Robust Discriminative Tracking via Landmark-Based Label Propagation

Yuwei Wu, Mingtao Pei, Min Yang, Junsong Yuan, *Member, IEEE*, and Yunde Jia, *Member, IEEE*

***Abstract*— The appearance of an object could be continuously changing during tracking, thereby being not independent identically distributed. A good discriminative tracker often needs a large number of training samples to fit the underlying data distribution, which is impractical for visual tracking. In this paper, we present a new discriminative tracker via landmark-based label propagation (LLP) that is nonparametric and makes no specific assumption about the sample distribution. With an undirected graph representation of samples, the LLP locally approximates the soft label of each sample by a linear combination of labels on its nearby landmarks. It is able to effectively propagate a limited amount of initial labels to a large amount of unlabeled samples. To this end, we introduce a local landmarks approximation method to compute the cross-similarity matrix between the whole data and landmarks. Moreover, a soft label prediction function incorporating the graph Laplacian regularizer is used to diffuse the known labels to all the unlabeled vertices in the graph, which explicitly considers the local geometrical structure of all samples. Tracking is then carried out within a Bayesian inference framework, where the soft label prediction value is used to construct the observation model. Both qualitative and quantitative evaluations on the benchmark data set containing 51 challenging image sequences demonstrate that the proposed algorithm outperforms the state-of-the-art methods.**

***Index Terms*— Visual tracking, label propagation, appearance changes, Laplacian regularizer.**

I. Introduction

A GOOD appearance model is one of the most critical prerequisites for successful visual tracking. Designing an effective appearance model is still a challenging task due to appearance variations caused by background clutter, object deformation, partial occlusions, and illumination changes, *etc.* Numerous tracking algorithms have been proposed to address this issue [1], [2], and existing tracking algorithms can be roughly categorized as either generative [3]–[7] or discriminative [8]–[14] approaches. Generative methods build an object representation, and then search for the region most similar to the object. However, generative models do not take into account background information. Discriminative methods train an online binary classifier to adaptively separate the object from the background, which are more robust against appearance variations of an object. In this paper, we focus on the discriminative tracking method.

In visual tracking scenarios, samples obtained by the tracker are drawn from an unknown underlying data distribution. The appearance of an object could be continuously changing and thus it is impossible to be independent and identically distributed (*i.i.d*). A good discriminative tracker often needs a large number of labeled samples to adequately fit the real data distribution [15]. This is because if the dimensionality of the data is large compared to the number of samples, then many statistical learning methods will be overfitting due to the "curse of dimensionality". However, precisely labeled samples only come from the first frame during tracking, *i.e.*, the number of labeled samples is very small. To acquire more labeled samples, in most existing discriminative tracking approaches, the current tracking result is used to extract positive samples and the surrounding regions are used to extract negative samples. Once the tracker location is not precise, the assigned labels may be noisy. Over time, the accumulation of errors can degrade the classifier and cause drift. This situation makes us wonder: *with a very small number of labeled samples, whether we can design a new discriminative tracker which makes no specific assumption about the sample distribution.*

In this paper, we take full advantage of the geometric structure of the data and thus present a new discriminative tracking approach with landmark-based label propagation (LLP). The LLP locally approximates the soft label of each sample by a linear combination of labels on its nearby landmarks. It is able to effectively propagate a limited amount of initial labels to a large amount of unlabeled samples, matching the needs of discriminative trackers. Under the graph representation of samples, we employ a local landmarks approximation (LLA) method to design a sparse and nonnegative adjacency matrix characterizing relationship among all samples. Based on the Nesterov's gradient projection algorithm, an efficient numerical algorithm is developed to solve the problem of the LLA with guaranteed quadratic convergence. Furthermore, the objective function of the label prediction provides a promising paradigm for modeling the geometrical structures of samples via Laplacian regularizer. Preserving the local manifold structure

Manuscript received April 21, 2014; revised October 8, 2014 and November 29, 2014; accepted February 11, 2015. Date of publication February 19, 2015; date of current version March 6, 2015. This work was supported in part by the 973 Program of China under Grant 2012CB720000, in part by the Natural Science Foundation of China under Grant 61203291, in part by the Specialized Fund for Joint Building Program of Beijing Municipal Education Commission, and in part by the Ministry of Education Tier-1 under Grant M4011272.040. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nilanjan Ray. *(Corresponding author: Mingtao Pei.)*

Y. Wu, M. Pei, M. Yang, and Y. Jia are with the Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, China (e-mail: wuyuwei@bit.edu.cn; peimt@bit.edu.cn; yangminbit@bit.edu.cn; jiayunde@bit.edu.cn).

J. Yuan is with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798 (e-mail: jsyuan@ntu.edu.sg).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2405479

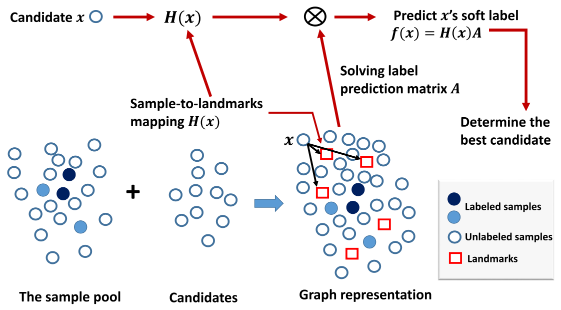


Fig. 1. Landmark-based label propagation for visual tracking. The proposed method treats both labeled and unlabeled samples as vertices in a graph. For each new frame, candidates predicted by particle filter are considered as unlabeled samples and utilized to constitute a new graph representation. The label of each sample is a locally weighted average of the labels on landmarks. Then the classification scores f of candidates are used to construct the observation model of the particle filter to determine the best candidate.

of samples can make our tracker have more discriminating power to handle appearance changes.

Fig. 1 shows the flow diagram of visual tracking using the LLP. Specifically, the proposed method treats both labeled and unlabeled samples as vertices in a graph and builds edges which are weighted by the affinities (similarities) between the corresponding sample pairs. For each new frame, candidates predicted by the particle filter are considered as unlabeled samples and utilized to constitute a new graph representation together with the collected samples stored in the sample pool. A small number of landmarks obtained from the entire sample space enable nonparametric regression that calculates the soft label of each sample as a locally weighted average of labels on landmarks. Tracking is carried out within a Bayesian inference framework where the soft label prediction value is used to construct the observation model. A candidate with the highest classification score is considered as the tracking result. To alleviate the drift problem, once the tracked object is located, the labels of the newly collected samples are assigned according to the classification score of the current tracking results, in which no self-labeling is involved. The proposed tracker adapts to drastic appearance variations, as validated in our experiments.

The remainder of this paper is organized as follows. For the ease of reading, we firstly discuss the related work in Sect. II. In Sect. III, we introduce the landmark-based label propagation method to train an effective classifier. Then the tracking algorithm based on the LLP is presented in Sect. IV. Experimental results and demonstrations are reported and analyzed in Sect. V and the conclusion is given in Sect. VI.

II. Related Work

Discriminative tracking has received wide attention for its adaptive ability to handle appearance changes. In this section, we only discuss the most relevant literature with our method. Interested readers may refer to [2] for a comprehensive review.

The essential component of discriminative trackers is the classifier learning. Many trackers employ online supervised learning methods to train the classifiers. Avidan [16] introduced an ensemble tracking method in which a set of weak classifiers is trained and combined for distinguishing the object and the background. The features used in [16] may contain redundant and irrelevant information which affects the classification performance. Collins *et al.* [17] developed an online feature selection mechanism using the two-class variance ratio to find the most discriminative RGB color combination in each frame. Grabner *et al.* [18] proposed an online boosting feature selection method for visual tracking. However, above-mentioned methods [16]–[18] only utilize *one* positive sample (*i.e.,* the tracking result in the current frame) and multiple negative samples to update the classifier. If the object location is not perfectly detected by the current classifier, the appearance model would be updated with a sub-optimal positive example. Over time the accumulation of errors can degrade the classifier, and can cause drift.

Numerous approaches also apply multiple positive samples and negative samples to train classifiers. Babenko *et al.* [11] integrated multiple instance learning (MIL) into online boosting algorithm to alleviate the drift problem. In the MIL tracker, the classifier is updated with positive and negative bags rather than individual labeled examples. Zhang and Maaten [19] developed a structure-preserving object tracker that learns spatial constraints between objects using an online structured SVM algorithm to improve the performance of single-object or multi-object tracking. Wu *et al.* [20] and Jiang *et al.* [21] addressed visual tracking by learning a suitable metric matrix to effectively capture appearance variations, such that different appearances of an object will be close to each other and be well distinguished from the background.

Discriminative trackers also exploit the semi-supervised learning scheme to address the appearance variations. Grabner *et al.* [9] employed an online semi-supervised learning framework to train a classifier by *only* labeling samples in the first frame and leaving subsequent samples unlabeled. Although this method has shown to be less susceptible to drift, it is not adaptive enough to handle fast appearance changes. Kalal *et al.* [13] developed a P-N learning method to train a binary classifier with structured unlabeled data. Zeisl *et al.* [22] presented a coherent framework which is able to combine both online semi-supervised learning and multiple instance learning.

Recently, researchers utilized the graph-based discriminative learning to construct the object appearance model for visual tracking. Zha *et al.* [23] employed the graph-based transductive learning to capture the underlying geometric structure of samples for tracking. With the 2^{nd}-order tensor representation, Gao *et al.* [24] designed two graphs for characterizing the intrinsic local geometrical structure of the tensor space. Based on the least square support vector machine, Li *et al.* [25] exploited a hypergraph propagation method to capture the contextual information on samples, which further improves the tracking accuracy. Kumar and Vleeschouwer [26] constructed a number of distinct graphs (*i.e.*, spatiotemporal, appearance and exclusion) to capture the spatio-temporal and the appearance information. Then, they formulated the multi-object tracking as a consistent labeling problem in the associated graphs.

In works of [9], [11], [13], and [20], candidates are not used to train a classifier, and therefore the class labels of them are assigned by the previous classifier. Different from these works, in our tracker, candidates are considered as unlabeled samples and utilized to constitute a new graph representation to update the current classifier for each new frame, as illustrated in Fig. 1. Explicitly taking into account the local manifold structure of labeled and unlabeled samples, we introduce a soft label propagation method defined over the graph, which has more discriminating power. In addition, once the tracked object is located, the new training samples are collected both in a supervised and unsupervised way which makes our tracker more stable and adaptive to appearance changes. More details are discussed in Sect. IV.

Our method differs from [23]–[25] both in the graph construction and the label propagation method. Methods in [23]–[25] construct the graph representation using kNN whose computational cost is expensive. In contrast, employing local landmarks approximation, we design a new form of the adjacency matrix characterizing the relationship between all samples. The total time complexity scales linearly with the number of samples. More importantly, our method is an inductive model which can be used to infer the labels of unseen data (*i.e.*, candidates). The label of each sample can be interpreted as the weighted combination of the labels on landmarks. Graph Laplacian is incorporated into the objective function of soft label prediction as a regularizer to preserve the local geometrical structure of samples.

III. Landmark-Based Label Propagation

In this section, we introduce a simple yet effective linear classifier. The core idea of our model is that the label of each sample can be interpreted as the weighted combination of the labels on landmarks. Employing local landmarks approximation, we design a new form of the adjacency matrix characterizing the relationship between all samples. Graph Laplacian is incorporated into the objective function of semi-supervised learning as a regularizer to preserve the local geometrical structure of samples, which makes our model have more discriminative power compared to traditional semi-supervised learning methods.

