Robust Object Tracking with Online Multi-lifespan Dictionary Learning

Junliang Xing, Jin Gao, Bing Li, Weiming Hu
National Laboratory of Pattern Recognition
Institute of Automation, CAS
Beijing 100190, P. R. China
{jlxing,jgao10,bli,wmhu}@nlpr.ia.ac.cn

Shuicheng Yan
Dept. of Electrical and Computer Engineering
National University of Singapore
Singapore 117576, Singapore
eleyans@nus.edu.sg

Abstract

Recently, sparse representation has been introduced for robust object tracking. By representing the object sparsely, i.e.*, using only a few templates via ℓ_1-norm minimization, these so-called ℓ_1-trackers exhibit promising tracking results. In this work, we address the object template building and updating problem in these ℓ_1-tracking approaches, which has not been fully studied. We propose to perform template updating, in a new perspective, as an online incremental dictionary learning problem, which is efficiently solved through an online optimization procedure. To guarantee the robustness and adaptability of the tracking algorithm, we also propose to build a multi-lifespan dictionary model. By building target dictionaries of different lifespans, effective object observations can be obtained to deal with the well-known drifting problem in tracking and thus improve the tracking accuracy. We derive effective observation models both generatively and discriminatively based on the online multi-lifespan dictionary learning model and deploy them to the Bayesian sequential estimation framework to perform tracking. The proposed approach has been extensively evaluated on ten challenging video sequences. Experimental results demonstrate the effectiveness of the online learned templates, as well as the state-of-the-art tracking performance of the proposed approach.*

1. Introduction

Visual object tracking, which aims to estimate the target state (*e.g.* position and size) in video sequences, is a very important research topic in the computer vision field. It has many applications like visual surveillance, vehicle navigation and human computer interaction [25]. Although many algorithms have been proposed in the last decades, object tracking still remains a very challenging problem for real-world applications due to the difficulties like background cluttering, image noises, illumination changes, object occlusions and fast motions, as shown in Figure 1.

Recently, sparse coding based methods have been successfully applied to visual tracking problem [19, 20, 5, 28, 13]. The basic assumption in these methods is that the target can be represented as a linear combination of only a few elements in a template set. Then the confidence of a target candidate can be modeled using the reconstruction error of the sparse representation. Denoting a template set of the target with m elements as $\mathbf{T} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_m] \in \mathbb{R}^{n \times m}$, and a target candidate as $\mathbf{y} \in \mathbb{R}^n$, which is used as the observations to estimate the object state $\mathbf{x}$ (see Section 3.3 for more details), the sparse representation of the candidate using the templates is obtained by solving

$$\min_{\mathbf{c}} \|\mathbf{T}\mathbf{c} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{c}\|_1, \tag{1}$$

where $\mathbf{c} \in \mathbb{R}^m$ is the coefficient vector for the sparse representation, and λ is the regularization parameter to control the sparsity of $\mathbf{c}$. Based on the derived result, the confidence of the candidate $\mathbf{y}$ with respect to the target state $\mathbf{x}$ is naturally modeled as,

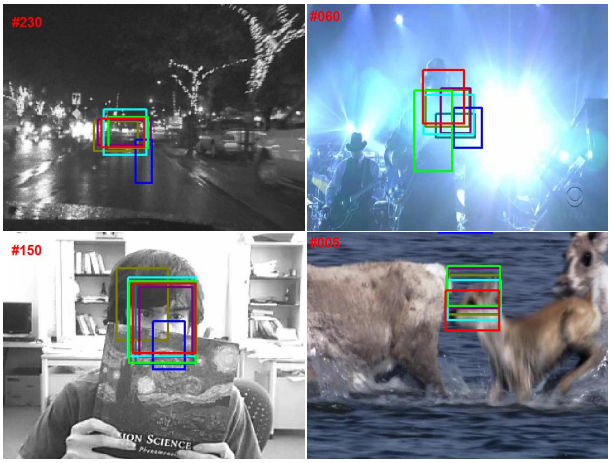


Figure 1. **Typical difficulties in object tracking problem**. (1) Image noises and background cluttering (top-left image), (2) illumination changes (top-right image), (3) object occlusions (bottom-left image), and (4) fast object motions (bottom-right image). The tracking results using fixed templates, fully updated templates, update method in [19], [26], [22], [13] and the proposed method in this work are respectively plotted in teal, olive, purple, cyan, blue, green and red colors. Best viewed in original color PDF file.

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{\Gamma}\exp(-\alpha\|\mathbf{Tc}-\mathbf{y}\|_2^2), \quad (2)$$

where α is a constant and Γ is a normalization factor to make $p(\mathbf{y}|\mathbf{x})$ a valid probability distribution.

Built on this idea, previous works on sparse coding based object tracking have mainly focused on two problems. One is how to design the object templates. Methods using global object templates [19, 26], local image patches [13], or a combination of them [28] have been proposed. The other problem is how to solve the minimization problem to perform efficient tracking, and relevant solutions include Interior Point method [19], Accelerated Proximal Gradient approach [5], Least Angle Regression algorithm [28, 13], and Augmented Lagrange Multiplier method [26].

Supposing that we have already designed the object templates and used them to track the target via a certain optimization procedure, how can we update the object templates effectively and efficiently? This is also a very important problem but has not been paid much attention in previous ℓ_1-trackers. Existing ℓ_1-trackers address the template update problem usually by adopting some classic methods (*e.g*. fixed templates with no updates [28] or incremental update with replacing [19, 27]) or some intuitive strategies [13, 26]. In this paper, we propose to perform the template update problem in the tracking scenario as an online incremental dictionary learning problem. We also explore the temporal nature of the object templates and propose to learn a multi-lifespan dictionary to improve the adaptability and robustness of a tracking algorithm. We apply the online multi-lifespan dictionary learning model into the Bayesian sequential estimation framework and design effective observation models both generatively and discriminatively for a particle filter implementation. Extensive experiments on public benchmark video sequences demonstrate the effectiveness of our online learned templates, and the state-of-the-art tracking performance of the proposed approach.

