Signal Processing: Image Communication 60 (2018) 183–192

Contents lists available at ScienceDirect

Signal Processing: Image Communication

journal homepage: www.elsevier.com/locate/image

Deep learning assisted robust visual tracking with adaptive particle filtering

Xiaoyan Qian *, Lei Han, Yuedong Wang, Meng Ding

College of Civil Aviation, Nanjing University of Aeronautics and Astronautics, 210016, Nanjing, China

ARTICLE INFO

Keywords:
Visual tracking
Deep learning
Particle filter

ABSTRACT

We propose a novel visual tracking algorithm based on the representations from a pre-trained Convolutional Neural Network (CNN). Our algorithm pre-trains a simplified CNN using a large set of videos with tracking ground truths to obtain a generic target representation. When tracking, Particle Filtering (PF) is combined to the fully-connected layer in the pre-trained CNN. Deep representations and hand-crafted features help to model tracking. To optimize the particles' distribution, the velocity and acceleration information aids to calculate dynamic model. Meanwhile, our algorithm updates the tracking model in a lazy manner to avoid shift and expensive computation. As compared to previous methods, our results demonstrate superior performances in existing tracking benchmarks.

1. Introduction

Visual tracking, i.e., tracking a specific target object in consecutive video frames to get its moving trajectory, is a fundamental problem in computer vision and has been actively studied for decades. A wide range of applications rely on robust visual tracking including security and surveillance, vehicle transportation, traffic monitoring and video compression [1].

Current visual tracking algorithms include generative and discriminative approaches. Generative methods establish appearance prior distribution and search for the target regions that fit the models best. Various generative appearance modeling includes template-models [2–4], incremental subspace learning [5], sparse representation [6,7] and back-propagation [8,9]. They can produce strong appearance features for the targets but lack of background information. So noises are easy to be brought into models once the occlusion appears, that will result in error and shift. In contrast, discriminative methods aim to build a model that distinguishes the target object from the background. These tracking algorithms typically learn classifiers based on multiple instance learning [10], online boosting [11] and structured output SVMs [12], etc. The two methods are usually limited to using too simple hand-crafted features for target representation, such as Haar-like features, color histogram features [13], LBP and so on. When facing different problems and tasks, we need to re-design these features, which is very time-consuming. In addition, each type of hand-crafted feature is commonly able to address a few specific classical challenges. However, none is able universally to handle the variety of issues that may occur in a given video especially when there is large occlusion, fast motion or complex background in the context of tracking.

In this work, we aim to investigate deep features for tracking tasks. Deep learning adopts hierarchical architecture to simulate human brain mechanism, which can generate outstanding representation for high non-structured visual data. It has been successfully applied to various computer vision tasks such as image classification and recognition, semantic segmentation. Visual tracking, however, has been less affected by these popular trends since it is difficult to collect a large amount of training data for video processing application. To generate enough target training candidates, the tracker draws samples in translation and scale dimension [14]. In addition, it is very difficult to get real-time online tracking for large-scale deep networks. To take advantage of deep learning network, we first give a simplified shallow network and train it offline based on large-scale samples from the existed dataset. Then the deep network helps extract features during online tracking. In addition, our network architecture is substantially less deep than the ones commonly used in typical recognition tasks [15,16]. It is more appropriate for visual tracking. Indeed, the problem visual tracking consists in distinguishing between only two classes: objects and background, which requires much less complex model than general visual recognition problems. Meanwhile the smaller network is obviously more efficient during online tracking.

Here, we exploit the advantages from deep learning and particle filtering (PF) and propose a novel tracking method. The contributions of this paper are summarized below:

* Corresponding author.
E-mail address: qianxiaoyan@nuaa.edu.cn (X. Qian).

http://dx.doi.org/10.1016/j.image.2017.09.001
Received 2 March 2017; Received in revised form 5 September 2017; Accepted 5 September 2017
Available online 21 September 2017
0923-5965/

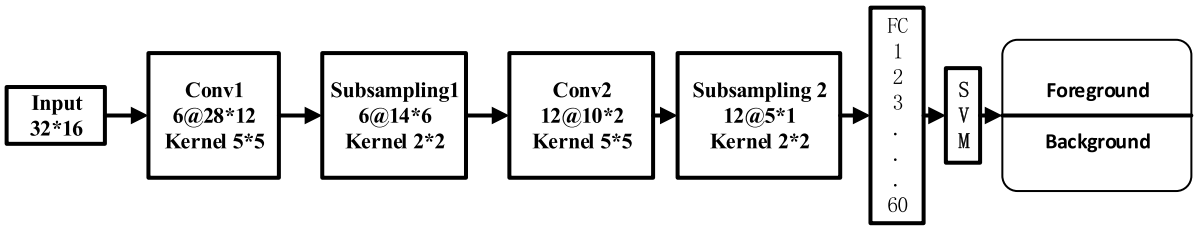


Fig. 1. Our learning framework.

(1) Although recent tracking methods based on deep learning typically attempt to learn a network in an online manner, our algorithm employs a pre-trained network to help PF tracking to model target objects. This achieves outstanding performance especially when there is large occlusion or fast motion.

(2) Particle filtering directs to choose candidates and locate targets through the Bayesian optimal estimate, which avoids human intervention and brings automatic tracking. And meanwhile, the fusion of deep features and hand-crafted features brings more stable observation model, which makes PF choose small number of particles while bring robust tracking and release the computational burden.