A. Problem Description

Suppose that we have l labeled samples $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{l}$ and u unlabeled samples $\{\boldsymbol{x}_i\}_{i=l+1}^{l+u}$, where $\boldsymbol{x}_i \in \mathbb{R}^d$, and $\boldsymbol{y}_i \in \mathbb{R}^c$ is the label vector. Since discriminative models take tracking as a binary classification task to separate the object from its surrounding background, the number of classes c equals 2. Denote $\boldsymbol{X} = \{x_1, x_2, \cdots, x_n\} \in \mathbb{R}^{d\times n}$ and $\boldsymbol{Y}_l = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_l\} \in \mathbb{R}^{l\times c}$, where $n = l + u$. If $\boldsymbol{x}_i$ belongs to the kth class ($1 \le k \le c$), the kth entry in $\boldsymbol{y}_i$ is 1 and all the other entries are 0's. In this paper, the data $\boldsymbol{X}$ is represented by the undirected graph $\mathcal{G} = \{\boldsymbol{X}, \boldsymbol{E}\}$, where the set of vertices is $\boldsymbol{X} = \{\boldsymbol{x}_i\}$ and the set of edges is $\boldsymbol{E} = \{e_{ij}\}$, where e_{ij} denotes the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Define a soft label prediction (*i.e.*, classification) function $f : \mathbb{R}^d \to \mathbb{R}^c$. A crucial component of our method is the estimation of a weighted graph $\mathcal{G}$ from $\boldsymbol{X}$. Then, the soft label of any sample can be inferred using $\mathcal{G}$ and known labels Y_l.

The time complexity of traditional graph-based semi-supervised learning methods is usually $O(n^3)$ with respect to the data size n, because $n\times n$ kernel matrix (*e.g.,* multiplication or inverse) is calculated in inferring the label prediction. Since full-size label prediction is infeasible when n is large, the work of [27] inspires us to exploit the idea of landmark samples. To accomplish the soft label prediction, we employ an economical and practical prediction function expressed as

$$f(\boldsymbol{x}) = \sum_{k=1}^{m} K(\boldsymbol{x}, \boldsymbol{d}_k)\boldsymbol{a}_k, \tag{1}$$

where $\boldsymbol{d}_k$ denotes the k-th landmark, $\boldsymbol{a}_k$ is the label of the k-th landmark, and $K(\boldsymbol{x}, \boldsymbol{d}_k)$ represents the cross-similarity weight between the data $\boldsymbol{x}$ and the landmark $\boldsymbol{d}_k$. The idea of Eq. (1) is that the label of each sample can be interpreted as the locally weighted average of variables $\boldsymbol{a}_k$'s defined on m landmarks [27], [28]. As a trade-off between computational efficiency and effectiveness, in this paper, k-means algorithm is used to select the centers as the set of landmarks $\boldsymbol{D} = \{\boldsymbol{d}_k\}_{k=1}^{m} \in \mathbb{R}^{d\times m}$.

Eq. (1) is deemed as a label propagation model, because it can diffuse the label of landmarks to all unlabeled samples, as discussed in Sect. III-D. It avoids optimizing the labels of all the samples, by just concentrating on the labels of the landmarks. Unlike the traditional label propagation method [29], our model takes full advantage of the geometric structure of the data and makes no specific assumption about the sample distribution.

The above model can be written in a matrix form

$$\boldsymbol{F} = \boldsymbol{H}\boldsymbol{A}, \tag{2}$$

where $\boldsymbol{F} = [f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \cdots, f(\boldsymbol{x}_n)]^\top \in \mathbb{R}^{n\times c}$ is the landmark-based label prediction function on all samples. $\boldsymbol{A} = [f(\boldsymbol{d}_1), f(\boldsymbol{d}_2), \cdots, f(\boldsymbol{d}_m)]^\top = [\boldsymbol{A}_1, \boldsymbol{A}_2, \cdots, \boldsymbol{A}_c] \in \mathbb{R}^{m\times c}$ denotes the label of landmarks $\boldsymbol{d}_k$'s. $\boldsymbol{H} \in \mathbb{R}^{n\times m}$ is the cross-similarity matrix between the whole data $\boldsymbol{X}$ and landmarks $\boldsymbol{d}_k$,

$$\boldsymbol{H}_{ik} = K(\boldsymbol{x}_i, \boldsymbol{d}_k) > 0, \quad 1 \le i \le n, \quad 1 \le k \le m.$$

In what follows, we will elaborate how to effectively solve $\boldsymbol{A}$ and $\boldsymbol{H}$.

B. Solving Optimal $\boldsymbol{H}$

Typically, we may employ Gaussian kernel or Epanechnikov quadratic kernel [30] to compute $\boldsymbol{H}$. However, choosing appropriate kernel bandwidths is difficult. Instead of adopting the predefined kernel, we learn an optimal $\boldsymbol{H}$ by considering the geometric structure information between labeled and unlabeled samples. We reconstruct $\boldsymbol{x}_i$ as a combination of its s closest landmarks in the feature space. In this work, we employ the Euclidean distance to select the $s = 10$ closest landmarks for the given sample $\boldsymbol{x}_i$. Recently, Wang *et al.* [31] proposed locality-constrained linear coding (LLC) which uses the locality constraints to project each descriptor into its local-coordinate system [32]. To enhance the coding efficiency, approximated LLC is proposed in [31], in which the locality constraint function is replaced by using the s closest landmarks. For each $\boldsymbol{x}_i$ approximated LLC is defined as

$$\min_{\boldsymbol{h}_i \in \mathbb{R}^s} \left\| \boldsymbol{x}_i - \widetilde{\boldsymbol{D}} \boldsymbol{h}_i \right\|^2, \tag{3}$$

where $\widetilde{\boldsymbol{D}} \in \mathbb{R}^{d \times s}$ is the s closest landmarks of $\boldsymbol{x}_i$.

Inspired by the idea of LLC, our goal is to design a both sparse and optimal cross-similarity matrix $\boldsymbol{H}$ between the whole data $\boldsymbol{X}$ and landmarks $\boldsymbol{D}$. A *Local Landmarks Approximation* (LLA) method is proposed to optimize the coefficient vector $\boldsymbol{h}_i \in \mathbb{R}^s$ for each data point $\boldsymbol{x}_i$, corresponding to the following problem:

$$\begin{aligned} \min_{\boldsymbol{h}_i \in \mathbb{R}^s} g(\boldsymbol{h}_i) &= \frac{1}{2} \left\| \boldsymbol{x}_i - \sum_{j=1}^{s} \boldsymbol{d}_j h_{ij} \right\|^2, \\ s.t.\ \mathbf{1}^\top \boldsymbol{h}_i &= 1, \quad h_{ij} \geq 0 \end{aligned} \tag{4}$$

where h_{ij} is the coefficient activated by the j^{th} nearby landmark of $\boldsymbol{x}_i$. The s entries of the vector $\boldsymbol{h}_i$ correspond to the s coefficients contributed by the s nearest landmarks. The constraint $\mathbf{1}^\top \boldsymbol{h}_i = 1$ follows the shift-invariant requirements. The main difference between LLC and our method is that we incorporate inequality constraints (*i.e.*, non-negative constraints) into the objective function as we require the similarity measure to be a positive value. Therefore we need to develop a different optimization algorithm to solve Eq. (4).

It is easy to see that the constraints set $C = \{\boldsymbol{h}_i \in \mathbb{R}^s : \mathbf{1}^\top \boldsymbol{h}_i = 1,\ h_{ij} \geq 0\}$ is a convex set. Standard quadratic programming (QP) algorithms can be used to solve Eq. (4) but most of them are computationally expensive for computing an approximation of the Hessian. To speed up the convergence rate, Nesterov's gradient projection (NGP) method [33], a first-order optimization procedure, is employed to solve the constrained optimization problem Eq. (4). A key step of NGP is how to efficiently project a vector $\boldsymbol{h}_i$ onto the corresponding constraint set C.

1) Euclidean Projection Onto the Simplex: For simplicity, let $\boldsymbol{v} \in \mathbb{R}^s$ denote the vector which needs to be mapped onto C, and $\boldsymbol{v}'$ be the output. Therefore, the Euclidean projection of $\boldsymbol{v} \in \mathbb{R}^s$ onto C is to solve the following optimization problem:

$$\begin{aligned} \Pi_C(\boldsymbol{v}) &= \arg\min_{\boldsymbol{v}' \in C} \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{v}'\|_2^2 \\ s.t.\ \mathbf{1}^\top \boldsymbol{v}' &= 1, \quad \boldsymbol{v}' \geq 0, \end{aligned} \tag{5}$$

where $\Pi_C(\boldsymbol{v})$ denotes the Euclidean projection operator on any $\boldsymbol{v} \in \mathbb{R}^s$.

The Lagrangian of the problem in (5) is

$$\mathcal{L}(\boldsymbol{v}', \omega) = \frac{1}{2} \|\boldsymbol{v} - \boldsymbol{v}'\|_2^2 + \mu \Big(\sum_{i=1}^{k} \boldsymbol{v}'_i - 1 \Big) - \omega \cdot \boldsymbol{v}', \tag{6}$$

where μ is a Lagrange multiplier and ω is a vector of non-negative Lagrange multipliers. By setting the derivative w.r.t. $\boldsymbol{v}'_i$ to zero, we have

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{v}'_i} = \boldsymbol{v}'_i - \boldsymbol{v}_i + \mu - \omega_i = 0. \tag{7}$$

Algorithm 1 Solving the Euclidean Projection Operator $\Pi_C(\boldsymbol{v})$

```
Input: A vector $v \in \mathbb{R}^s$.
Output: A vector $\Pi_C(v) = v' = [v'_1, v'_2, \cdots, v'_s]^\top$ such that
        $v'_i = \max\{v_i - \mu, 0\}$.
Sort $v$ into $z$ such that $z_1 \geq z_2 \geq \cdots \geq z_k$;
Compute $\rho = \max\left\{ i \in [1:s] :\ z_i - \frac{1}{i}\big(\sum_{r=1}^{i} z_r - 1\big) > 0 \right\}$;
Compute $\mu = \frac{1}{\rho}\big(\sum_{i=1}^{\rho} z_i - 1\big)$.
```

The complementary slackness KKT condition implies that whenever $\boldsymbol{v}'_i > 0$ we have $\omega_i = 0$. Thus, we can get $\boldsymbol{v}'_i = \max\{\boldsymbol{v}_i - \mu, 0\}$, where $\mu = \frac{1}{\rho}\big(\sum_{i=1}^{\rho} z_i - 1\big)$ and $\rho = \max\left\{ i \in [1:s] :\ z_i - \frac{1}{i}\big(\sum_{r=1}^{i} z_r - 1\big) > 0 \right\}$. z denotes the vector obtained by sorting $\boldsymbol{v}$ in a descending order. The projection operator $\Pi_C(\cdot)$ can be implemented efficiently in $O(s \log s)$ [34]. The euclidean projection onto the simplex is summarized in Algorithm 1. For more details, please refer to [34].