2. Related Work

Object tracking methods can be roughly grouped into two categories: generative and discriminative. Generative methods, which use a descriptive appearance model to represent the object, perform tracking as a searching problem over candidate regions to find the most similar one. Examples of generative tracking methods are eigentracker [6], mean shift tracker [7], and fragment tracker [1]. Discriminative methods formulate object tracking as a binary classification problem and find the location that can best separate the target from the background. Examples of discriminative tracking methods are online boosting tracker [11], ensemble tracker [3], and multi-instance learning tracker [4].

With recent advances in sparse representation, sparse coding based object trackers demonstrate to be a promising tracking framework [19, 20, 5, 28, 13, 27, 26]. In the ℓ_1-tracker first proposed by Xue and Lin [19], a combination of object templates and trivial templates is employed to tackle the occlusion problem. To make the templates directly robust to occlusion, local image patches can be used as the object dictionary [13, 28]. In order to improve the efficiency of the optimization process, Xue and Lin [20] further propose a minimal error bounded strategy is to reduce the number of the ℓ_1-norm related minimization problem, and later some other efficient optimization procedures are further adopted to solve the problem [5, 28, 13, 26].

Table 1. Comparison of object templates in popular ℓ_1-trackers.

Method	Efficiency	Building	Updating	Robustness	Adaptivity
Xue and Lin[19]	Low	Manually	Intuitively	Low	Low
Xue and Lin[20]	Low	Manually	Intuitively	Low	Low
Bao *et al*. [5]	**High**	Manually	Intuitively	Low	Low
Zhang *et al*. [27]	Low	Manually	Intuitively	Low	**High**
Zhang *et al*. [26]	**High**	Manually	Intuitively	Low	**High**
Jia *et al*. [13]	**High**	**Learned**	**Learned**	Low	**High**
Zhong *et al*. [28]	**High**	Manually	Intuitively	**High**	Low
Proposed Method	**High**	**Learned**	**Learned**	**High**	**High**

Besides template design and optimization method, template update is even a more important problem in the sparse coding based tracking framework. Although fixed object template may work well for short video sequences when the target stays nearly unchanged, it may be incompetent for long sequences where the target often undergoes different kinds of changes (see Figure 1). To adapt to appearance changes of the target during tracking, the templates in [19, 20, 5, 26] are updated based on the weights assigned to each template, and the similarities between templates and the current estimation of the target candidate. In [28], the global object templates are kept fixed during tracking to ensure the discriminative power of the model, while the local patch templates are constantly updated to adapt to object changes. In order to incrementally update the templates, a more reasonable strategy can be found in [13], where old templates are given slow update probabilities and the incremental subspace learning algorithm in [22] is employed which restricts the template vectors to be orthogonal.

Table 1 gives a comparative summary of the object templates used in most popular ℓ_1 trackers. Although all these methods provide different strategies contributing to template update, most of them use some predefined templates and update them intuitively, which may not fully unleash the potential capability of templates. The proposed online learning based template building and updating algorithm, which is both robust and adaptive for tracking, well addresses the problems of object templates in the ℓ_1 trackers.

3. Proposed Approach

Given the object tracking results, the main objective of this paper is to online automatically learn "good" object templates, which can, in turn, benefit the ongoing object tracking process with improved robustness and adaptability. Our core idea to achieve this objective is trying not to impose heavy constraints on the template building and

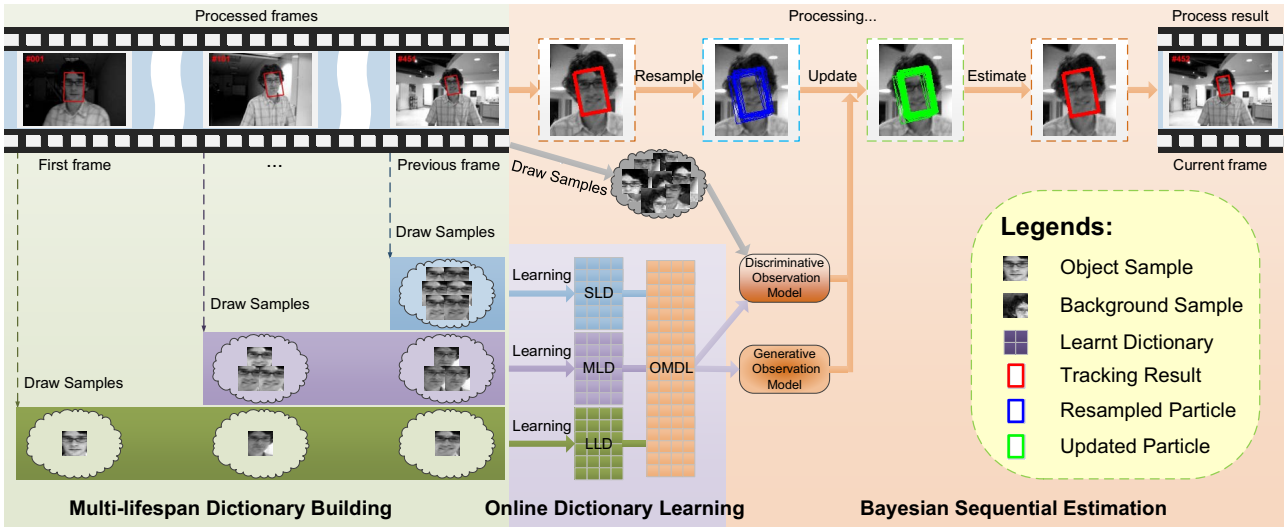


Figure 2. **Overall description of the proposed approach**. We perform template update as an online dictionary learning problem and propose to learn a multi-lifespan dictionary, *i.e*. the Short Lifespan Dictionary (SLD), the Middle Lifespan Dictionary (MLD) and the Long Lifespan Dictionary (LLD), to model the object. The SLD learned from samples densely collected from the previous frame makes the best adaptation to the target. The LLD learned from the most accurate samples collected from all the frames ensures the robustness of the model. The MLD, between the SLD and the LLD, balances the adaptability and robustness of the final model. The Online Multi-lifespan Dictionary Learning (OMDL) model, together with the background samples collected in the previous frame, is used to deduce observation models both generatively and discriminatively. These observation models are then applied into the Bayesian sequential estimation framework using particle filter implementation to perform adaptive and robust object tracking.