(3) We employ the motion information to reshape the particle distribution in the propagation step and to weight the observation model in measurement step. This approach further improves the efficiency and accuracy of traditional PF tracking.

The rest of the paper is organized as follows. We first review related work in Section 2. Section 3 describes our learning and tracking algorithm, and Section 4 demonstrates the experimental results in tracking benchmark datasets.

2. Related work

PF is extended from the Kalman filter (KF) for non-linear and non-Gaussian models. In comparison to the KF, it is more robust as it can approach the Bayesian optimal estimate with a sufficiently large number of particles. So PF gets a wide application in visual tracking. After Isard and Blake [17] have first applied PF for object tracking, more and more research begins to focus on PF tracking, such as CPF [18], SPT [19], IVT [20], L1 [21] and ASLS [22]. To deal with complicated and inconstant deformation and environments, these algorithms need to design complex appearance and motion models. Wang et al. [23] proposes an auto-reconstructing particle filter. By splitting and merging trackers, this algorithm can reduce the probability of losing targets. Volkan et al. [24] proposes an audio assisted visual tracking. They employ the direction of arrival angles of the audio sources to reshape the typical Gaussian noise distribution in the particles' propagation step and to weight the observation model in the measurement step. Zhou et al. use an adaptive 1st/2nd order propagation dynamic model to solve the problem of a strong motion [25].

On the other hand, a few tracking algorithms have begun to use the representations from deep learning in recent years. Since it is difficult to collect a large amount of training data online, there are tracking algorithms based on a pre-trained network. Wang et al. [26] proposes a de-noising auto-encoder to learn generic image features. Since this network is trained with tiny gray images and has no shared weight, its weak representation power sometimes results in lower accuracy compared to the methods based on hand-craft features. To improve the tracking accuracy, Hong et al. [27] put an online SVM on top of the hidden layer in the pre-trained Convolutional Neural Network (CNN). To locate targets, the algorithm needs to learn appearance models by SVM and produce saliency map online. The process is time-consuming. Li et al. [28] introduce a novel truncated structural loss function and train the deep network online with a robust sample selection mechanism. The network used in this work is shallow since learning a deep network using a limited number of training examples is challenging.

3. Deep visual tracking

This section describes the comprehensive procedure of our tracking algorithm. We first discuss the architecture of the deep learning network. After that, online adaptive PF tracking is described. Finally, we give the detailed description for constructing tracking models using deep features, hand-crafted features and motion information.

3.1. The architecture of deep learning network

We design a substantially smaller CNN framework than the ones used in recognition tasks [15]. Our network has five layers, which consists of two convolutional layers, two pooling operations and one fully connected (FC) layer. Fig. 1 shows the structure of our network, which can be summarized as: input (32 × 16) → convolution 1 (6 × 28 × 12) → pooling 1 (6 × 14 × 6) → convolution 2 (12 × 10 × 2) → pooling 2(12 × 5 × 1) → FC (1 × 60). The input is locally normalized as a 32 × 16 image patch, which draws a balance between the representation power and computational load. The first convolution layer contains 6 kernels with size of 5 × 5 , followed by a pooling operation that reduces the 6 obtained feature maps to a lower dimension each from 28 × 12 to 14 × 6. The second convolution layer contains 12 kernels each of size 5 × 5. After the pooling operation, a 60-dimensional feature vector is mapped by the fully connected layer. The Sigmoid is adopted as the activation function for convolutional layers (shown in Fig. 2(a)) just as:

$$S_x = sigmoid\left(f_x * S_x + b_x\right) = \frac{1}{1+\mathrm{e}^{-(f_x * S_x + b_x)}} \quad (x = 1, 2) \tag{1}$$

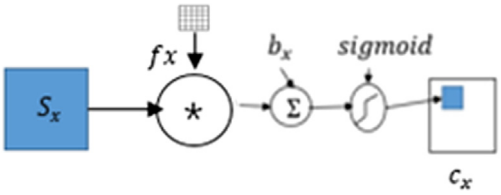
Here, f_1 and f_2 are 5 × 5 convolutional kernels helping to produce two-layer feature maps conv1 and conv2 separately. b_x is an offset value. S_1 is the normalized input image and S_2 denotes the feature maps produced by the first-layer pooling operation. Weighted average is used for pooling operations. The pooling kernels are just like Fig. 2(b). $a1, a2, a3, a4$ are four weighted values. The new value $p'(i, j)$ after subsampling is calculated by:

$$p'(i, j) = \frac{sum}{a1 + a2 + a3 + a4} \tag{2}$$

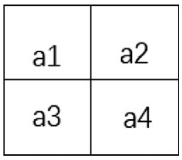
$$sum = a1p(i, j) + a2p(i, j+1) + a3p(i+1, j) + a4p(i+1, j+1) \tag{3}$$

Here, $p(i, j), p(i, j+1), p(i+1, j), p(i+1, j+1)$ denote four neighbors' values in the convolutional features maps. These simple operations not only reduce the data number but also extract the useful deep information.