2) Nesterov's Gradient Projection (NGP): We use NGP to solve the constrained optimization problem Eq. (4) by adopting the Euclidean projection. Denote

$$\mathcal{Q}_{\beta,\boldsymbol{v}}(\boldsymbol{h}_i) = g(\boldsymbol{v}) + \nabla g(\boldsymbol{v})^\top (\boldsymbol{h}_i - \boldsymbol{v}) + \frac{\beta}{2} \|\boldsymbol{h}_i - \boldsymbol{v}\|_2^2, \tag{8}$$

which is the first-order Taylor expansion of $g(\boldsymbol{h}_i)$ at $\boldsymbol{v}$ with the squared Euclidean distance between $\boldsymbol{h}_i$ and $\boldsymbol{v}$ as a regularization term. Here $\nabla g(\boldsymbol{v})$ is the gradient of $g(\boldsymbol{h}_i)$ at $\boldsymbol{v}$. According to Eq. (5), we can easily obtain

$$\arg\min_{\boldsymbol{h}_i \in C} \mathcal{Q}_{\beta,\boldsymbol{v}}(\boldsymbol{h}_i) = \Pi_C\Big(\boldsymbol{v} - \frac{1}{\beta} \nabla g(\boldsymbol{v}) \Big). \tag{9}$$

From Eq. (9), the solution of Eq. (4) can be obtained by generating a sequence $\{\boldsymbol{h}_i^{(t)}\}$ at $\boldsymbol{v}^{(t)} = \boldsymbol{h}_i^{(t)} + \alpha_t \big(\boldsymbol{h}_i^{(t)} - \boldsymbol{h}_i^{(t-1)}\big)$, *i.e.*,

$$\begin{aligned} \boldsymbol{h}_i^{(t+1)} &= \Pi_C\Big(\boldsymbol{v}^{(t)} - \frac{1}{\beta_t} \nabla g(\boldsymbol{v}^{(t)}) \Big) \\ &= \arg\min_{\boldsymbol{h}_i \in C} \mathcal{Q}_{\beta_t, \boldsymbol{v}^{(t)}}(\boldsymbol{h}_i). \end{aligned} \tag{10}$$

In NGP, choosing proper parameters β_t and α_t is also significant for the convergence property. Similar to [33], we set $\alpha_t = (\delta_{t-1} - 1)/\delta_t$ with $\delta_t = \big(1 + \sqrt{1 + 4\delta_{t-1}^2}\big)/2$, $\delta_0 = 0$ and $\delta_1 = 1$. β_t is selected by finding the smallest nonnegative integer j such that $g(\boldsymbol{h}_i) \leq \mathcal{Q}_{\beta_t, \boldsymbol{v}^{(t)}}(\boldsymbol{h}_i)$ with $\beta_t = 2^j \beta_{t-1}$. In [35], Nesterov states that NGP has a convergence rate $O(1/t^2)$. The convergence property is summarized in Theorem 1. The solving process of Eq. (4) is summarized in Algorithm 2.

Theorem 1: Employing NGP to solve the constrained optimization problem (4) by adopting the Euclidean projection, for any t, we have

$$g(\boldsymbol{s}_i^{(t+1)}) - \min_{\boldsymbol{s}_i \in C} g(\boldsymbol{s}_i) \leq \frac{2\widehat{\beta}_L \|\boldsymbol{s}_i^{(0)} - \boldsymbol{s}_i^*\|_2^2}{(t+1)^2}, \tag{11}$$

where $\widehat{\beta}_L = \max(2\beta_L, \beta_0)$, β_0 is the initial estimation of gradient Lipschitz constant β_L of $g(\boldsymbol{s}_i)$. For $\forall \boldsymbol{s}_i$ and $\forall \boldsymbol{v}$, β_L satisfies $\|\nabla g(\boldsymbol{s}_i) - \nabla g(\boldsymbol{v})\|_2 \leq \beta_L \|\boldsymbol{s}_i - \boldsymbol{v}\|_2$. Besides, the first t steps of the method require t evaluations of $\nabla g(\boldsymbol{s}_i)$ and no more than $2t + \log_2(\widehat{\beta}_L/\beta_0)$ evaluations of $g(\boldsymbol{s}_i)$.

Algorithm 2 Nesterov's Gradient Projection for Solving the Optimal $\boldsymbol{H}$

```
Input: samples set $X \in \mathbb{R}^{d\times n}$ , the set of landmarks $D \in \mathbb{R}^{d\times m}$, the
       number of neighbors of each sample $s$.
Output: $\{H_i\}_{i=1}^{n}$
for $i = 1 \rightarrow n$ do
    for each $x_i$ find $s$ nearest neighbors in $D$;
    initialize $h_i^{(0)} = h_i^{(1)} = \mathbf{1}/s$, $\delta_0 = 0$, $\delta_1 = 1$, $\beta_0 = 1$;
    for $t = 1, 2, \cdots$ do
        $\alpha_t = (\delta_{t-1} - 1)/\delta_t$, $v^{(t)} = h_i^{(t)} + \alpha_t\big(h_i^{(t)} - h_i^{(t-1)}\big)$;
        for $j = 1, 2, \cdots$ do
            $\beta = 2^j \beta_{t-1}$;
            $h_i^{(t)} = \Pi_C\Big(v^{(t)} - \frac{1}{\beta}\nabla g(v^{(t)})\Big)$;
            if $g(h_i) \le \mathcal{Q}_{\beta_t, v^{(t)}}(h_i)$ then
                update $\beta_t = \beta$, $h_i^{(t+1)} = h_i^{(t)}$ ;
                break;
            end
        end
        $\delta_{t+1} = \frac{1+\sqrt{1+4\delta_t^2}}{2}$;
    end
    Compute $H_i$ using $h_i$.
end
```

After getting the optimal weight vector $\boldsymbol{h}_i$, we set $\boldsymbol{H}_{i,\langle i\rangle} = \boldsymbol{h}_i$, where $\langle i\rangle$ is the vector of indices corresponding to the s nearest landmarks and the cardinality $|\langle i\rangle| = s$. For the remaining entries of $\boldsymbol{H}_{i,\overline{\langle i\rangle}}$, we set 0's. Apparently, $\boldsymbol{H}_{ij} = 0$ when landmark $\boldsymbol{d}_j$ is far away from $\boldsymbol{x}_i$ and $\boldsymbol{H}_{ij} \neq 0$ is only for the s closest landmarks of $\boldsymbol{x}_i$. In contrast to weights defined by kernel function (*e.g.*, Gaussian kernel), the LLA is able to provide optimized and sparser weights, as validated in our experiments.

C. Solving Label Prediction Matrix A

Note that the adjacency matrix $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ between all samples encountered in practice usually has low numerical-rank compared to the matrix size [36]. We consider *whether we can construct a nonnegative and empirically sparse graph adjacency matrix* $\boldsymbol{W}$ *with the nonnegative and sparse* $\boldsymbol{H} \in \mathbb{R}^{n\times m}$ *introduced in Sect. III-B*. Interestingly, each row $\boldsymbol{H}_i$ in $\boldsymbol{H}$ can be a new representation of raw sample $\boldsymbol{x}_i$. $\boldsymbol{x}_i \rightarrow \boldsymbol{H}_i$ is reminiscent of *sparse coding* [31] with the basis $\boldsymbol{D}$ since $\boldsymbol{x}_i \approx \widetilde{\boldsymbol{D}}\boldsymbol{h_i} = \boldsymbol{D}\boldsymbol{H}_i$, where $\widetilde{\boldsymbol{D}} \in \mathbb{R}^{d\times s}$ is a sub-matrix composed of s nearest landmarks of $\boldsymbol{x}_i$. That is to say, samples $\boldsymbol{X} \in \mathbb{R}^{d\times n}$ can be represented in the new space, no matter what the original features are. Intuitively, we can design the adjacency matrix $\boldsymbol{W}$ to be a low-rank form

$$\boldsymbol{W} = \boldsymbol{H}\boldsymbol{H}^\top, \tag{12}$$

where the inner product is regarded as the metric to measure the adjacent weight between samples. Eq. (12) implies that if two samples are correlative (*i.e.*, $\boldsymbol{W}_{ij} > 0$), they share at least one landmark, otherwise $\boldsymbol{W}_{ij} = 0$. $\boldsymbol{W}$ defined in Eq. (12) naturally preserves some good properties (*e.g.*, sparseness and nonnegativeness). The effectiveness of $\boldsymbol{W}$ will be demonstrated in Sect. V-E2.

We define the degree of $\boldsymbol{x}_i$ as $\Delta_i = \sum_{j=1}^{n} \boldsymbol{W}_{ij}$. Therefore, the vertex degree matrix of the whole $\mathcal{G}$ is $\boldsymbol{\Delta} = diag(\Delta_1, \Delta_2, \cdots, \Delta_n)$. To compute the label prediction matrix $\boldsymbol{A}$, we exploit the following optimization framework [27]:

$$\min \frac{\eta}{2}\|f\|_{\mathcal{G}} + L(f_l, \boldsymbol{y}_l). \tag{13}$$

The first term $\|f\|_{\mathcal{G}}$ in Eq. (13) enforces the smoothness of f with regard to the manifold structure of the graph, and is formulated as

$$\begin{aligned}
\|f\|_{\mathcal{G}}^2 &= \sum_{i,j=1}^{n} \big\|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_j)\big\|^2 \boldsymbol{W}_{ij} \\
&= \sum_{i,j=1}^{n} \Big(\|f(\boldsymbol{x}_i)\|^2 + \|f(\boldsymbol{x}_j)\|^2 - 2f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)\Big)\boldsymbol{W}_{ij} \\
&= \sum_{i=1}^{n} \|f(\boldsymbol{x}_i)\|^2 \boldsymbol{\Delta}_{ii} + \sum_{j=1}^{n} \|f(\boldsymbol{x}_j)\|^2 \boldsymbol{\Delta}_{jj}, \\
&\quad - 2\sum_{i,j=1}^{n} f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)\boldsymbol{W}_{ij} \\
&= 2\mathbf{Tr}\big(\boldsymbol{F}^\top\boldsymbol{\Delta}\boldsymbol{F} - \boldsymbol{F}^\top\boldsymbol{W}\boldsymbol{F}\big) \\
&= 2\mathbf{Tr}\big(\boldsymbol{F}^\top\boldsymbol{L}\boldsymbol{F}\big)
\end{aligned} \tag{14}$$

where $\boldsymbol{L} = \boldsymbol{\Delta} - \boldsymbol{W}$ is the graph-based regularization matrix $\boldsymbol{L} \in \mathbb{R}^{n\times n}$, and $\mathbf{Tr}(\cdot)$ is a matrix trace operation. Substituting $\boldsymbol{W} = \boldsymbol{H}\boldsymbol{H}^\top$ into Eq. (14), Laplacian graph regularization can be approximated as

$$\boldsymbol{F}^\top\boldsymbol{L}\boldsymbol{F} = \boldsymbol{F}^\top(diag(\boldsymbol{H}\boldsymbol{H}^\top\mathbf{1}) - \boldsymbol{H}\boldsymbol{H}^\top)\boldsymbol{F}, \tag{15}$$

where nonnegative $\boldsymbol{W}$ guarantees the positive semi-definite (PSD) property of $\boldsymbol{L}$. Keeping $\boldsymbol{L}$ PSD is important as it ensures that the graph regularizer $\boldsymbol{F}^\top\boldsymbol{L}\boldsymbol{F}$ is convex.