updating process and making them most suitable for tracking. To this end, we formulate this template building and updating problem as online dictionary learning, which automatically updates object templates that can better adapt to the data for tracking. In order to further improve the robustness and adaptability of the learned templates, we explore the temporal property of the learned dictionary and propose to build a dictionary with multiple lifespans to possess distinct temporal properties. Based on the learned multi-lifespan dictionary, we deduce effective observation models both generatively and discriminatively, and deploy them into the Bayesian sequential estimation framework to perform tracking using a particle filter implementation. Note that Li *et al*. [15] also use multiple detectors with different lifespans. Their objective is to improve the computation efficiency of the multi-lifespan detectors which are used sequentially in a cascade particle filter. Our multi-lifespan model, however, mainly aims to ease the contradiction between the adaptivity and robustness of template based object tracking algorithms, and the multi-lifespan dictionaries are fused in parallel in a multi-state particle filter. Figure 2 gives the overall description of our approach, the details of which are elaborated in the following subsections.

3.1. Tracking as Online Dictionary Learning

In sparse coding based tracking algorithms, a target candidate is represented as a linear combination of a few elements from a template set. The building and updating of this template set, therefore, have great impact on the final tracking results. Previous works usually build this template set by directly sampling from the initialization of tracking, and then use some intuitive strategies to update the set during tracking [19, 28, 13]. From a different viewpoint, here we want to automatically learn this template set to make it best adapt to the video data to be tracked. We do not want to impose any constraints on the learned templates, but only expect that they can better represent the target that have been tracked and will be tracked.

Suppose all the possible target candidates are within a template set $\mathcal{Y}=\{\mathbf{y}_1,\ldots,\mathbf{y}_N\}$, where $\mathbf{y}_i\in\mathbb{R}^n$ denotes one n-dimensional sample and N is the template size which can be very large, since samples are continually obtained when a new frame has been tracked. A good template set should then have the minimal cost to represent all the elements in this set. Denote the representation cost function as:

$$f(\mathbf{D}) \triangleq \frac{1}{|\mathcal{Y}|}\sum_{\mathbf{y}\in\mathcal{Y}} l(\mathbf{y},\mathbf{D}), \tag{3}$$

where $\mathbf{D}\in\mathbb{R}^{n\times m}$ is the learned template set, distinguished from the predefined template set $\mathbf{T}$ as in previous works, and $l(\cdot)$ is the loss function such that $l(\mathbf{y},\mathbf{D})$ is small if $\mathbf{D}$ is "good" at representing the candidate $\mathbf{y}$. In a sparse coding based object tracking framework, the loss function can be naturally modeled as:

$$l(\mathbf{y},\mathbf{D}) \triangleq \min_{\mathbf{c}\in\mathbb{R}^m} \frac{1}{2}\|\mathbf{y}-\mathbf{D}\mathbf{c}\|_2^2 + \lambda\|\mathbf{c}\|_1. \tag{4}$$

To prevent the ℓ_2-norm of $\mathbf{D}$ from being arbitrarily large, which may lead to arbitrarily small values of $\mathbf{c}$, it usually constrains its columns $\mathbf{d}_1,\ldots,\mathbf{d}_m$ to have ℓ_2-norms less

than or equal to 1. The constraint set of $\mathbf{D}$ thus can be represented as:

$$\mathcal{D} \triangleq \{\mathbf{D} \in \mathbb{R}^{n\times m}, \text{s.t. } \forall j \in \{1,\dots,m\}, \|\mathbf{d}_j\|_2 \le 1\}. \quad (5)$$

Now, putting everything together, the template learning can be formulated as the following optimization problem,

$$\begin{aligned} \mathbf{D}^* &= \underset{\mathbf{D}}{\operatorname{argmin}} \tfrac{1}{|\mathcal{Y}|} \sum_{\mathbf{y}\in\mathcal{Y}} l(\mathbf{y}, \mathbf{D}) \\ \text{s.t. } & l(\mathbf{y}, \mathbf{D}) = \min_{\mathbf{c}} \tfrac{1}{2}\|\mathbf{y} - \mathbf{D}\mathbf{c}\|_2^2 + \lambda\|\mathbf{c}\|_1 \\ & \mathbf{D} \in \mathcal{D}. \end{aligned} \quad (6)$$

The above problem is also referred to as Dictionary Learning which has found many applications in signal processing [2, 24, 23], machine learning [8, 18] and lately computer vision for face recognition [16] and image restoration [17]. Generally, it can be optimized using a two-step iterated procedure: the first step fixes $\mathbf{D}$ and minimizes the cost function with respect to the coefficient vector $\mathbf{c}$; and the second step fixes the coefficient vector $\mathbf{c}$ and performs gradient descent like methods to minimize the cost function.

In our scenario for template updating in online object tracking, since the target candidates are obtained consecutively, the above two-step iterated procedure must be redesigned to be performed in an online manner to learn the dictionary incrementally. In Algorithm 1, we summarize this redesigned procedure of online dictionary learning for template update. The learning procedure receives the dictionary learned in the previous frame as input, and updates the dictionary incrementally according to the samples collected in the current frame. In the algorithm, Step 1 is solved using the LARS method [9] and Step 2 admits an analytical solution. The involved matrix inversion is calculated in an online manner using the Sherman-Morrison formula [21] to make the learning process more efficient. The introduced variables $\mathbf{C}_t$ and $\mathbf{Y}_t$ are intermediate results associated with the dictionary $\mathbf{D}_t$ and are stored for incremental learning. The initial value for $\mathbf{D}_0$, $\mathbf{C}_0$ and $\mathbf{Y}_0$ are obtained from the tracker. To improve the robustness of the learned dictionary, multiple samples are collected around the tracking result $\mathbf{x}_t$ in frame $\mathbf{I}_t$, and M is the parameter to control the explicit number of the collected samples. Note that here we do not impose any constraints on the explicit dictionary format, which can be object templates, image patches or even extracted features.