Since deep learning network requires a large number of training samples, while this is often not available in visual tracking as there exists only a few number of reliable positive instances extracted from the initial frame, here we also treat this network as an offline feature extraction step on a predefined dataset for online tracking applications. The dataset is provided by visual tracker benchmark [29] and includes a large number of positive and negative samples. There are many kinds of challenging aspects by data augmentation in this dataset, such as illumination variation, rotation, scale variation, fast motion and so on. After the fully-connected layer, SVM is followed which helps distinguish the positive and negative samples. During training, the Stochastic Gradient Decent (SGD) method optimizes the parameters in our model.



(a) Convolutional structure.



(b) Pooling.

Fig. 2. The deep network structure.

3.2. Adaptive online PF tracking

Here we use PF framework to track the moving object. Traditional PF uses color histogram as the appearance model. It cannot deal with the sudden change effectively because it looks the propagation model as a simple zero-mean Gaussian process with the consistent variance for the position. To get continuous and robust tracking, we provide multiple cues that fuse deep features with hand-crafted features and add the motion information into the dynamics propagation model.

To locate the moving object correctly, our online PF framework includes four steps.

Step 1: The particles are initialized as $X_0^n \sim p\left(X_0\right), w_0^n = \frac{1}{N}$ for $n = 1, \ldots, N$. Here N is the number of particles and w_0^n are the initial weights of the particles. The state vector is defined as $X = \left[s\ x\ y\ c\right]$, where x and y are the central vertical and horizontal positions of the rectangle with the size of s around the target that we wish to track. $c = \left[c_1\ c_2\right]$ is a 1-D feature vector including two parts: $c_1 \in R^{1\times 60}$ is deep feature vector from the pre-trained learning network. $c_2 = \left[h_1, h_2, \ldots, h_L\right] \in R^{1\times L}$ is the color histogram and L denotes the number of histogram bins. Here, we also calculate the color histogram in HSV color space since it is observed to be more robust to illumination variation. Different from the color histogram only produced from H channel, we consider all three channels and the space distance. The values of three channels are fused by:

$$L\left(i,j\right) = H\left(i,j\right)Q_s + S\left(i,j\right)Q_v + V\left(i,j\right) + 1 \tag{4}$$

Here (i, j) is the location of each pixel in one particle. Q_s and Q_v are bin levels of S and V channels: $Q_s = Q_v = 4$. H, S, V denote their values, which are quantified as:

$$H\left(i,j\right) = \begin{cases} 0, H\left(i,j\right) \in [0,10) \\ 1, H\left(i,j\right) \in [10,20) \\ \cdot \\ \cdot \\ \cdot \\ 45, H\left(i,j\right) \in [350,360) \end{cases} \tag{5}$$

$$S\left(i,j\right) = \begin{cases} 0, S\left(i,j\right) \in [0,0.173) \cup [0.923,1) \\ 1, S\left(i,j\right) \in [0.173,0.423) \\ 3, S\left(i,j\right) \in [0.423,09.673) \\ 4, S\left(i,j\right) \in [0.673,0.423) \end{cases} \tag{6}$$

$$V\left(i,j\right) = \begin{cases} 0, V\left(i,j\right) \in [0,0.25) \\ 1, S\left(i,j\right) \in [0.25,0.5) \\ 3, S\left(i,j\right) \in [0.5,0.75) \\ 4, S\left(i,j\right) \in [0.75,1) \end{cases} \tag{7}$$

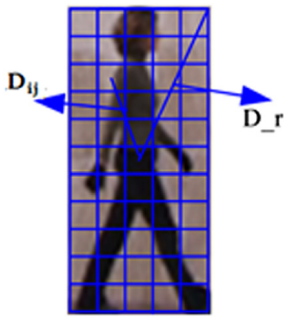


Fig. 3. The distance diagram.

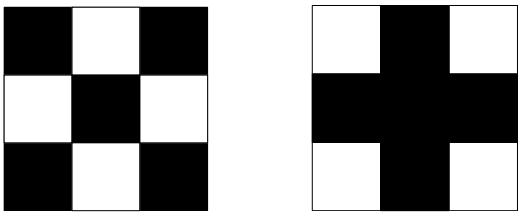


Fig. 4. Different features with the same color number.

So $L = 45\times 4 + 3\times 4 + 3 + 1 = 196$. Each element h_t in c_2 is calculated as:

$$h_t = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(\delta\left(L\left(i,j\right) - h_t\right) + k_{ij}\right)}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}k_{ij}} \tag{8}$$

Where $M \times N$ is the number of the pixels in one particle. δ is Kronecker delta function. k_{ij} is the distance ratio shown in formula (9) and Fig. 3:

$$k_{ij} = 1 - \left(D_{ij}/D_r\right)^2 \tag{9}$$

In this way, the pixels are more efficient for the target recognition if they are much nearer to the central position. On the other hand, the distance helps to distinguish different targets with the same color histograms just like Fig. 4.