The second term $L(\cdot,\cdot)$ in Eq. (13) is an empirical loss function, which requires that the prediction f should be consistent with the known class labels. η is a positive regularization parameter. $f_l \in \mathbb{R}^{l\times c}$ is the sub-matrix corresponding to the labeled samples in $f \in \mathbb{R}^{n\times c}$.

By plugging $\boldsymbol{F} = \boldsymbol{H}\boldsymbol{A}$ into Eq. (13) and choosing the loss function $L(\cdot,\cdot)$ as the $L2$-norm, the convex differentiable objective function for solving label prediction matrix $\boldsymbol{A}$ can be formulated as

$$\begin{aligned}
\min_{\boldsymbol{A}} \mathcal{L}(\boldsymbol{A}) &= \eta\ \mathbf{Tr}\big(\boldsymbol{F}^\top\boldsymbol{L}\boldsymbol{F}\big) + \|\boldsymbol{H}_l\boldsymbol{A} - \boldsymbol{Y}_l\|_F^2 \\
&= \eta\ \mathbf{Tr}\big((\boldsymbol{H}\boldsymbol{A})^\top\boldsymbol{L}(\boldsymbol{H}\boldsymbol{A})\big) + \|\boldsymbol{H}_l\boldsymbol{A} - \boldsymbol{Y}_l\|_F^2.
\end{aligned} \tag{16}$$

Here, $\boldsymbol{H}_l \in \mathbb{R}^{l\times m}$ is made up of the rows $\boldsymbol{H}$ that corresponds to the labeled samples, and $\boldsymbol{L}$ is defined in Eq. (15). We easily obtain

$$\frac{\partial\mathcal{L}}{\partial\boldsymbol{A}} = 2\eta\big(\boldsymbol{H}^\top\boldsymbol{L}\boldsymbol{H}\boldsymbol{A}\big) + 2\boldsymbol{H}_l^\top(\boldsymbol{H}_l\boldsymbol{A} - \boldsymbol{Y}_l). \tag{17}$$

By setting the derivative w.r.t. $\boldsymbol{A}$ to zero, the globally optimal solution of Eq. (16) is given by

$$\boldsymbol{A}^* = \big(\boldsymbol{H}_l^\top\boldsymbol{H}_l + \eta\boldsymbol{H}^\top\boldsymbol{L}\boldsymbol{H}\big)^{-1}\boldsymbol{H}_l^\top\boldsymbol{Y}_l. \tag{18}$$

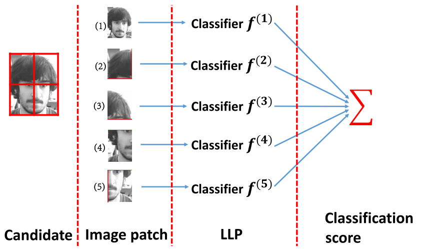


Fig. 2. Object representation using five different image patches. The candidate is normalized to the same size (24 × 24 in our experiment), each image patch is with 12 × 12.

D. Soft Label Propagation

Through applying the label propagation model Eq. (2), we are able to predict the soft label for any sample $\boldsymbol{x}_i$ (unlabeled training samples or new test samples) as

$$\widehat{f}(\boldsymbol{x}_i) = \max_{k \in \{1,2\}} \frac{\boldsymbol{H}(\boldsymbol{x}_i)\ \boldsymbol{A}_k}{\boldsymbol{1}^\top \boldsymbol{H} \boldsymbol{A}_k}, \tag{19}$$

where $\{\boldsymbol{A}_k\}_{k=1}^{c} \in \mathbb{R}^{m\times 1}$ is the column vector of $\boldsymbol{A}$, and $\boldsymbol{H}(\boldsymbol{x}_i) \in \mathbb{R}^{1\times m}$ represents the weight between $\boldsymbol{x}_i$ and landmarks $\boldsymbol{d}_k$'s. Specifically, if $\boldsymbol{x}_i$ belongs to unlabeled training samples, $\boldsymbol{H}(\boldsymbol{x}_i) = \boldsymbol{H}_i$ where $\boldsymbol{H}_i$ denotes the i-th row of $\boldsymbol{H}$, $i = l+1, \cdots, n$. If $\boldsymbol{x}_i$ is a new test sample, we need to compute the vector $\boldsymbol{H}_i$ as $\boldsymbol{H}(\boldsymbol{x}_i)$ described in Algorithm 2, then update $\boldsymbol{H} \in \mathbb{R}^{(n+1)\times m}$, *i.e.*, $\boldsymbol{H} \leftarrow [\boldsymbol{H}; \boldsymbol{H}_i]$. After deriving the soft label prediction (*i.e.*, classification) of each sample, the classification score can be utilized as the similarity measure for tracking. In the next section, we will elaborate the application of the proposed landmark-based label propagation in tracking.

IV. LLP Tracker

In this section, with the landmark-based label propagation introduced in Sect. III, we propose the LLP tracker based on Bayesian inference. In our tracker, the patch-based image representation is able to handle partial occlusion. Once the tracked object is located, the labels of the newly collected samples are determined by the classification score of the current tracking results, in which no self-labeling is involved. This labeling strategy is effective to alleviate the drift problem.

A. Object Representation

In order to potentially alleviate the drift caused by partial occlusions, we employ the part-based scheme to train the classifier in our tracking framework. As a trade-off between computational efficiency and effectiveness, the object is divided into 5 different image patches empirically. That is, an object is represented by five image feature vectors inside the object region. The first patch is the entire object. Then the object is partitioned into 2 × 2 subsets which constitute the 4 remaining patches. These five image patches correspond to the five parts of an object, respectively, as exemplified in Fig. 2. Finally, image patches corresponding to the same part of all samples construct a sub-sample set $\boldsymbol{X}^{(\tau)}$, $\tau = 1, 2, \cdots, 5$. For example, the first patch of all samples constitute the first sub-sample set. Each sub-sample set $\boldsymbol{X}^{(\tau)}$ is used to train a single classifier $f^{(\tau)}$ using the label propagation model previously predefined in Eq. (2). The final tracking result can be determined by the sum of the classification scores of the five image patches inside the object region:

$$SC = \sum_{\tau=1}^{5} \omega_\tau f^{(\tau)}, \tag{20}$$

where ω_τ is the weight of the τ-th image patch ($\sum_{\tau=1}^{5} \omega_\tau = 1$ and $\omega_\tau = 0.2$ in the experiments). This part-based scheme could potentially alleviate the drift caused by partial occlusions.

B. Classifier Initialization

To initialize the classifier in the first frame, we draw positive and negative samples around the object location. Suppose the object is labeled manually, perturbation (e.g., shifting 1 or 2 pixels) around the object is performed for collecting N_p positive samples $\boldsymbol{X}_{N_p}$. Similarly, N_n negative samples $\boldsymbol{X}_{N_n}$ are collected far away from the located object (*e.g.*, within an annular region a few pixels away from the object). $\boldsymbol{X_1} = \boldsymbol{X}_{N_p} \bigcup \boldsymbol{X}_{N_n}$ is the initialized labeled sample set. According to discussion in Sect. IV-A, each sample in $\boldsymbol{X_1}$ is partitioned into 5 different patches. $\boldsymbol{X_1}$ thus contains 5 subsets. The k-means algorithm is used to select the centers as the set of landmarks $\boldsymbol{D}$ in each subset. Using labeled samples and landmarks, we can train prior classifiers via the LLP.

C. Updating the Samples and Landmarks

For each new frame, candidates predicted by the particle filter are considered as unlabeled samples $\widehat{\boldsymbol{X}}$. According to Eq. (19), we can get the classification score of each candidate. A candidate with higher classification score indicates that it is more likely to be generated from the target class. The most likely candidate is considered as the tracking result for this frame. Then, perturbation (*i.e.*, the same scheme in the first frame) around the tracking result is performed for collecting sample set $\boldsymbol{X}_C$. If the classification score of the located object is higher than the predefined threshold ϵ (*i.e.*, the current tracking result is reliable), samples in $\boldsymbol{X}_C$ are regarded as labeled ones, otherwise regarded as unlabeled ones. That is, samples are collected both in a supervised and unsupervised way, and thus the stability and adaptivity in tracking objects of changing appearance are preserved.

To make our tracker more adaptive to appearance changes, we construct a *sample pool* $\boldsymbol{X}_P$ and a *sample buffer pool* $\boldsymbol{X}'$ to update the samples and landmarks, as shown in Fig. 3. We keep a set of T previous $\boldsymbol{X}_C$ to constitute the sample buffer pool $\boldsymbol{X}'$, *i.e.*, $\boldsymbol{X}' = [\boldsymbol{X}_{C-T+1}; \boldsymbol{X}_{C-T+2}; \cdots; \boldsymbol{X}_C]$, where $\boldsymbol{X}_C$ denotes the sample set collected from the current frame. Every T frames, $\boldsymbol{X}'$ is utilized to update $\boldsymbol{X}_P$. After updating the sample pool, we will leave $\boldsymbol{X}'$ empty and then reconfigure it. In our experiment, we set the sample pool capacity $\Theta(\boldsymbol{X}_P)$.[1] If the total number of samples in the sample pool is larger than $\Theta(\boldsymbol{X}_P)$, some samples in $\boldsymbol{X}_P$ will be randomly replaced with samples in $\boldsymbol{X}'$. To reduce the risk of

[1] The cardinality $\Theta(\boldsymbol{X}_P)$ denotes the number of samples in the sample pool.

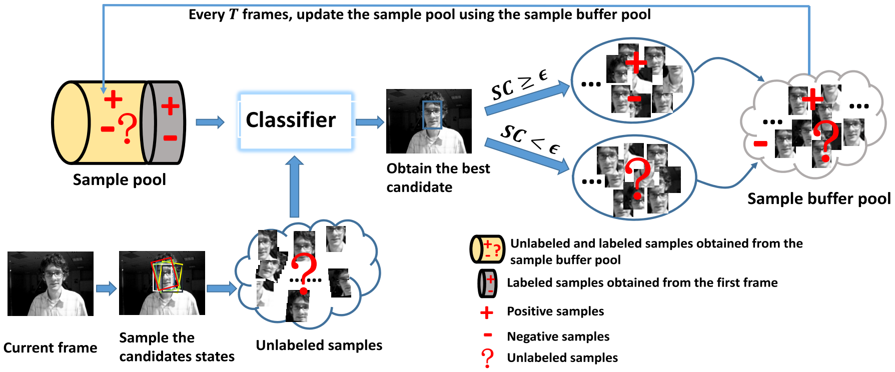

Every T frames, update the sample pool using the sample buffer pool
Sample pool
Classifier
Obtain the best candidate
SC ≥ ε
SC < ε
Sample buffer pool
Current frame
Sample the candidates states
Unlabeled samples
Unlabeled and labeled samples obtained from the sample buffer pool
Labeled samples obtained from the first frame
+ Positive samples
- Negative samples
? Unlabeled samples

Fig. 3. Illustration of constructing the *sample pool* and *sample buffer pool*. For each new frame, candidates cropped by the particle filter are considered as unlabeled samples. Every T frames, the sample pool is updated by the sample buffer pool. After updating, we will leave the sample buffer pool blank and then reconfigure it.