3.2. Multi-lifespan Dictionary Building

The adaptability and robustness are the two key characteristics that a tracker should possess. Adaptability means that the tracker should accommodate to target appearance changes quickly, while robustness refers to the ability to keep on working under different situations. These two characteristics, however, often contradict with each other in many tracking algorithms. As for template based object tracking, if the template is updated with a faster speed, the tracker can better adapt to the changes of the target but may be more likely to drift due to noises accumulated along with fast updating. On the contrary, if the template is updated with a slower speed, the tracker is not easy to drift but may not catch up with the changes of the target. Based on the formulation of template update using online dictionary learning, we explore the temporal properties of the learned dictionary and propose to build a multi-lifespan dictionary learning model to further improve the tracking effectiveness and guarantee adaptability and robustness simultaneously.

Algorithm 1 Online dictionary learning for template update

Input: frame data $\mathbf{I}_t$, tracking results $\mathbf{x}_t$, learned dictionary $\mathbf{D}_{t-1}$, $\mathbf{C}_{t-1}$, $\mathbf{Y}_{t-1}$ in the previous frame, λ (regularization parameter), M (sample drawing number).

Output: learned dictionary $\mathbf{D}_t$ in the current frame.

1: **Initialization**: $\mathbf{D}_t \leftarrow \mathbf{D}_{t-1}$,$\mathbf{C}_t \leftarrow \mathbf{C}_{t-1}$,$\mathbf{Y}_t \leftarrow \mathbf{Y}_{t-1}$.

2: **for** $i = 1 \rightarrow M$ **do**

3: **Step 1**: fix $\mathbf{D}_t$ to find the best coefficients,

$$\mathbf{c}_t^{(i)} = \underset{\mathbf{c}\in\mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{y}_t^{(i)} - \mathbf{D}_t\mathbf{c}\|_2^2 + \lambda\|\mathbf{c}\|_1.$$

4: **Step 2**: fix $\{\mathbf{c}_t^{(i)}\}$ to update the dictionary,

$$\mathbf{C}_t \leftarrow \mathbf{C}_t - \frac{\mathbf{C}_t\mathbf{c}_t^{(i)}\mathbf{c}_t^{(i)\top}\mathbf{C}_t}{1+\mathbf{c}_t^{(i)\top}\mathbf{C}_t\mathbf{c}_t^{(i)}}, \mathbf{Y}_t \leftarrow \mathbf{Y}_t + \mathbf{y}_t^{(i)}\mathbf{c}_t^{(i)\top},$$

$$\begin{aligned} \mathbf{D}_t &= \underset{\mathbf{D}\in\mathcal{D}}{\operatorname{argmin}} \sum_{j=1}^{i} \frac{1}{2}\|\mathbf{y}_t^{(j)} - \mathbf{D}\mathbf{c}_t^{(j)}\|_2^2 + \lambda\|\mathbf{c}_t^{(j)}\|_1, \\ &= \left(\textstyle\sum_{j=1}^{i} \mathbf{c}_t^{(j)}\mathbf{c}_t^{(j)\top}\right)^{-1} \left(\textstyle\sum_{j=1}^{i} \mathbf{y}_t^{(j)}\mathbf{c}_t^{(j)\top}\right), \\ &= \mathbf{C}_t\mathbf{Y}_t. \end{aligned}$$

5: **end for**

6: Save dictionary $\mathbf{D}_t$, intermediate variable $\mathbf{C}_t$ and $\mathbf{Y}_t$.

In order to explore the properties of the online learned dictionary, we represent it using a 5-tuple based on its learning procedure, *i.e.*,

$$\mathsf{D} = \left\langle \mathbf{D}_t, \mathbf{C}_t, \mathbf{Y}_t, \{\mathbf{y}_{t_s:t_e}^{(i)}\}_{i=1}^{M}, \lambda \right\rangle, \quad (7)$$

where $\{\mathbf{y}_{t_s:t_e}^{(i)}\}_{i=1}^{M}$ denotes all the candidates sampled to train the dictionary and completely determinate the learned dictionary together with regularization parameter λ. Here the subscript t_s and t_e, *i.e.* the start and end frame number of candidates, reflect the temporal property of the training data. By collecting training candidates from different temporal intervals with a corresponding sampling strategy[1], we can learn dictionaries of multiple temporal properties. Multi-lifespan dictionaries provide a very good solution to the contradiction when simultaneously pursuing the adaptability and robustness of the tracker.

As show in Figure 2, we simultaneously learn three different lifespan dictionaries, the Short Lifespan Dictionary (SLD), the Middle Lifespan Dictionary (MLD), and the

[1]Here sampling strategy refers to different sampling variance and candidate number M in each frame.

Figure 3. **Online dictionary learning for template update**. Line 1: tracking results; Line 2-4: examples of the learned dictionaries at frame 100; Line 5-7: collected negative samples used for the discriminative observation model (see Section 3.3).