Step 2: In this step, particle propagation is employed by a dynamic model just similar to the 1st/2nd propagation model in [25]:

$$S_t = S_{t-1} + f_{t-1} \tag{10}$$

Where f_{t-1} is a multivariate Gaussian random variable, which is given by:

$$f_{t-1} = \frac{1}{\sqrt{2\pi}\sigma_{t-1}}\exp\left(-\frac{\left(\text{cen} - \mu_{t-1}\right)^2}{2\sigma_{t-1}^2}\right) \tag{11}$$

Here, cen is the central position of the particle in the previous frame. μ_{t-1}, σ_{t-1} are separately mean and variance, which are defined by the prior motion information: μ_{t-1} is the mean velocity of the three continuous frames and σ_{t-1} denotes the accelerated speed.

$$\mu_{t-1} = \left(V_{t-1} + V_{t-2} + V_{t-3}\right)/3 \tag{12}$$

$$\sigma_{t-1} = V_{t-1} - V_{t-2} \tag{13}$$

In this way, we can get more effective and reasonable particle samples in the current frame under the guidance of variable motion information. The former velocity gives the moving direction and can well direct the samples' location. And meanwhile, accelerated velocity shows the changing degree of the target's motion. The faster the target is, the bigger this value is and the wider the samples' spreading is. Comparing with the traditional PF, our method can bring robust tracking when the target or the camera moves quickly since the particle samples are selected adaptively.

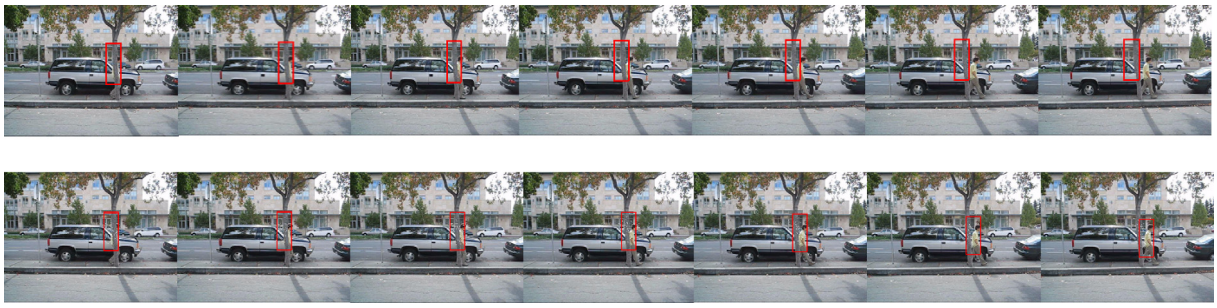


Fig. 5. The first row shows PF tracking fails after occlusion for a while. The second row shows that the algorithm keeps tracking successfully when there are motion information and deep features.

Fig. 6. The 1st and 2nd rows are the tracking results separately produced by PF and our algorithm. In this video, fast motion occurs.

Step 3: The particles are weighted by the observation model. Since the positions of the particles are changed, the importance weights are not only revised by the difference of appearance features between current particles and the observed model, but also by the Euclidean distance d_{t-1}^{i} of the particles from that with the biggest weight in the previous step. The weight of each current particle is defined by:

$$w_t^i = \left(e^{-\lambda D(i)}\right) \frac{\sum_{i=1}^{N} d_{t-1}^{i}}{d_{t-1}^{i}} i = 1, \ldots, N \tag{14}$$

The weights are normalized to ensure $\sum_{i=1}^{N} w_t^i = 1$. Where λ is the design parameter and $D(i)$ is the Bhattacharyya distance:

$$\mathrm{D}(i) = \sqrt{1 - \sum_{u=1}^{60+L} \sqrt{m(u)\, c_i(u)}} \tag{15}$$

Where $m(u)$ is the feature vector of the current dynamic model which is initialized by the user and updated in time. $c_i(u)$ is the feature vector of the ith particle. d_{t-1}^{i} is the position Euclidean distance:

$$d_{t-1}^{i} = \sqrt{\left(x_{t-1}^{i} - x_{t-1}^{max}\right)^2 + \left(y_{t-1}^{i} - y_{t-1}^{max}\right)^2} \tag{16}$$

Here, $(x_{t-1}^{i}, y_{t-1}^{i})$ is the position of the ith particle, $(x_{t-1}^{max}, y_{t-1}^{max})$ is the position of the particle with maximum $D(i)$.

In this way, our algorithm makes sure that the particles have higher importance weights in terms of Bhattacharyya distance and Euclidean distance if they are more alike the model and nearer to the particle which is most like the model.

Step 4. The tracked object is estimated by

$$\left(x_t, y_t\right) = \sum_{n=1}^{N} w_t^n \left(x_t^n, y_t^n\right) \tag{17}$$

With our proposed dynamic and observation model, the algorithm can keep tracking the target even if there is the overall occlusion and the target re-appears in the scene. Fig. 5 shows an occlusion case where the target walks behind one tree and the surrounding background has similar colors to it. Another fast motion case is shown in Fig. 6 where the toy moves quickly. The visual tracker has no deep and motion cues during tracking which causes losing the target as it is depicted in the first rows of Figs. 5 and 6.