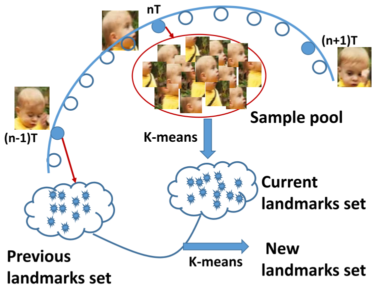


Fig. 4. The set of landmarks updating. The updated set of landmarks is obtained by carrying out twice k-means.

visual drift, we always retain the samples $\boldsymbol{X}_1$ obtained from the first frame in the sample pool. That is, $\boldsymbol{X}_P = [\boldsymbol{X}_1; \boldsymbol{X}']$. Note that candidates are considered as unlabeled samples and utilized to train the classifier together with collected samples stored in the sample pool.

Similarly, landmarks also should be updated using the sample pool every T frames. Specifically, we first implement the k-means algorithm in the current sample pool $\boldsymbol{X}_P$ to obtain a new landmarks set. Then, the updated set of landmarks can be gained by carrying out the k-means algorithm again using both the new and the previous landmarks set which are able to better characterize the samples distribution. The landmarks updating are illustrated in Fig. 4.

D. Bayesian State Inference

Object tracking can be considered as a Bayesian inference task in a Markov model with hidden state variables. Given the observation set of the object $\mathcal{O}_{1:t} = \{\boldsymbol{o_1}, \boldsymbol{o_2}, \cdots, \boldsymbol{o_t}\}$, the optimal state $\boldsymbol{s_t}$ of the tracked object is obtained by the maximum a posteriori estimation $p\left(\boldsymbol{s}_t^i \middle| \mathcal{O}_{1:t}\right)$, where $\boldsymbol{s}_t^i$ indicates the state of the i-th sample. The posterior probability $p\left(\boldsymbol{s}_t \middle| \mathcal{O}_{1:t}\right)$ is formulated by Bayes theorem as $p\left(\boldsymbol{s}_t \middle| \mathcal{O}_{1:t}\right) \propto p(\boldsymbol{o}_t|\boldsymbol{s}_t) \int p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) p\left(\boldsymbol{s}_{t-1} \middle| \mathcal{O}_{1:t-1}\right) \, d\boldsymbol{s}_{t-1}$. This inference is governed by the dynamic model $p\left(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}\right)$ which models the temporal correlation of the tracking results in consecutive frames, and by the observation model $p(\boldsymbol{o}_t|\boldsymbol{s}_t)$ which estimates the likelihood of observing $\boldsymbol{o_t}$ at state $\boldsymbol{s_t}$.

With particle filtering, the posterior $p\left(\boldsymbol{s}_t \middle| \mathcal{O}_{1:t}\right)$ is approximated by a finite set of N_s samples or particles $\left\{\boldsymbol{s}_t^i\right\}_{i=1}^{N_s}$ with importance weights $\left\{\omega_t^i\right\}_{i=1}^{N_s}$. The particle sample $\boldsymbol{s}_t^i$ is drawn from an importance distribution $q(\boldsymbol{s}_t|\boldsymbol{s}_{1:t-1}, \mathcal{O}_{1:t})$, which for simplicity is set to the dynamic model $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$. The importance weight ω_t^i of particle i is equal to the observation likelihood $p(\boldsymbol{o}_t|\boldsymbol{s}_t^i)$. We apply an affine image warp to model the object motion between two consecutive frames. Let $\boldsymbol{s_t} = \{x_t, y_t, \theta_t, s_t, \eta_t, \psi_t\}$, where x_t, y_t, θ_t, s_t, η_t, ψ_t denote x, y translations, rotation angle, scale, aspect ratio and skew at time t, respectively. The state transition distribution $p\left(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}\right)$ is modeled by Brownian motion, *i.e.*, $p\left(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}\right) = \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{s}_{t-1}, \sum)$, where $\sum$ is a diagonal covariance matrix whose diagonal elements are the corresponding variances of respective parameters. The observation model $p(\boldsymbol{o}_t|\boldsymbol{s}_t)$ is defined as

$$p(\boldsymbol{o}_t|\boldsymbol{s}_t) \propto SC_t, \tag{21}$$

where $SC_t = \widehat{f}\left(\boldsymbol{x}^{(t)}\right)$ is the classification score at time t based on Eq. (19). The detailed description of the proposed tracking method is summarized in Algorithm 3.

V. Experiments

We run our tracker on the benchmark dataset [37] including 51 challenging image sequences. The total number of frames on the benchmark is more than 29000. We evaluate the proposed tracker against the 11 state-of-the-art tracking algorithms including ONNDL [38], RET [39], CT [40], VTD [5], MIL [11], SCM [41], Struck [12], TLD [13], ASLSA [3], LSST [4] and SPT [14]. For fair comparisons, we use the source codes provided by the benchmark with the same parameters except ONNDL, RET, LSST and SPT whose parameters of the particle filter are set as in our tracker. Since the trackers involve randomness, we run them 5 times and report the average result for each sequence. The MATLAB source code and experimental results of the 12 trackers are available at http://iitlab.bit.edu.cn/mcislab/~wuyuwei/download.html.

A. Experimental Setup

Note that we fix the parameters of our tracker for all sequences to demonstrate its robustness and stability. The number of particles is 400 and the state transition matrix is $[8, 8, 0.01, 0.001, 0.005, 0]$ in the particle filter. We resize

Algorithm 3 The Proposed Tracking Algorithm

```
Input: Image frames $F_1, F_2, \cdots, F_n$; Object state $\boldsymbol{s}_1$.
Output: Tracking results $\widehat{\boldsymbol{s}}_t$ at time $t$.
for $t = 1 \to n$ do
    if $t == 1$ then
        Obtain labeled samples set $\boldsymbol{X_1} = \boldsymbol{X}_{N_p} \bigcup \boldsymbol{X}_{N_n}$ ;
        Obtain the initial the set of landmarks $\boldsymbol{D}$ with $k$-means;
        Initialize the sample pool $\boldsymbol{X_P} = \boldsymbol{X_1}$;
        Initialize the sample buffer pool $\boldsymbol{X'} = \emptyset$.
    end
    ① Sample the object candidates $\widehat{\boldsymbol{X}}$ as unlabeled samples
    according to the motion model $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$ ;
    ② Let $\boldsymbol{X} = [\boldsymbol{X}_P; \widehat{\boldsymbol{X}}]$;
    ③ Get 5 subsets of $\boldsymbol{X}$ according to the discussion in Sect. IV-A;
    ④ Solve optimal $\boldsymbol{H}$ for each subset using Eq. (4);
    ⑤ Solve label prediction matrix $\boldsymbol{A}$ for each subset using Eq. (18);
    ⑥ Infer the soft label of each candidate using Eq. (19) and get the
    best candidate based on Eq. (21);
    ⑦ Collect training samples set $\boldsymbol{X}_C$ in the current frame;
    if $SC_t \geq \epsilon$ then
        Samples in $\boldsymbol{X}_C$ are regarded as labeled ones;
        else
        Samples in $\boldsymbol{X}_C$ are regarded as unlabeled ones;
    end
    ⑧ $\boldsymbol{X'} = [\boldsymbol{X'}; \boldsymbol{X}_C]$;
    if $mod(t, T) == 0$ then
        Update $\boldsymbol{X}_P$ with $\boldsymbol{X'}$;
        Update the set of landmarks $\boldsymbol{D}$;
        $\boldsymbol{X'} = \emptyset$.
    end
end
```

the object image to 24 × 24 pixels and each image patch is 12 × 12 pixels, as illustrated in Fig. 2. 144 dimensional gray scale feature and 128 dimensional HOG feature are extracted from each image patch, and they are concatenated into a single feature vector of 272 dimensions. In the first frame, $N_p = 20$ positive samples and $N_n = 100$ negative samples are used to initialize the classifier. The predefined threshold of classification score ϵ is set to 0.3. Given the object location at the current frame, if $SC \geq 0.3$, 2 positive samples and 50 negative samples are used for the supervised learning. If $SC < 0.3$, the tracking result is treated as the unreliable one and 100 unlabeled samples are utilized for the unsupervised learning. The sample pool capacity $\Theta(\boldsymbol{X}_P)$ is set to 310, in which the number of positive, negative and unlabeled samples are 50, 160 and 100, respectively. The number of landmarks is set to 30 empirically and the regularization parameter expressed in Eq. (18) is set to $\eta = 0.01$. The set of landmarks $\boldsymbol{D}$ is updated every $T = 10$ frames.

B. Evaluation Criteria

One widely used evaluation method to measure the tracking results is the center location error. It is based on the relative position errors (in pixels) between the central locations of the tracked object and those of the ground truth. Ideally, an optimal tracker is expected to have a small error. However, when the tracker lost the object for some frames, the output location can be random and therefore the average center location errors may not evaluate the tracking performance correctly [37]. In this paper, the *precision plot* is also adopted to measure the overall tracking performance. It shows the percentage of frames whose estimated location is within the given threshold distance (*e.g.*, 20 pixels) of the ground truth. More accurate trackers have higher precision at lower thresholds. If a tracker loses the object it is difficult to reach a higher precision [42].

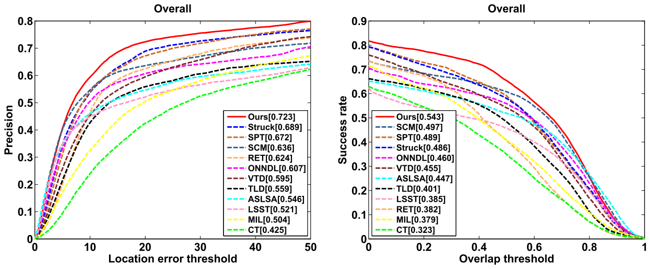


Fig. 5. Overall performance comparisons of precision plot and success rate. The performance score for each tracker is shown in the legend (best viewed on high-resolution display).

The tracking overlap rate indicates stability of each algorithm as it takes the size and pose of the target object into account [43]. It is defined by $score = \frac{area(ROI_T \bigcap ROI_G)}{area(ROI_T \bigcup ROI_G)}$, where ROI_T is the tracking bounding box and ROI_G is the ground truth. This can be used to evaluate the success rate of any tracking approach. Generally, the tracking result is considered as a success when the $score$ is greater than the given threshold t_s. It may not be fair or representative for tracker evaluation using one success rate value at a specific threshold (*e.g.*, $t_s = 0.5$). Further, we count the number of successful frames as the thresholds vary from 0 to 1 and plot the *success rate* curve for our tracker and the compared trackers. The area under curve (AUC) of each success rate plot is employed to rank the tracking algorithms. More robust trackers have higher success rates at higher thresholds.

C. Overall Performance

The overall performance for the 12 trackers is summarized by the precision plot and the success rate on the 51 sequence, as shown in Fig. 5. For precision plots, we use the results at error threshold of 20 pixels for ranking these 12 trackers. The AUC score for each tracker is shown in the legend. In success rate, our tracker is 4.6% above the SCM, and outperforms the Struck by 3.4% in precision plot. SCM, ASLSA and LSST trackers also perform well in success rate, which suggests sparse representations are effective models to account for the appearance change, especially for occlusion. Since the Struck does not handle scale variation, the success rate of Struck is higher than SCM, LSST and ALSA when the overlap threshold t_s is small, but less than SCM, LSST and ASLSA when t_s is large (*e.g.*, $t_s = 0.4$).