Long Lifespan Dictionary (LLD). The SLD is trained using the candidates densely sampled only in the previous frame (*i.e.*, $t_s = t-1$) and made the best adaptation to the target in current frame. The LLD, on the contrary, is trained using accurately sampled candidates in all previous frames to establish a robust object appearance model (*i.e.*, t_s=1). Between the SLD and LLD, the MLD tries to build an intermediate model that compromises the modes built by SLD and LLD (*i.e.*, t_s=$t/2$). Denoting SLD, MLD and LLD respectively as D^S, D^M and D^L, the final online multi-lifespan dictionary learning model (OMDL) is represented as:

$$\mathsf{D}^* = \left\{\mathsf{D}^S, \mathsf{D}^M, \mathsf{D}^L\right\}. \tag{8}$$

In Figure 3, we give some examples of the three lifespan dictionaries learned using the online dictionary learning algorithm. It can be observed that the SLD successfully captures a more adaptive object appearance model, while the LLD builds a more robust object appearance model. The MLD obtains a good intermediate object model that balances the models built by LLD and SLD.

3.3. Bayesian Sequential Estimation

We deploy the OMDL model into the Bayesian sequential estimation framework, which performs tracking by solving the maximum a posterior (MAP) problem,

$$\hat{\mathbf{x}}_t = \underset{\mathbf{x}_t}{\operatorname{argmax}}\, p(\mathbf{x}_t|\mathbf{y}_{1:t}), \tag{9}$$

where $\mathbf{y}_{1:t} = \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$ represents all the observation candidates until the current frame. The posterior probability $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is calculated recursively by the Bayesian theorem using particle filter [12],

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{x}_t|\mathbf{y}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}, \tag{10}$$

where $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is the dynamic model and $p(\mathbf{x}_t|\mathbf{y}_t)$ is the observation model. We employ the affine transformation to model the object motion between consecutive frames. The observation model, which is of fundamental importance to the success of the tracker, is modeled using the OMDL in both generative and discriminative manners.

The generative observation model using OMDL first solves the problem

$$\min_{\mathbf{c}} \|\mathbf{D}_t^P\mathbf{c} - \mathbf{y}_t\|_2^2 + \lambda\|\mathbf{c}\|_1, \tag{11}$$

where $\mathbf{D}_t^P = \left[\mathbf{D}_t^S, \mathbf{D}_t^M, \mathbf{D}_t^L\right]$. Based on the solution of this problem, the general observation model is built as:

$$g(\mathbf{y}_t|\mathbf{x}_t) = \sum_{I\in\{S,M,L\}} \exp\left(-\alpha\|\mathbf{D}_t^I\mathbf{c}^I - \mathbf{y}_t\|_2^2\right), \tag{12}$$

where superscript I denotes the corresponding decomposition of $\mathbf{D}_t$ and $\mathbf{c}$, and α is a constant as in Eqn. (2). Here we fuse the reconstruction confidences of the OMDL model equally to build the general observation model. Although more sophisticated strategies, *e.g.* weighted fusing based on the reconstruction error, can be adopted, we find this simple strategy works well in the experiments.

The discriminative observation model using OMDL first collects some background samples (denoted as $\mathbf{D}_t^N$, see Figure 3) around the target in the previous frame and then selects the discriminative features by solving the problem

$$\min_{\mathbf{s}} \|\mathbf{D}_t^{*T}\mathbf{s} - \mathbf{p}\|_2^2 + \lambda\|\mathbf{s}\|_1, \tag{13}$$

where $\mathbf{D}_t^* = [\mathbf{D}_t^P, \mathbf{D}_t^N]$ and $\mathbf{p}$ is the label vector for $\mathbf{D}_t^*$ (+1 for object samples and -1 for background samples). The solution of this problem is a vector $\mathbf{s}$ with indexes of non-zero elements indicating selected features, which can be used to form the projection matrix $\mathbf{P}$ by removing all-zero rows of matrix $\mathbf{S} = \mathrm{Diag}(\mathbf{s} \neq 0)$. After solving

$$\min_{\mathbf{c}} \|\mathbf{D}_t'\mathbf{c} - \mathbf{y}_t'\|_2^2 + \lambda\|\mathbf{c}\|_1, \text{ with } \mathbf{D}_t' = \mathbf{P}\mathbf{D}_t^*, \mathbf{y}_t' = \mathbf{P}\mathbf{y}_t, \tag{14}$$

the discriminative observation model is built as:

$$d(\mathbf{y}_t|\mathbf{x}_t) = \exp\left(-\beta\left(\|\mathbf{D}_t'^P\mathbf{c}^P - \mathbf{y}_t'\|_2^2 - \|\mathbf{D}_t'^N\mathbf{c}^N - \mathbf{y}_t'\|_2^2\right)\right), \tag{15}$$

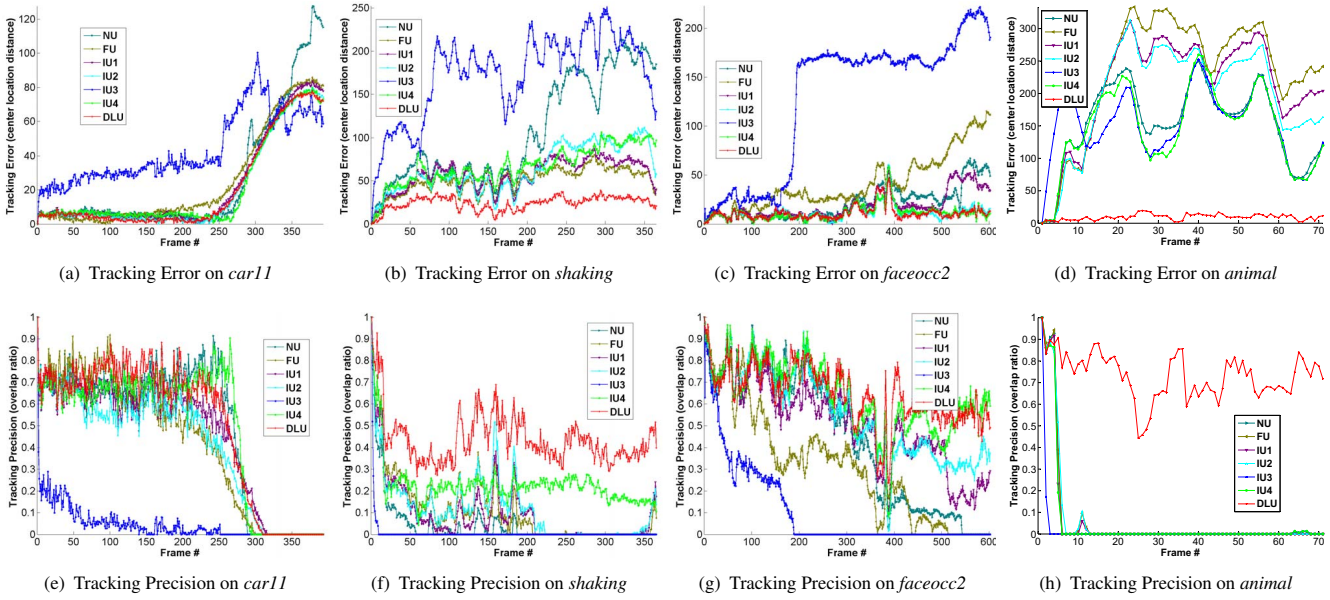
where the superscript $\mathbf{P}$ and $\mathbf{N}$ are used to decompose $\mathbf{D}_t'$ and $\mathbf{c}$. The final observation model is then represented as,