3.3. Model update

Occlusion, background disturbance, target deformation as well as the camera parameters can influence robustness of the tracker. However, in case the appearance of the object is not always changing, a well-learned tracking model can remain discriminate for a long time. Furthermore, over-updating is easy to result in shift and bring expensive computation. Motivated by these intuitions, we propose to update the tracking model in a lazy manner. Supposing the weight of the target in the current frame is w_t, and the weights in ten continuous frames before are from w_{t-10} to w_{t-1}, we compare the mean value $\overline{w} = \frac{\sum_{i=1}^{10} w_{t-i}}{10}$ with current w_t :

If $\left|w_t - \overline{w}\right| < T$ (the threshold pre-defined), the observation model will not be updated;

Else, the model gets new appearance features. Our algorithm chooses N/5 particles with the maximum weights from the current frame and calculates their mean feature $\overline{C}$. Then the new feature vector of the model is updated as:

$$\mathrm{C_{new}} = \alpha \times \overline{C} + (1 - \alpha) \times \mathrm{C_{old}} \tag{18}$$

Here α is the weighed parameter and $\mathrm{C_{old}}$ is the feature vector of the model before updating. In this way, the model can maintain the former appearance as well as the current features. The threshold and the weighted average control the updating degree. This method allows our algorithm to handle challenges including motion, deformation and occlusion.

After updating, the new weight of the current target is re-calculated according to formula (14).

4. Experimental evaluations

In this section, the proposed and baseline algorithms are evaluated on the TB-100 sequences that contain 100 fully annotated videos with substantial variations. For offline learning, we collect 9075 positive and 17 535 negative samples from the recently released tracking benchmark dataset [29]. Our deep learning architecture and SVM provide a baseline for the tracking. The 20 tested videos are shown in Fig. 7. The comparison results are presented in plots and tables.

Fig. 7. The tested videos.

Table 1
The ascension statistics of average precision for 20 sequences (sequences).

	>10%	>50%	>80%	90%
PF → Ours	20	20	20	12
CNN → Ours	18	14	10	4
2nd order → PF Ours	18	11	6	0

4.1. Parameter setting

In our algorithm, λ is set as 0.2 just used in traditional PF tracking. To get suitable number of particles, we test on several videos using different numbers of particles. From the test, we find number = 40 can bring good efficiency and partial results are shown in Fig. 8. To decide the weighted parameter α, we change it from 0 to 1 (interval is 0.1) and calculate four quantitative parameters just like Fig. 8. Finally, we set α as 0.1 through overall comparison. The kernels' parameters in subsampling $a1, a2, a3$ and $a4$ are 0.25. During training and tracking, they keep the same values.

4.2. Quantitative analysis

To highlight the contributions of deep features and hand-crafted features quantitatively, we firstly compare our proposed method with traditional PF only using color histogram features, 2nd order PF of adaptive dynamic model similar to the propagation model in [25] and the reduced version denoted by CNN, which depends only on FC features as traditional tracking-by-detection methods do. Then we also give the comparisons with the other three state-of-the-art trackers of FCT [30], DFT [31] and L1APG [32] in all the 20 benchmark sequences.

To evaluate these trackers, we run them throughout the test sequences with initialization from the ground truth position in the first frame and report one-pass evaluation (OPE) values [29]. In OPE, precision is defined as the average Euclidean distance between the center locations of the tracked targets and the labeled ground truths. Overlap rate is defined as Jaccard coefficient:

$$S = \frac{r_t \cap r_m}{r_t \cup r_m} \tag{19}$$

Here, r_t, r_m are the ground truth and the tracking region respectively. $\cap$ and $\cup$ represent the intersection and union of two bounding boxes. Then the overlap rate for one video serials is the average of all Jaccard coefficients.

Fig. 9 illustrates the OPE comparisons of traditional PF, 2nd order PF, CNN only with FC features and our method in terms of bar graphs. It can be seen that traditional PF tracker modeled only by color histograms has the worst tracking results in most sequences comparing with the other three trackers. Once velocity and acceleration information is introduced into the propagation model, PF tracking performance gets big improvement. Among these, CNN tracking only with FC features cannot deal well with heavy occlusion and quick motion in the sequences just like *Lemming*, *Biker*, *Skiing* and *MountainBike*. However, 2nd order PF can well overcome these problems especially when the quick motion occurs. FC features has competitive performance to 2nd order PF if there is deformation, partial occlusion or scale variation such as in *Faceocc1*, *Jogging* and *Dog*. To further analyze the strength and weakness of FC features and hand-crafted features, Tables 1 and 2 give the statistics for OPE values of the 20 benchmark sequences. The second and the third rows in Tables 1 and 2 denote the total sequences among 20 sequences whose OPE growth is over some percentage (listed in the first rows) from other three trackers to our method respectively. In the precision table (Table 1), our proposed method outperforms traditional PF, CNN and 2nd order PF by more than 10% in 18 sequences or even in all sequences. The precision value for CNN tracking even goes down over 50% in 14 sequences if it only depends on FC features. The precision also drops down over 50% in 11 sequences if tracking only depends on hand-crafted features in 2nd order PF. Meanwhile, our method can still outperform traditional PF, CNN and 2nd order PF by a wide margin of more than 10% in term of average overlap in most of sequences (seen in Table 2). From above, we can find that combining FC features, hand-crafted features and dynamic model brings better tracking performance than only using any of them. That is because a visual sequence always includes several challenging aspects such as fast motion, occlusion and background clutter etc. Hand-crafted features and dynamic model sometimes can deal well with fast motion and heavy occlusion. On the contrary, FC features are good for deformation, low