Overall, our tracker outperforms the other 11 trackers both in precision plot and success rate. The good performance of our method can be attributed to the fact that the classifier generalizes well on the new data from a limited number of training samples. That is, our method has excellent generalization ability. In addition, the local manifold structure of samples makes the classifier have more discriminating power.

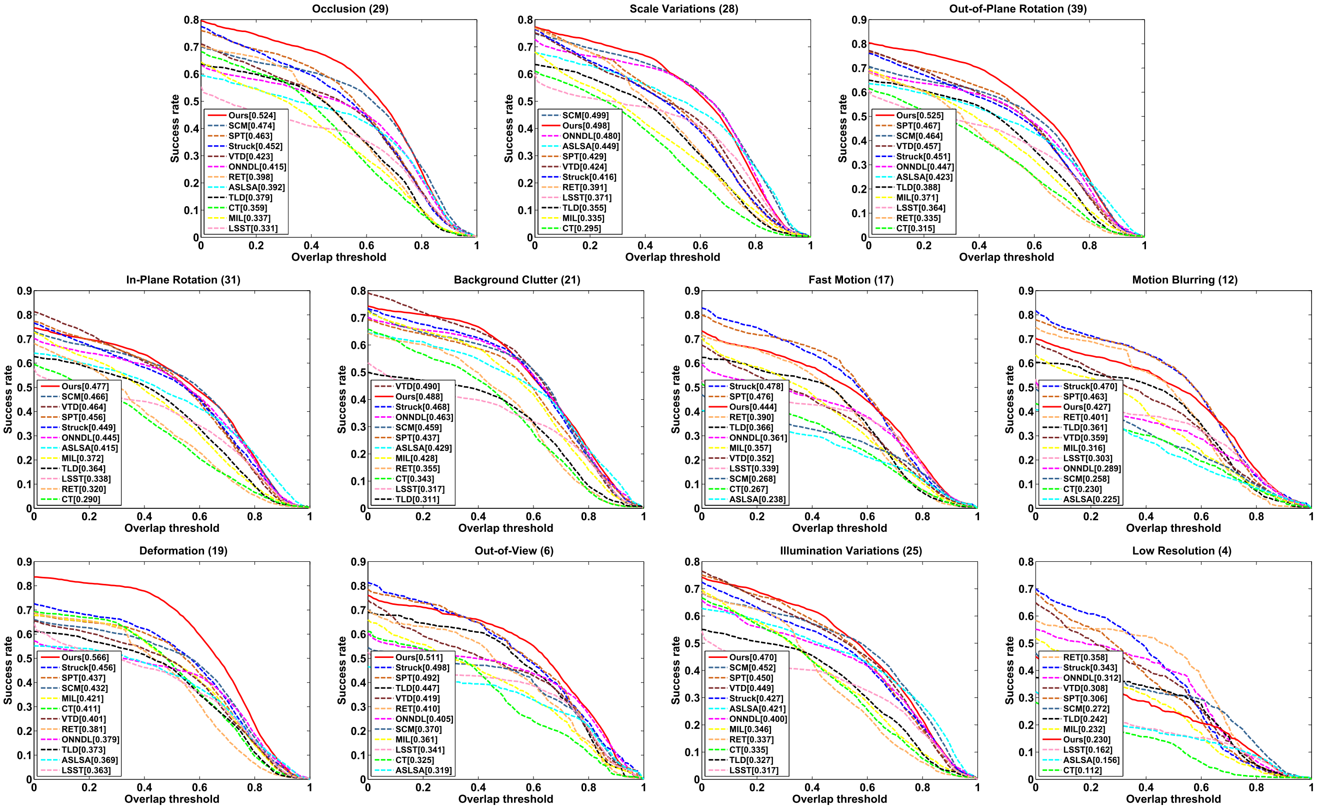


Fig. 6. Attribute-based performance analysis in success rate. The performance score of each tracker is shown in the legend (best viewed on high-resolution display).

D. Attribute-Based Performance

Apart from summarizing the performance on the whole sequences, we also construct the 11 subsets corresponding to different attributes to test the tracking performance under specific challenging conditions. Because the AUC score of the success plot is more accurate than the score at one threshold (*e.g.*, 20 pixels) of the precision plot, in the following we mainly analyze the rankings based on success plots, as shown in Fig. 6.

On the OCC subset, SCM, ASLSA, LSST and our method get better results than others. The results suggest that local image representations are more effective than holistic templates in dealing with occlusions. On the SV subset, we see that trackers with affine motion models (*e.g.*, our method, SCM, ASLSA and LSST) are able to cope with scale variation better than others that only consider translational motion (*e.g.*, Struck and MIL). Similarly, on the OPR and IPR subsets, besides our tracker, the SCM and ASLSA trackers are also able to obtain satisfactory results. The performance of SCM and ASLSA trackers can be attributed to the efficient spare representations of local image patches.

When the object undergoes fast motion and/or motion blur, our method performs worse than the Struck, SPT trackers due to the poor dynamic models in the particle filter. Our tracker can be further improved with more effective state transition matrix of the particle filter. In the LR subset, our tracker does not perform well, because low-resolution objects which are resized to 24 × 24 may not capture sufficient visual information to represent objects for tracking.

E. Diagnostic Analysis

In this section, we analyze two aspects of our landmark-based label propagation that are important for good tracking results, *i.e.*, the weight $\boldsymbol{H}$ between the whole samples and landmarks, and the label prediction matrix $\boldsymbol{A}$.

1) Effectiveness of the Optimal $\boldsymbol{H}$*:* To evaluate the contribution of the optimal $\boldsymbol{H}$ described in Sect. III-B to the overall performance of our tracker, we compute the Nadaraya-Watson kernel regression [30] for comparison. It assigns weights smoothly with

$$\boldsymbol{H}_{ik} = \frac{K_\sigma(\boldsymbol{x}_i, \boldsymbol{d}_k)}{\sum_{j=1}^{m} K_\sigma(\boldsymbol{x}_i, \boldsymbol{d}_j)} \quad 1 \leq i \leq n, \quad 1 \leq j \leq m. \qquad (22)$$

Two kernel functions are exploited in the Nadaraya-Watson kernel regression to measure the cross-similarity matrix between the whole data $\boldsymbol{X}$ and landmarks $\boldsymbol{d}_k$'s. We first adopt Gaussian kernel $K_\sigma(\boldsymbol{x}_i, \boldsymbol{d}_k) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{d}_k\|^2/\sigma^2\right)$ for the kernel regression. Therefore, the corresponding tracking method is called as the *BaseLine1* tracker. Epanechnikov quadratic kernel expressed as

$$K_\sigma(\boldsymbol{x}_i, \boldsymbol{d}_k) = \begin{cases} \frac{3}{4}\left(1 - \|\boldsymbol{x}_i - \boldsymbol{d}_k\|^2\right) & if \quad |\boldsymbol{x}_i - \boldsymbol{d}_k| \leq 1; \\ 0 & otherwise. \end{cases}$$

is also employed for the kernel regression, whose corresponding tracking method is referred to as the *BaseLine2* tracker. We use a more robust way to get σ which uses the nearest neighborhood size s of $\boldsymbol{x}_i$ to replace σ, *i.e.*, $\sigma(\boldsymbol{x}_i) = \|\boldsymbol{x}_i - \boldsymbol{d}_s\|^2$, where $\boldsymbol{d}_s$ is the sth closest landmarks of $\boldsymbol{x}_i$.

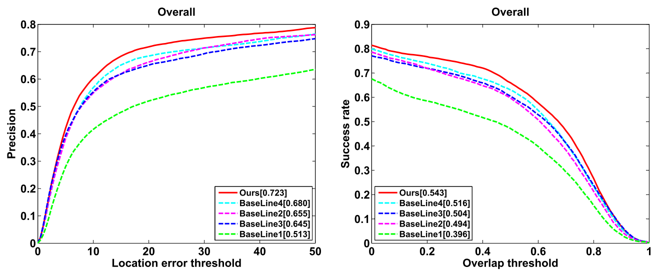


Fig. 7. Diagnostic analysis of our tracker on the 51 sequence. With fixed $\boldsymbol{A}$, *BaseLine1* and *BaseLine2* use different cross-similarity matrix $\boldsymbol{H}$. Similarly, *BaseLine3* and *BaseLine4* use different soft label prediction matrix $\boldsymbol{A}$ with fixed $\boldsymbol{H}$.

The only difference between baseline algorithms and *Ours* is that baseline algorithms utilize the predefined kernel functions to solve cross-similarity matrix $\boldsymbol{H}$ while *Ours* takes advantage of the LLA method to optimize $\boldsymbol{H}$. The overall tracking performance of these two baseline algorithms and our method on the benchmark is presented in Fig. 7. On the whole, our method obtains more accurate tracking results than baseline algorithms.

2) Effectiveness of the Prediction Matrix $\boldsymbol{A}$*:* We design another two baseline algorithms to evaluate the effectiveness of the soft label prediction matrix $\boldsymbol{A}$ described in Sect. III-C. In the *BaseLine3*, we do not consider the Laplacian graph regularizer in Eq. (16), *i.e.*, $\eta = 0$, and thus $\boldsymbol{A}$ becomes the least-squares solution. In the *BaseLine4*, we directly construct the adjacent matrix $\boldsymbol{W}$ using the kNN algorithm instead of $\boldsymbol{W} = \boldsymbol{H}\boldsymbol{H}^\top$. If $\boldsymbol{x}_i$ is among the k-neighbors of $\boldsymbol{x}_j$ or $\boldsymbol{x}_j$ is among the k-neighbors of $\boldsymbol{x}_i$, $\boldsymbol{W}_{ij} = 1$, otherwise, $\boldsymbol{W}_{ij} = 0$. The overall tracking performance on the benchmark is illustrated in Fig. 7. Surprisingly, even without Laplacian graph regularizer, the *BaseLine3* produces the precision score of 0.645 and the success score of 0.504, outperforming the SCM tracker, which implies that the success is due to the framework of the landmark-based label propagation. The overall performance can be further improved using our scheme of solving $\boldsymbol{A}$ described in Sect. III-C.

F. Qualitative Comparisons

1) Significant Pose Variations: Fig. 8 shows tracking results of three challenging sequences with significant pose variations to verify the effectiveness of our method. In the *Basketball* sequence, the object appearance change drastically as the players run side to side, especially for close-fitting defence between players. We see that SPT, CT, RET and SCM trackers are easy to drift at the beginning of the sequence (*e.g.*, ♯60). The TLD, ONNDL, Struck and MIL algorithms drift to another player as the appearance between players in the same team is very similar (*e.g.*, ♯473). VTD, ASLSA and our methods are able to track the whole sequence successfully. We note that the VTD perform better than the other methods. This can be attributed to that the object appearance can be well approximated by multiple basic observation models.

The *Freeman4* sequence is used to test the performance of our method in handling pose changes. There are partial occlusions and scale changes when the object walks toward the camera. Most methods fail to track the object. For example, CT does not manage to get a stable result due to potential randomness. Although TLD has a re-initialization mechanism after occlusion, it locks onto the wrong person as the surrounding background is very similar to the object (*e.g.*, ♯142). In comparison, our method is able to provide a tracking bounding box that is much more accurate and consistent.

In the *Shaking* sequence, the target undergoes illumination change besides pose variations. the Struck, LSST, TLD, CT and RET trackers drift from the object quickly when the spotlight blinks suddenly (*e.g.*, ♯60). SCM, VTD and our trackers are able to successfully track the object throughout the sequence with relatively accurate sizes of the bounding box. SPT, ONNDL, MIL and ASLSA methods are also able to track the object in this sequence but with a lower success rate than our method. In this sequence, the VTD performs better than the other methods.