$$p(\mathbf{y}_t|\mathbf{x}_t) \propto g(\mathbf{y}_t|\mathbf{x}_t)d(\mathbf{y}_t|\mathbf{x}_t). \tag{16}$$

4. Experiments

We implement the proposed approach in MATLAB and run the experiments on an Intel Core 3.4 GHz PC with 4 GB memory. The regularization parameter λ in the sparse coding problems is fixed as 0.01. We use global object template normalized to 32×32 pixels as the training data to learn the multi-lifespan dictionary model. The dictionary numbers for the SLD, MLD and LLD are all set to 20 and incrementally learned with 128, 8 and 1 sample(s) respectively at every frame. The number of the negative samples collected at each frame is set to 60. The constant α and β to control the Gaussian kernel shape are set to 2.0 and 6.0 respectively. Note that these parameters are fixed in all the experiments.

We first conduct experiments to compare the tracking results using six different template update methods. Then we evaluate the tracking performance of our approach compared with six state-of-the-art tracking algorithms. At last, we give some analyses and discussions of our method.



(a) Tracking Error on *car11* (b) Tracking Error on *shaking* (c) Tracking Error on *faceocc2* (d) Tracking Error on *animal*

(e) Tracking Precision on *car11* (f) Tracking Precision on *shaking* (g) Tracking Precision on *faceocc2* (h) Tracking Precision on *animal*

Figure 4. **Tracking error and precision of seven different methods for template update**. The experiment is conducted on four challenging video sequences, including *car11*, *shaking*, *faceocc2* and *animal*. Best viewed in original color PDF file.

4.1. Template Update Method Evaluation

Template update is very important for object tracking, especially in complex scenes where the target undergoes great changes. We compare our dictionary learning method for template update (DLU) with six typical ones, including no updates (NU, using fixed templates), fully update (FU, updating the whole templates using the tracking results in the current frame), the intuitive update method in [19] (IU1) and [26] (IU2), the incremental subspace learning method in [22] (IU3) and [13] (IU4). In order to concentrate on the template update method and make a fair comparison, the templates in these seven methods are all built from global target appearances and the number of templates is set to 60. The effectiveness of the obtained templates is judged by their descriptive power of the target, which is evaluated using the same measure generated from Eqn. (2). The experiments are performed on four challenging image sequences, *car11*, *shaking*, *faceocc2* and *animal* (see Figure 1) with the same initial rectangles in the first frame, which cover most challenging situations for template updating.

We employ two well-accepted metrics, center location distance and overlap ratio, to respectively evaluate the tracking error and precision of the seven template update methods. The center location distance is normalized by the object size. In Figure 4, we plot the full quantitative experimental results of the seven methods on the four test sequences. Our dictionary learning method for template update obtains the minimal tracking error and the highest tracking precision on aggregate, especially on the test sequence *animal*, where all the other template update methods fail to follow the deer running at high speed from the fifth frame but only leaving our method to track the target until the end of the sequence. It is really surprising that the incremental update method in [22], which uses the eigenvector of the target samples as template and updates it incrementally in the eigenspace, performs poorly on the four test sequences and even is no better than the fixed template method and fully update method. The reason behind this may be that forcing the template to be orthotropic cannot well adapt to the challenging tracking situations with non-white image noises, especially when using these templates to perform sparse representation [23]. This may be the reason why Xu *et al*. [13] proposes a modified template update method to better deploy the subspace learning into sparse representation. From Figure 4 it is observed that fixed templates and fully update method may perform well when the target does not change much. But with the accumulation of tracking errors and when the target undergoes great changes, these two methods tend to perform worse than other four methods using incremental template update.

Note that in the *car11* sequence, all the seven methods fail to track the car at about the 300th frame when the car suddenly turns right. This is because a generative observation model may not be enough to perform robust tracking, which is also the reason why we deduce two different kinds of observation models for the final algorithm in Section 3.3.

4.2. Tracking Performance Evaluation

We further evaluate the performance of our final tracking approach on 10 video sequences popularly used in previous works [14, 4, 5, 28, 13, 10], including *sylv*, *bike*, *david*, *woman*, *coke11*, *jumping*, and the four sequences used in the first experiment. These ten video sequences together present an even wider range of challenges to a tracking algorithm (see Figure 5). The tracking results are compared

Table 2. Average tracking errors (in pixels). The best and second best results are respectively shown in **red** and **blue** colors.