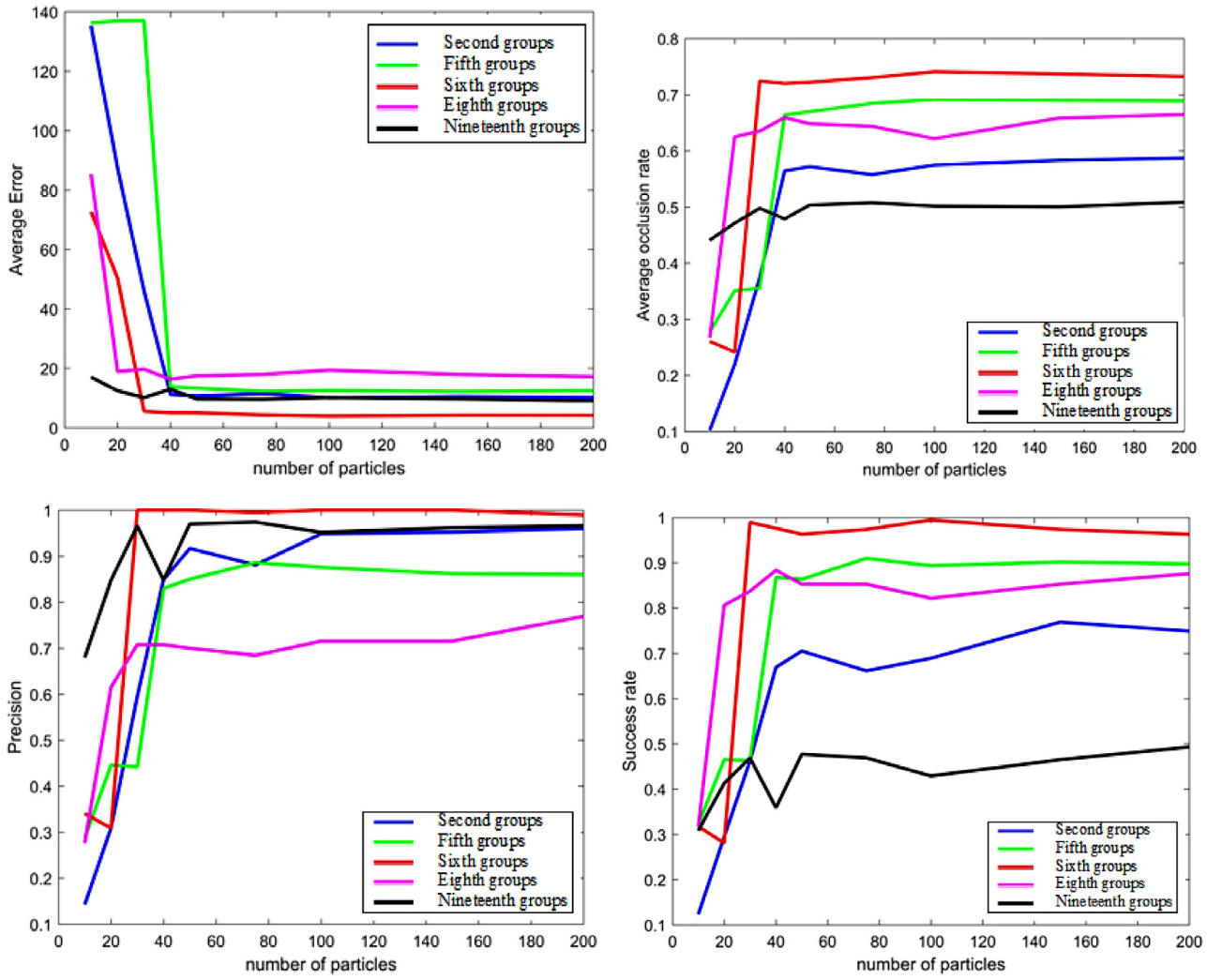


Fig. 8. The illustration of four parameters using different particle's number.

resolution, etc. Hence, when they complement each other, the tracking performance gets improvement in a big degree.

Since traditional PF tracker only with color histograms is the worst, we only compare the proposed method with CNN, 2nd order PF, and three state-of-art trackers in the following. Tables 3 and 4 list the precision and overlap rate values for 20 videos respectively. At the bottoms of each table, the average and standard deviations are also calculated. As illustrated in the tables, our method consistently outperforms the other methods in most of the challenges. Though other trackers perform better in several sequences, the standard deviation shows that our method achieves competitive results (more than 70% and more than 15% in terms of precision rate and overlap rate respectively) because of the deep FC features and hand-crafted features. Furthermore, the PF tracker with 2nd order dynamic information can also outperform the other four trackers.

To gain more insight about the proposed method, we also evaluate the performance of trackers based on temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE) since some trackers may be sensitive to the initialization and start frame. For TRE, each sequence is partitioned into 30 segments. For each segment, we calculate the ratio of the successful frames among the whole frames. The successful frames are referred to their OPE values bigger than the mild thresholds. For precision plot and overlap plot, the thresholds vary from 0 to 50 and from 0 to 1 separately. For SRE, we sample the initial bounding box in the first frame by shifting and scaling the ground truth. Here, we use

Table 2
The ascension statistics of average overlap for 20 sequences (sequences).