2) Heavy Occlusion: Fig. 9 shows results from three challenging sequences with heavy occlusion. Images of the *Woman* sequence are acquired by a moving camera and the object color sometimes appears similar to the background clutter. Many methods cannot keep tracking of the object after occlusion. The CT, SCM, MIL, VTD, TLD and ONNDL trackers fail to capture the object after the woman walks behind the white car (*e.g.*, ♯127). The appearance model fuses more background interference due to an occlusion, which significantly influences the samples online updating of the MIL, TLD and ASLSA trackers. The LSST tracker fails gradually over time (*e.g.*, ♯297). Although the RET method tracks well, our method, SPT and Struck trackers achieve more stable performance in the entire sequence.

In the *SUV* sequence, most of the trackers drift when the long-term occlusion happens (*e.g.*, ♯552). Tracking such an object is extremely challenging because the vehicle is almost indistinguishable behind the trees, even for human eyes. Although VTD, SPT and ASLSA trackers take partial occlusion into account, the results are not satisfied. The Struck, RET and ONNDL trackers get slightly better results. In comparisons, our tracker and SCM have relatively lower center location errors and higher success rates. The TLD tracker is equipped with a detection procedure to succeed in tracking after occlusions, which can explain why the TLD tracker obtains relatively high success rate but with high center location error in this sequence.

In the *Liquor* sequence, the object suffers from background clutter besides heavy occlusions for many times. The CT, MIL, LSST and ASLSA trackers drift first when the occlusion occurs (*e.g.*, ♯361). Although the TLD, RET, VTD, SPT and Struck trackers obtain slightly better results than SCM and ONNDL trackers, they lose the object after several occlusions (*e.g.*, ♯733). Overall, our method achieves both the lowest tracking error and the highest overlap rate. The ASLSA and LSST methods are generative models that do not take into account the useful information from the background, and they are not effective in separating two nearby objects with similar appearance. Though the SCM tracker incorporates the discriminative model, its classifier does not update online, making

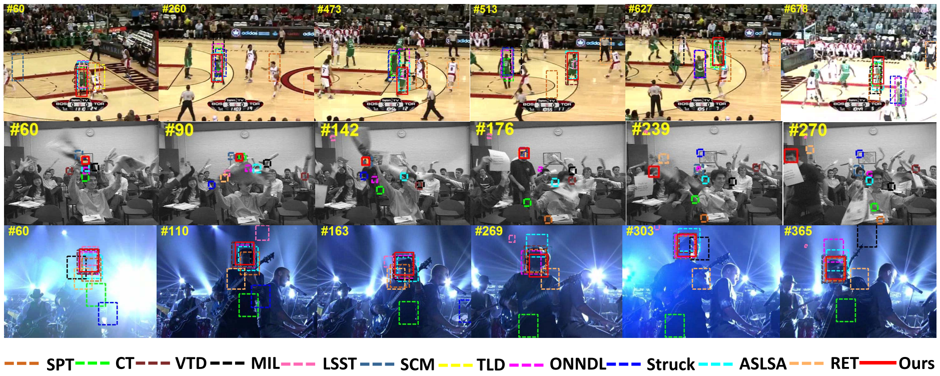


Fig. 8. Qualitative tracking results of the 12 trackers over sequences “Basketball”, “Freeman4” and “Shaking” from top to bottom (best viewed on high-resolution display). Object appearance changes drastically due to large variations of pose.

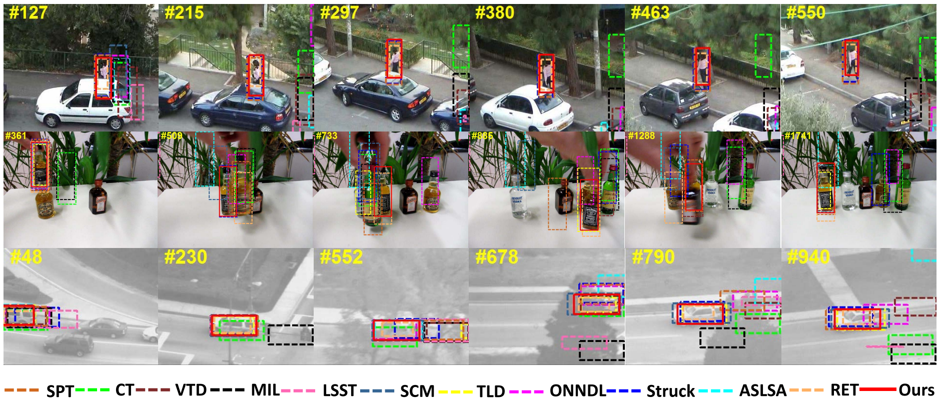


Fig. 9. Qualitative tracking results of the 12 trackers over sequences “Woman”, “Liquor” and “SUV” from top to bottom (best viewed on high-resolution display). Objects undergo heavy occlusion.

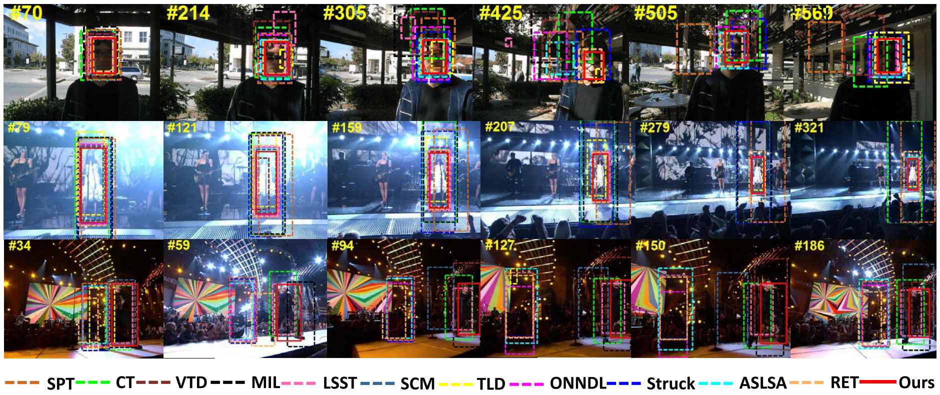


Fig. 10. Qualitative tracking results of the eleven trackers over sequences “Trellis”, “Singer1” and “Singer2” from top to bottom (best viewed on high-resolution display). Objects undergo illumination changes.

it unable to adaptively capture the difference between the object and the background over time. Although the localized Haar-like features used in the MIL and TLD trackers are robust to partial occlusion [11], they cannot perform well in this sequence because of the large scale appearance changes caused by frequent occlusions and background clutter. Our tracker performs well as it assigns the sample labels both in a supervised and unsupervised way during the classifier learning which makes the updated classifier better differentiate the object from the cluttered background.

3) Illumination Changes: Fig. 10 shows tracking results of three challenging sequences to evaluate whether our method is able to tackle drastic illumination changes. In *Trellis* sequence, a man walks under a trellis. Suffering from large changes in environmental illumination and head pose, the CT, TLD, MIL, SPT and LSST trackers drift gradually (*e.g.*, ♯214). In contrast, RET, ASLSA, SCM, Struck and our trackers have relatively high overlap rates. Note that the ASLSA get the best results, which is attributed to the efficient alignment pooling on the sparse coding of local image patches. For the *Singer1*

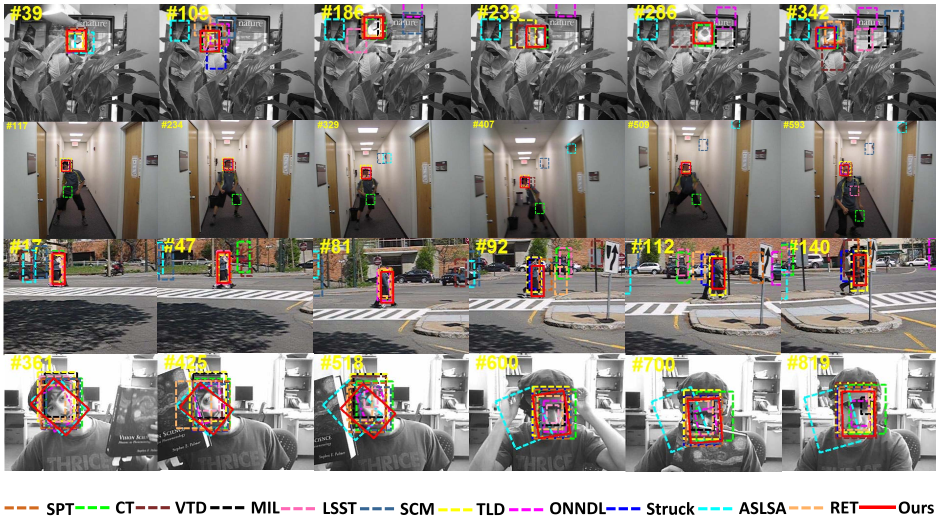


Fig. 11. Qualitative tracking results of the eleven trackers over sequences "Tiger1", "Boy", "Couple" and "FaccOcc2" from top to bottom (best viewed on high-resolution display). The challenges include camera jitter, fast motion, in-plane rotation, occlusion and background clutter, *etc*.

sequence, there are large scale changes of the object and unknown camera motion in addition to illumination change. The SPT tracker gets lost in tracking the object after drastic illumination changes (*e.g.*, ♯121) whereas ONNDL, LSST and RET algorithms perform slightly better. The CT, Struck and MIL trackers perform reasonably well in terms of the center location error but with lower overlap rate, because they can not deal with scale changes well (e.g., ♯207, ♯279 and ♯321). In the *Singer2* sequence, the contrast between the foreground and the background is very low besides illumination change. Most trackers drift away at the beginning of the sequence when the stage light changes drastically (*e.g.*, ♯59). The VTD tracker performs slightly better as the edge feature is less sensitive to illumination change. In contrast, our method succeeds in tracking the object accurately.

Overall, the SCM, ASLSA and our trackers obtains the relatively robust tracking results in the presence of illumination changes. The reason that these three methods perform well can be explained as follows. In SCM and ASLSA trackers, part-based sparse representations with pooling strategy are less sensitive to illumination and pose change, thereby achieving good tracking performance. Our tracker uses an online update mechanism to account for the appearance variations of the object and background over time. More importantly, with the graph representation, our tracker provides a promising paradigm for modeling the manifold structures of samples, which makes the classifier have more discriminating power. Therefore, our tracker is more adaptive to handle appearance changes.

4) Other Challenges: Fig. 11 presents the tracking results where the objects suffer other challenges including motion blur, rotation and scale, *etc*. In the *Tiger1* sequences, the appearances of the object change significantly as a result of scale, pose variation, illumination change and motion blur at the same time. The LSST and ASLSA trackers drift to the background at the beginning of this sequence (*e.g.*, ♯39). The ONNDL, TLD, MIL, VTD and SCM fail gradually when the object frequently undergoes occlusion and pose changes (*e.g.*, ♯180, ♯233). In comparisons, the CT, Struck, RET, SPT and our methods track the object well until the end of this sequence. In the Struck, RET, SPT and our trackers, the discriminative appearance models are updated in an online manner, which take into account the difference between the foreground and the background over time and thereby alleviating the drift problem. Note that the CT tracker gets the best results as it effectively selects the most discriminative random features for updating the classifier, thereby better handling drastic appearance change in this sequence.