Sequence	Frag	IVT	MIL	VTD	ℓ_1	MTT	Ours
sylv	0.245	0.875	**0.156**	0.220	0.961	0.260	**0.139**
bike	2.109	0.075	0.083	0.086	0.082	**0.070**	**0.054**
car11	1.436	**0.062**	0.848	**0.065**	0.378	0.403	0.161
david	0.946	**0.057**	0.194	0.351	0.210	1.103	**0.110**
woman	1.302	1.590	1.351	**1.126**	1.305	2.281	**0.135**
animal	0.934	0.101	0.182	**0.056**	0.059	**0.047**	**0.047**
coke11	1.247	0.894	0.381	0.759	0.954	**0.338**	**0.178**
shaking	0.704	1.005	**0.222**	0.279	1.286	0.336	**0.161**
jumping	0.169	**0.094**	0.245	1.121	1.417	0.666	**0.081**
faceocc2	0.137	**0.101**	0.252	0.117	0.149	**0.097**	0.113
Average	0.923	0.485	**0.391**	0.418	0.680	0.560	**0.118**

Table 3. Average tracking precision. The best and second best results are respectively shown in **red** and **blue** colors.

Sequence	Frag	IVT	MIL	VTD	ℓ_1	MTT	Ours
sylv	0.617	0.450	0.751	**0.810**	0.323	0.770	**0.833**
bike	0.136	**0.983**	0.917	**1.000**	0.908	**1.000**	**1.000**
car11	0.097	**1.000**	0.102	**0.972**	0.682	0.687	0.781
david	0.089	**0.905**	0.537	0.615	0.435	0.320	**0.779**
woman	0.256	0.204	0.209	**0.309**	0.215	0.198	**0.440**
animal	0.099	0.887	0.747	**0.972**	**0.972**	**1.000**	**1.000**
coke11	0.051	0.119	0.271	0.068	0.085	**0.559**	**0.678**
shaking	0.222	0.025	0.414	**0.784**	0.011	0.099	**0.578**
jumping	0.690	**0.959**	0.233	0.230	0.118	0.198	**0.984**
faceocc2	0.767	0.772	0.537	0.743	0.419	**0.929**	**0.826**
Average	0.302	0.630	0.472	**0.650**	0.417	0.576	**0.790**

with six state-of-the-art algorithms, the fragment tracker (Frag) [1], the incremental visual tracking (IVT) algorithm [22], the multi-instance learning (MIL) tracker [4], the visual tracking decomposition (VTD) method [14], the latest ℓ_1-tracker (ℓ_1) [5] and its multi-task tracking (MTT) version [27]. The implementations of these algorithms are all provided by their corresponding authors with suggested parameter settings. To make a more fair comparison, we set the rotation parameters in the motion model of IVT, ℓ_1, MTT and ours to be zero, since Frag, MIL and VTD do not rotate the object samples and the ground-truths of the test sequences also do not consider the rotation of the target. All the seven algorithms are initialized with the same initial bounding box according to the ground-truth, with other parameters set as suggested by their corresponding authors. We analyze the experimental results both quantitatively and qualitatively.

Table 2 and 3 list the average tracking errors and precisions for all seven algorithms. The proposed tracking approach, on the whole, performs well against other six algorithms, especially on the sequence *sylv*, *woman*, *animal*, *coke11*, and *jumping*, on which some other algorithms may fail to follow the targets but ours can successfully track them until the end of the sequence. The IVT, VTD and MTT also perform well on these ten sequences and can track the targets in most situations. Together with the results in the first experiment, it can be concluded that the incremental subspace learning method in IVT is more suitable to model the tracking confidence directly using the reconstruction error over all the orthorhombic templates, rather than a few templates, which may lose too much information. The templates learned using our dictionary learning method, on the contrary, can well adapt to the tracking data, especially in the sparse coding based tracking framework. In Figure 5, some example tracking results are drawn to given a more vivid comparison. Due to the page limitation, we provide more experimental results in the supplementary material.

4.3. Speed Analysis and Discussions

Our tracking algorithm runs at about 2.5 fps in the current MATLAB implementation without using optimization technologies like parallel computing or GPU acceleration. Table 4 lists the speed of several popular ℓ_1-trackers by running the codes provided by the corresponding authors on our test platform. For the first two ℓ_1-trackers in Table 4, whose implementations are not publicly available, we calculate their speed based on the report from [5]. It can be observed that our algorithm is faster than most other trackers. The main reason is that our approach does not need to add the trivial templates as those adopted in [19, 20, 5, 27] due to the design of observation model. Therefore, although we use multi-lifespan dictionaries, the total number of templates is greatly reduced, *e.g.*, from 1084 (60+32×32) to 60. What is more, currently our learning for template update is performed frame-by-frame for easy implementation, which may not be necessary for tracking. It can be performed only every several frames, *e.g.*, every five frames like in [28] and [13]. In that implementation, the speed of the learning procedure can be further improved.

Table 4. Running speed comparison of several popular ℓ_1-trackers.

Algorithm	[19]	[20]	[5]	[27]	[28]	[13]	Ours
Speed	0.01fps	0.05fps	1fps	2fps	2.5fps	2.5fps	2.5fps

5. Conclusions and Future Work

We study the template update problem in the sparse coding based object tracking framework. We formulate the template update problem as online dictionary learning, which make the template better adapt to the tracking data. We propose to learn a multi-lifespan dictionary to simultaneously ensure adaptability and robustness of the tracker. The online learned multi-lifespan dictionary has been deployed into the Bayesian sequential estimation framework using particle filter to perform tracking. Extensive experiments on challenging image sequences demonstrate the effectiveness of the proposed method. Currently, the lifespan dictionary is only learned from global object templates, and in our future work, we plan to add local image patch based dictionary to further improve the tracking performance.

Acknowledgement: This work is partly supported by NSFC (Grant No. 60935002), the National 863 High-Tech R&D Program of China (Grant No. 2012AA012504), the Natural Science Foundation of Beijing (Grant No. 4121003), the Guangdong Natural Science Foundation (Grant No. S2012020011081), and Singapore Ministry of Education (Grant No. MOE2010-T2-1-087).