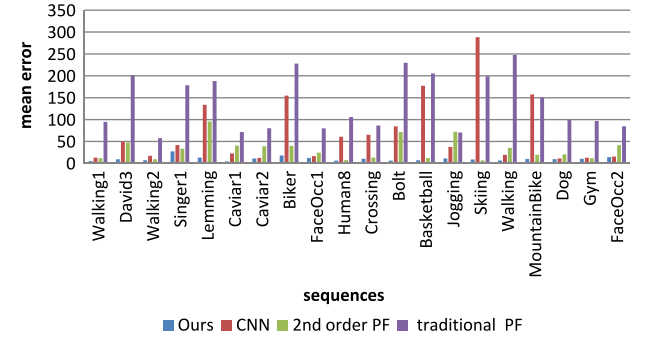
	>10%	>50%	>100%
PF → Ours	18	13	10
CNN → Ours	14	10	8
2nd order → PF Ours	15	7	4

12 spatial shifts including 6 center shifts and 6 corner shifts. Thus, we evaluate each tracker 12 times for SRE over precision plot and overlap plot. Fig. 10 shows partial TRE and SRE results on videos with main nine attributes respectively. From Fig. 10(a) to (i), each column gives overlap plots of SRE, precision plots of SRE ,overlap plots of TRE and precision plots of TRE from the left to the right. The main challenging aspects are background clutter, illumination variation, occlusion, deformation, scale variation, fast motion and out of view, fast motion and in-plane rotation, motion blue, occlusion, rotation from the top to the bottom in Fig. 10. Among the video plots, our results are denoted by the blue curve.

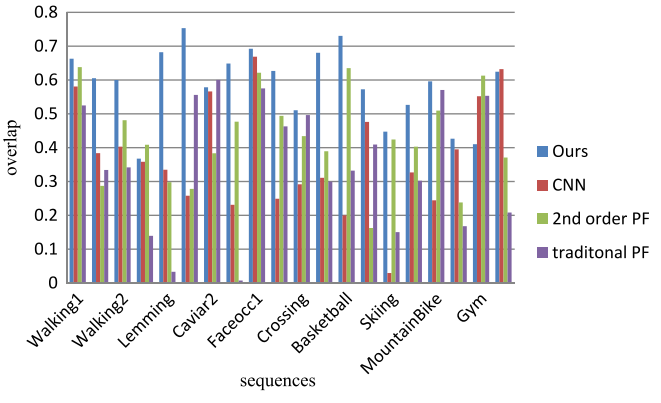
On the videos with attributes of *background clutter*, *illumination variation*, deformation and *occlusion*, our method ranks 1st among all evaluated trackers with four different parameters (shown in Fig. 10(a)–(d)). Maybe these attributes make other trackers difficult to extract effective hand-crafted features from the targets, thereby leading to undesirable results. On the contrary, CNN can help to extract dense and discriminative information across the entire target region, and hence our

Table 3
Average precision scores on individual attributes. Red: best, blue: worst.

	Ours	CNN	DFT	FCT	L1APG	2nd order PF
Walking1	5.3779	12.8731	2.3564	12.5534	6.3798	11.656
David3	9.4386	49.678	26.1301	96.5482	110.3156	47.7797
Walking2	7.1923	17.4059	3.6964	125.8739	83.6182	9.3997
Singer1	27.311	41.9728	20.3325	32.6005	38.299	33.5194
Lemming	13.3243	133.844	54.2555	156.5287	140.3978	95.6639
Caviar1	4.454	22.6494	9.2031	110.7531	96.0934	40.5959
Caviar2	11.2846	12.4095	27.9867	75.3774	41.6977	39.1456
Biker	17.988	154.6249	117.0677	127.8998	20.471	40.0702
FaceOcc1	12.423	16.6144	6.9717	42.1155	15.4717	24.5001
Human8	6.2202	60.7301	38.9751	89.9433	91.9946	7.2184
Crossing	10.4662	65.3404	30.7865	7.784	82.5667	13.2774
Bolt	6.3166	84.6142	357.6193	167.7892	340.7481	71.6245
Basketball	7.1547	177.2422	6.4478	15.3874	11.8957	12.1009
Jogging	11.2467	37.5883	102.5644	91.2456	11.7064	72.3087
Skiing	8.7315	288.3471	282.6739	254.7189	101.9855	6.9031
Walking	6.4915	19.3285	62.0025	5.5297	11.6702	35.4608
MountainBike	9.8949	157.505	6.3023	9.5722	148.7667	19.6397
Dog	9.6455	11.2667	13.6408	11.9087	11.6092	20.6452
Gym	10.4988	12.9391	82.7128	48.7498	74.8013	11.7631
FaceOcc2	14.2483	15.4684	7.3721	18.6169	9.3887	41.5887
Average	10.48543	69.6221	62.95488	75.07481	72.49387	32.74305
Deviation	5.0385	73.2542	92.5213	65.9462	76.5047	23.9216



(a) Mean error comparison.



(b) Overlap comparison.

Fig. 9. Comparisons for traditional PF, 2nd order PF, CNN with FC features and our method.

method brings accurate tracking together with adaptive PF. Therefore, our method, CNN tracker in green and 2nd-order PF in red rank top 3.

For the videos with attributes such as *scale variation*, *fast motion and out of view*, *fast motion and in-plane rotation* seen from Fig. 10(e) to (g), our method almost ranks the 1st on all evaluated algorithms. Sometimes it ranks 2nd at some threshold with a narrow margin to the 1st trackers, such as DFT. On the videos with attributes of motion blue, occlusion and rotation, our proposed method sometime drops down to rank 3rd (seen in Fig. 10(h) and (i)) by a very narrow margin to the other trackers.