In the *Boy* sequences, a boy jumps irregularly where the object undergoes fast motion and out-of-plane. It is difficult to predict their locations. Most methods achieve relatively lower center location errors and higher success rates except CT, SCM and ASLSA trackers. As demonstrated in Fig. 6, SCM and ASLSA trackers do not perform well in this sequence as the drastic appearance changes cause by fast motion and/or motion blur, are not effectively accounted for the sparse representation.

The object in the *Couple* sequence is difficult to track as it moves through the scene with camera jitter and partial occlusion. The TLD, Struck, MIL and our trackers perform well with higher success rates and lower location errors. While the ONNDL, RET and SPT methods perform better than the CT, SCM, ASLSA, VTD and LSST trackers, they all lose the object when occlusion occurs (*e.g.*, ♯112).

In the sequence *FaceOcc2*, the object undergoes in-plane rotation and frequent occlusions. The MIL, RET and TLD trackers fail after the object suffers from the partial occlusion (*e.g.*, ♯425). Struck, ASLSA, LSST and ONNDL are slightly better but gradually drifts after frequent occlusion (*e.g.*, ♯600). Though CT, VTD, and SPT trackers are able to keep track of the object to the end, SCM and our methods achieve both the lowest tracking error and the highest overlap rate.

G. Computational Complexity

The most time consuming part of our tracking algorithm is the computation of the label prediction function $\boldsymbol{f}$.

Specifically, the time complexity of seeking m landmarks using k-means clustering is $\mathcal{O}(mn)$ where n is the number of samples. The time complexity of solving the optimal $\boldsymbol{H}$ and the prediction matrix $\boldsymbol{A}$ is $\mathcal{O}(smn)$ and $\mathcal{O}(m^3 + m^2n)$, respectively, where s is the number of nearest landmarks of each sample. We use a fixed number $m \ll n$ of landmarks for calculating $\boldsymbol{f}$, which is independent of the sample size n. Thus, the total time complexity is $\mathcal{O}(m^2n)$ which scales linearly with the n. The proposed approach was implemented in MATLAB on a Intel Core2 2.5 GHz processor with 4GB RAM. Our tracker is about 1.5 frame/sec for all experiments. No code optimization is performed.

VI. Conclusion

In this paper, we have proposed the landmark-based label propagation for visual tracking, in which the label of each sample can be interpreted as the weighted combination of labels on landmarks. Through solving the cross-similarity matrix $\boldsymbol{H}$ and the label prediction matrix $\boldsymbol{A}$, our model is able to effectively propagate a limited number of landmarks' labels to all the unlabeled candidates, matching the needs of the discriminative tracker. Explicitly considering the local geometrical structure of the samples, the graph-based regularizer is incorporated into the LLP tracker, which makes our method have better discriminating power and thus is more adaptive to handle appearance changes. Comparison with 11 state-of-the-art tracking methods on the benchmark dataset have demonstrated that the LLP tracker is more robust to illumination changes, pose variations and partial occlusions, *etc.*

Acknowledgements

The authors would like to thank both the editor and the reviewers for the invaluable comments and suggestions that help a great deal in improving the manuscript. The authors would like to thank Yang He for his efforts on conducting experiments.

References

[1] S. Salti, A. Cavallaro, and L. Di Stefano, "Adaptive appearance modeling for video tracking: Survey and evaluation," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4334–4348, Oct. 2012.

[2] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. van den Hengel, "A survey of appearance models in visual object tracking," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, 2013, Art. ID 58.

[3] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1822–1829.

[4] D. Wang, H. Lu, and M.-H. Yang, "Least soft-threshold squares tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2371–2378.

[5] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1269–1276.

[6] X. Mei and H. Ling, "Robust visual tracking using $\ell 1$ minimization," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1436–1443.

[7] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.

[8] B. Zhuang, H. Lu, Z. Xiao, and D. Wang, "Visual tracking via discriminative sparse similarity map," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1872–1881, Apr. 2014.

[9] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. 10th Eur. Conf. Comput. Vis.*, 2008, pp. 234–247.

[10] M. Yang, J. Yuan, and Y. Wu, "Spatial selection for attentional visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2007, pp. 1–8.

[11] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[12] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.

[13] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.

[14] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel, "Part-based visual tracking with online latent structural learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2363–2370.

[15] Q. Yu, T. B. Dinh, and G. Medioni, "Online tracking and reacquisition using co-trained generative and discriminative trackers," in *Computer Vision*. Berlin, Germany: Springer-Verlag, 2008, pp. 678–691.

[16] S. Avidan, "Ensemble tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 261–271, Feb. 2007.

[17] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1631–1643, Oct. 2005.

[18] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vis. Conf.*, 2006, vol. 1, no. 5, p. 6.

[19] L. Zhang and L. J. P. van der Maaten, "Preserving structure in model-free tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 756–769, Apr. 2014.

[20] Y. Wu, B. Ma, M. Yang, Y. Jia, and J. Zhang, "Metric learning based structural appearance model for robust visual tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 5, pp. 865–877, May 2014.

[21] N. Jiang, W. Liu, and Y. Wu, "Learning adaptive metric for robust visual tracking," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2288–2300, Aug. 2011.

[22] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof, "On-line semi-supervised multiple-instance boosting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, p. 1879.

[23] Y. Zha, Y. Yang, and D. Bi, "Graph-based transductive learning for robust visual tracking," *Pattern Recognit.*, vol. 43, no. 1, pp. 187–196, 2010.

[24] J. Gao, J. Xing, W. Hu, and S. Maybank, "Discriminant tracking using tensor representation with semi-supervised improvement," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1569–1576.

[25] X. Li, C. Shen, A. Dick, and A. van den Hengel, "Learning compact binary codes for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2419–2426.

[26] K. C. A. Kumar and C. De Vleeschouwer, "Discriminative label propagation for multi-object tracking with sporadic appearance features," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2000–2007.

[27] K. Zhang, J. T. Kwok, and B. Parvin, "Prototype vector machine for large scale semi-supervised learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 1233–1240.

[28] W. Liu, J. Wang, and S.-F. Chang, "Robust and scalable graph-based semisupervised learning," *Proc. IEEE*, vol. 100, no. 9, pp. 2624–2638, Sep. 2012.

[29] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CALD-02-107, 2002.

[30] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York, NY, USA: Springer-Verlag, 2009.

[31] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 3360–3367.

[32] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran & Associates Inc., 2009, pp. 2223–2231.

[33] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*, vol. 87. New York, NY, USA: Springer-Verlag, 2004.

[34] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 272–279.

[35] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $O(1/k^2)$," *Soviet Math. Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

[36] C. K. I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2001.

[37] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.

[38] N. Wang, J. Wang, and D.-Y. Yeung, "Online robust non-negative dictionary learning for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 657–664.

[39] Q. Bai, Z. Wu, S. Sclaroff, M. Betke, and C. Monnier, "Randomized ensemble tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2040–2047.

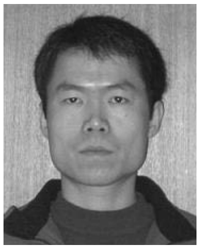
[40] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 864–877.

[41] W. Zhong, H. Lu, and M. Yang, "Robust object tracking via sparsity-based collaborative model," *IEEE Trans. Image Process.*, vol. 23, no. 5, pp. 2356–2368, May 2014.

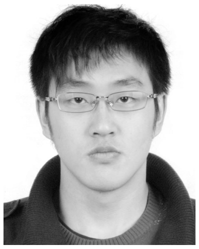
[42] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. 12th Eur. Conf. Comput. Vis.*, 2012, pp. 702–715.

[43] T. Nawaz and A. Cavallaro, "A protocol for evaluating video trackers under real-world conditions," *IEEE Trans. Image Process.*, vol. 22, no. 4, pp. 1354–1361, Apr. 2013.

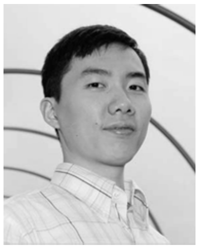
Yuwei Wu received the Ph.D. degree in computer science from the Beijing Institute of Technology (BIT), Beijing, China, in 2014. He is currently a Research Fellow with the Rapid-Rich Object Search Laboratory, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He has strong research interests in computer vision, medical image processing, and object tracking. He received the National Scholarship for Graduate Students and the Academic Scholarship for the Ph.D. Candidates from the Ministry of Education in China, the Outstanding Ph.D. Thesis Award and the XU TELI Excellent Scholarship from BIT, and the CASC Scholarship from the China Aerospace Science and Industry Corporation.



Mingtao Pei received the Ph.D. degree in computer science from the Beijing Institute of Technology (BIT), in 2004. He served as an Associate Professor with the School of Computer Science, BIT. He was a Visiting Scholar with the Center of Image and Vision Science, University of California at Los Angeles, from 2009 to 2011. His main research interest is computer vision with an emphasis on event recognition and machine learning. He is a member of the China Computer Federation.



Min Yang received the B.S. degree from the Beijing Institute of Technology (BIT), in 2010. He is currently pursuing the Ph.D. degree with the Beijing Laboratory of Intelligent Information Technology, School of Computer Science, BIT, under the supervision of Prof. Y. Jia. His research interests include computer vision, pattern recognition, and machine learning.



Junsong Yuan (M'08) received the Ph.D. degree from Northwestern University, USA, and the M.Eng. degree from the National University of Singapore. Before that, he graduated from Special Class for the Gifted Young of Huazhong University of Science and Technology, China. He is currently a Nanyang Assistant Professor and the Program Director of Video Analytics with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He has authored over 100 technical papers, and hold three U.S. patents and two provisional U.S. patents. His research interests include computer vision, video analytics, large-scale visual search and mining, and human–computer interaction.

He received the Nanyang Assistant Professorship and the Tan Chin Tuan Exchange Fellowship from Nanyang Technological University, the Outstanding EECS Ph.D. Thesis Award from Northwestern University, the Best Doctoral Spotlight Award from the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009), and the National Outstanding Student Award from the Ministry of Education, China. He served as an Area Chair of the IEEE Winter Conference on Computer Vision in 2014, the IEEE Conference on Multimedia Expo (ICME 2014), and the Asian Conference on Computer Vision (ACCV 2014), the Organizing Chair of ACCV 2014, and the Co-Chair of workshops at CVPR 2012 and 2013, and the IEEE Conference on Computer Vision in 2013. He serves as an Associate Editor of *The Visual Computer* journal, and the *Journal of Multimedia*. He recently gives tutorials at the IEEE ICIP13, FG13, ICME12, SIGGRAPH VRCAI12, and PCM12.

Yunde Jia (M'11) received the B.S., M.S., and Ph.D. degrees in mechatronics from the Beijing Institute of Technology (BIT), China, in 1983, 1986, and 2000, respectively. He is currently a Professor of Computer Science with BIT, where he currently serves as the Director of the Beijing Laboratory of Intelligent Information Technology. He was a Visiting Scientist with Carnegie Mellon University, USA, from 1995 to 1997, and a Visiting Fellow with Australian National University, Australia, in 2011. He served as the Executive Dean of the School of Computer Science with BIT from 2005 to 2008. His current research interests include computer vision, media computing, and intelligent systems.