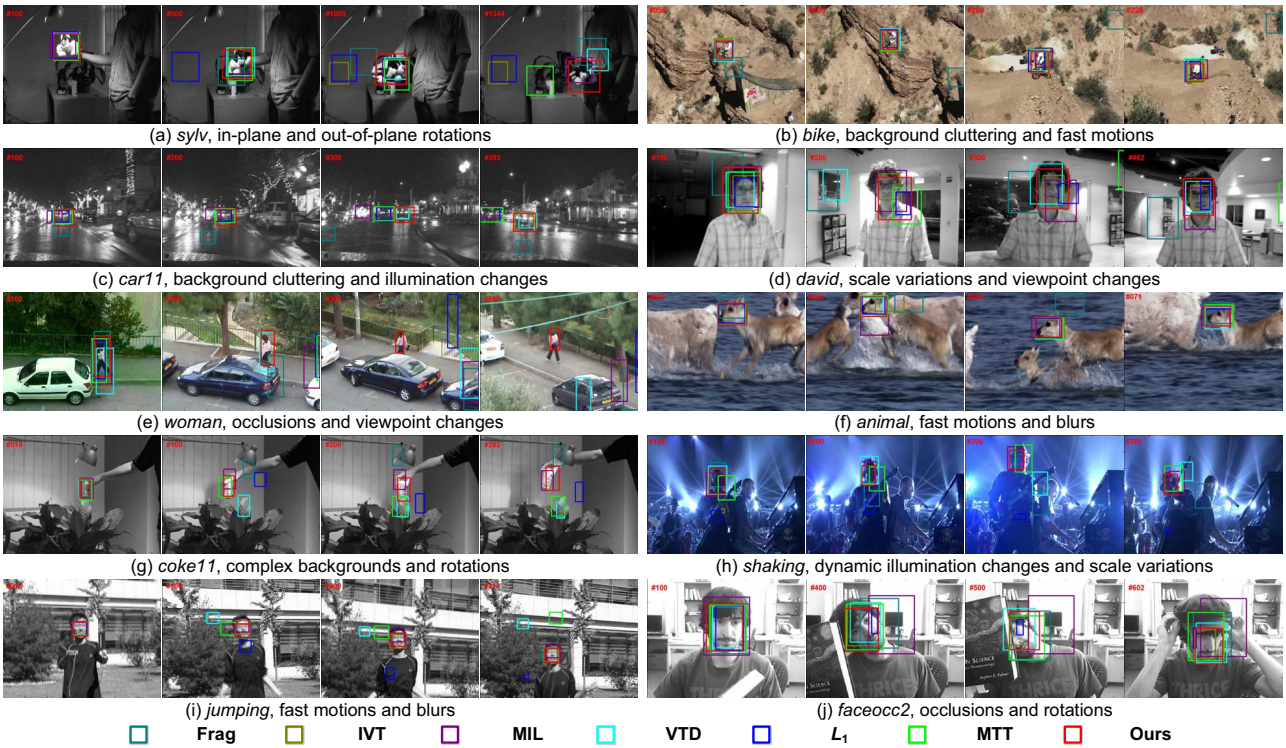


Figure 5. **Example tracking results of the seven different algorithms**. Our tracking algorithm performs well against the six state-of-the-art tracking algorithms, and works robustly and adaptively under a lot of difficult situations, like complex backgrounds ((c) and (g)), illumination changes ((c) and (h)), rotations ((a) and (g)), fast motions ((b) and (c)), scale variations ((d) and (h)), viewpoint changes ((d) and (e)), and occlusions ((e) and (j)). Best viewed in original color PDF file.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006.

[2] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *TSP*, 54(11):4311–22, 2006.

[3] S. Avidan. Ensemble tracking. *TPAMI*, 29(2):261–71, 2007.

[4] B. Babenko, M. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *TPAMI*, 33(8):1619–32, 2011.

[5] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust ℓ_1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012.

[6] M. Black and A. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *IJCV*, 25(1):63–84, 1998.

[7] D. Comaniciu and P. Meer. Kernel-based object tracking. *TPAMI*, 25(5):564–77, 2003.

[8] K. Delgado, J. Murray, B. Rao, K. Engan, T. Lee, and T. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Comput.*, 15(2):349–96, 2003.

[9] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Ann. Stat.*, 32(2):407–51, 2004.

[10] M. Godec, P. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. *CVIU*, 117(10):1245–1256, 2013.

[11] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via online boosting. In *BMVC*, 2006.

[12] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.

[13] X. Jia, H. Lu, and M. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012.

[14] J. Kwon and K. Lee. Visual tracking decomposition. In *CVPR*, 2010.

[15] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans. *TPAMI*, 30(10):1728–40, 2008.

[16] L. Ma, C. Wang, B. Xiao, and W. Zhou. Sparse representation for face recognition based on discriminative low-rank dictionary learning. In *CVPR*, 2012.

[17] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *TPAMI*, 34(4):791–804, 2012.

[18] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *JMLR*, 11:19–60, 2010.

[19] X. Mei and H. Ling. Robust visual tracking using ℓ_1 minimization. In *ICCV*, 2009.

[20] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai. Minimum error bounded efficient ℓ_1 tracker with occlusion detection. In *CVPR*, 2011.

[21] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes: The Art of Scientific Computing (3rd Edition)*. Cambridge University Press, New York, 2007.

[22] D. Ross, J. Lim, R. Lin, and M. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–41, 2008.

[23] I. Tosic and P. Frossard. Dictionary learning. *IEEE Signal Process. Mag.*, 28(2):27–38, 2011.

[24] M. Yaghoobi, T. Blumensath, and M. Davies. Dictionary learning for sparse approximations with the majorization method. *TSP*, 57(6):2178–91, 2009.

[25] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):1–45, 2006. Article 13.

[26] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Low-rank sparse learning for robust visual tracking. In *ECCV*, 2012.

[27] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via multi-task sparse learning. In *CVPR*, 2012.

[28] W. Zhong, H. Lu, and M. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012.