.
[13] J. Ho, K.-C. Lee, M.-H. Yang, and D. Kriegman. "Visual tracking using learned subspaces", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 782-789, 2004.
[14] Z. Hong, X. Mei, and D. Tao. "Dual-Force Metric Learning for Robust Distracter Resistant Tracker", *Proc. of European Conf. on Computer Vision*, 513-527, 2012.
[15] W. Hu, X. Li, X. Zhang, X. Shi, S.J. Maybank, and Z. Zhang. "Incremental Tensor Subspace Learning and Its Applications to Foreground Segmentation and Tracking." *Int. J. Computer Vision*, 91(3):303-327, 2011.
[16] M. Isard and A. Blake. "Condensation - conditional density propagation for visual tracking", *Int. J. Computer Vision*, 28(1):5-28, 1998.
[17] X. Jia, H. Lu, and M. Yang. "Visual Tracking via Adaptive Structural Local Sparse Appearance Model", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1822-1829, 2012.
[18] Z. Kalal, J. Matas, and K. Mikolajczyk. "P-N Learning: Bootstrapping Binary Classifiers by Structural Constraints", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 49-56, 2010.
[19] Z. Kalal, K. Mikolajczyk, and J. Matas. "Tracking-Learning-Detection", *IEEE Trans. Pattern Anal. Mach. Intell.* 34(7): 1409-1422, 2012.
[20] Z. Khan, T. Balch, and F. Dellaert. "A Rao-Blackwellized particle filter for EigenTracking", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 980-986, 2004.
[21] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. "A method for large-scale ℓ_1-regularized least squares", *IEEE J. on Sel. Topics in Signal Processing*, 1(4):606-617, 2007.
[22] S. Kwak, W. Nam, B. Han, and J. H. Han. "Learning Occlusion with Likelihoods for Visual Tracking", *Proc. of Int. Conf. on Computer Vision*, 1551-1558, 2011.
[23] J. Kwon and K. M. Lee. "Visual Tracking Decomposition", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1269-1276, 2010.
[24] Y. Lei, X. Ding and S. Wang. "Visual Tracker Using Sequential Bayesian Learning: Discriminative, Generative, and Hybrid", *IEEE Trans. Systems, Man, and Cybernetics, Part B*, 38(6), 1578-1591, 2008.
[25] H. Li, C. Shen, and Q. Shi. "Real-time visual tracking using compressive sensing", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1305-1312, 2011.
[26] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski. "Robust and Fast Collaborative Tracking with Two Stage Sparse Optimization", *Proc. of European Conf. on Computer Vision*, 624-637, 2010.
[27] B. Liu, J. Huang, L. Yang, and C. Kulikowsk. "Robust Tracking Using Local Sparse Appearance Model and K-Selection", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1313-1320, 2011.
[28] R. Liu, and J. Cheng, and H. Lu, "A robust boosting tracker with minimum error bound in a co-training framework", *Proc. of Int. Conf. on Computer Vision*, 1459-1466, 2009.
[29] J. Mairal, F. Bach, J. Ponce and G. Sapiro. "Online Learning for Matrix Factorization and Sparse Coding." *J. of Machine Learning Research*, 11:19-60, 2010.
[30] X. Mei and H. Ling. "Robust Visual Tracking and Vehicle Classification via Sparse Representation", *IEEE Trans. Pattern Anal. Mach. Intell.* 33(11): 2259-2272, 2011.
[31] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai. "Minimum Error Bounded Efficient ℓ_1 Tracker with Occlusion Detection", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1257-1264, 2011.
[32] F. Porikli, O. Tuzel and P. Meer. "Covariance tracking using model update based on lie algebra", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 728-735, 2006.
[33] D. Ross, J. Lim, R. Lin and M. Yang. "Incremental learning for robust visual tracking", *Int. J. Computer Vision*, 77:125-141,2008.
[34] C. Shen, J. Kim and H. Wang. "Generalized Kernel-based Visual Tracking", *IEEE Trans. on Circuits and Systems for Video Technology*, 20(1), 119-130, 2010.
[35] D. Wang, H. Lu and M. Yang. "Online Object Tracking With Sparse Prototypes", IEEE Trans. on Image Processing, 22(1), 314-325, 2013.
[36] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. "Robust Face Recognition via Sparse Representation", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 31(1):210-227, 2009.
[37] Y. Wu, J. Cheng, J. Wang, H. Lu, J. Wang, H. Ling, E. Blasch, and L. Bai, "Real-time Probabilistic Covariance Tracking with Efficient Model Update", *IEEE Trans. on Image Processing*, 21(5): 2824-2837, 2012.
[38] Y. Wu, H. Ling, J. Yu, F. Li, X. Mei, and E. Cheng. "Blurred Target Tracking by Blur-driven Tracker", *Proc. of Int. Conf. on Computer Vision*, 1100-1107, 2011.
[39] A. Yang, Z. Zhou, A. Ganesh, S. Sastry, and Y. Ma. "A Review of Fast l1-Minimization Algorithms for Robust Face Recognition", arxiv.org, 2012.
[40] C. Yang, R. Duraiswami, and L. Davis. "Fast multiple object tracking via a hierarchical particle filter", *Proc. of Int. Conf. on Computer Vision*, 212-219, 2005.
[41] A. Yilmaz, O. Javed, and M. Shah. 'Object tracking: A survey". *ACM Comput. Survey*, 38(4), 2006.
[42] Z. Yin and R. Collins. "Object tracking and detection after occlusion via numerical hybrid local and global mode-seeking", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 1-8, 2008.
[43] Q. Yu, T. B. Dinh, and G. Medioni. "Online tracking and reacquistion using co-trained generative and discriminative trackers", *European Conf. on Computer Vision*, 678-691, 2008.
[44] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. "Robust Visual Tracking via Multi-Task Sparse Learning", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2042-2049, 2012.
[45] K. Zhang, L. Zhang, and M. Yang. "Real-time Compressive Tracking", *European Conf. on Computer Vision*, 864-877, 2012.
[46] X. Zhang, W. Li, W. Hu, H. Ling, and S. Maybank. "Block Covariance Based ℓ_1 Tracker with a Subtle Template Dictionary", *Pattern Recognition*, 46(7):1750-1761, 2013.
[47] S. K. Zhou, R. Chellappa, and B. Moghaddam. "Visual tracking and recognition using appearance-adaptive models in particle filters", *IEEE Trans. on Image Processing*, 13(11):1491-1506, 2004.