From the above, it can be seen that our method outperforms other methods in most challenges. In addition, it is generally better than PF and CNN trackers. It is probably because the fusion of deep features and hand-crafted features is more effective to represent the target. And the adaptive PF is useful to localize target in general.

4.3. Qualitative comparisons

(1) Deformation: Fig. 11(a) is a *Skiing* sequence. Tracking results at frames {5, 9, 13, 17, 59} for all 6 methods are shown. The different tracking methods are color-coded. DFT starts to drift from the target at frame 4 when the target begins to jump, while FCT and CNN start to show target drifting at frame 7 and finally lose tracking at frame 9. L1APG begins to drift from the target from frame 13 and loses the target at frame 17. 2nd-order PF and ours track the target quite well. The target is successfully tracked throughout the entire sequence.

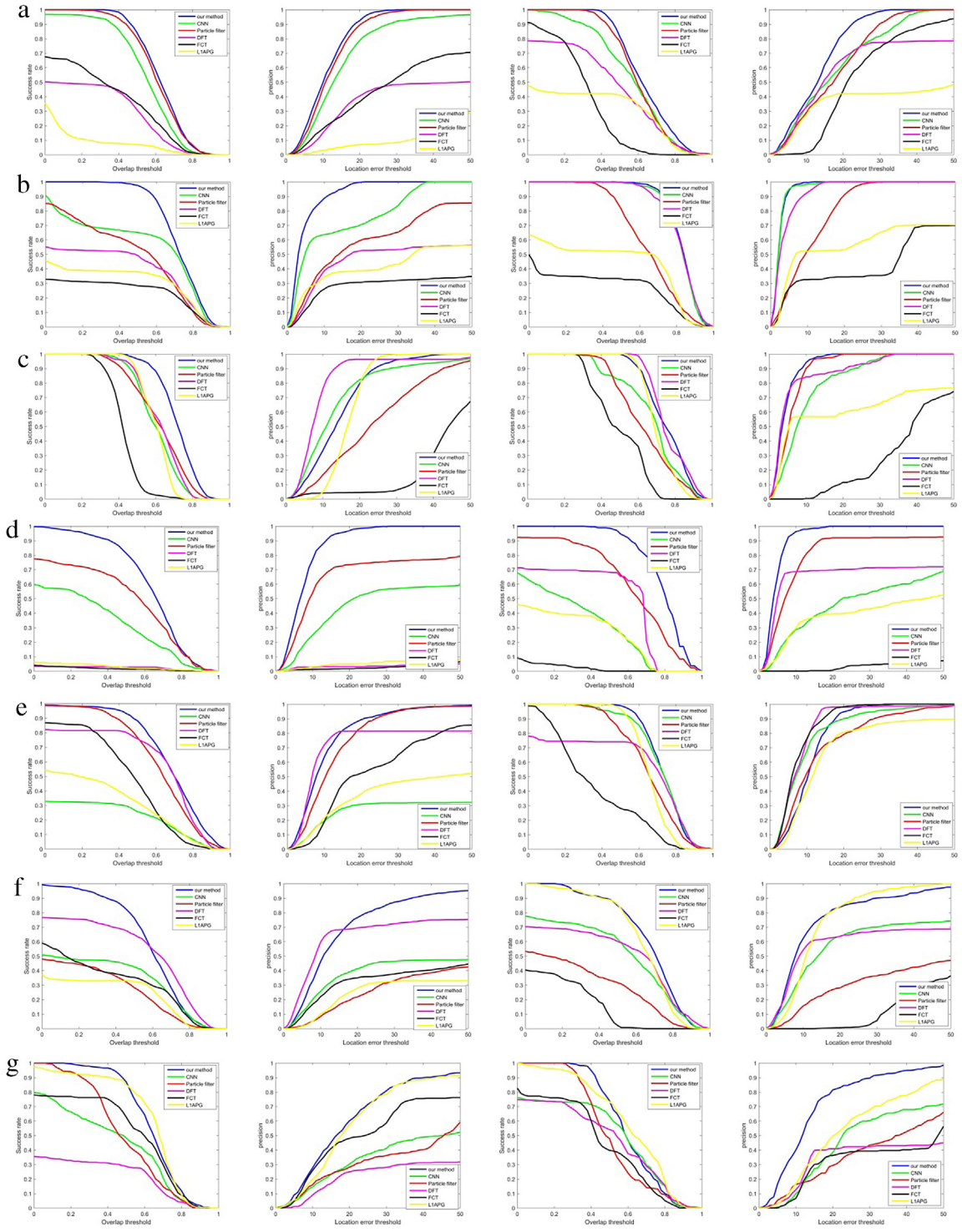
In the *Gym* sequence, the target keeps moving. Her body, arms and legs alter actions continuously. Tracking results for frames {23, 243, 258, 328, 500} are presented in Fig. 11(b). L1APG, DFT and FCT all undergo target drift. The other three methods can track the target through the whole video sequence. From Fig. 11(a) and (b), it can be seen that CNN method will fail when the target moves quickly.

(2) Fast motion. The *Bolt* sequence contains similar background and fast motion, which cause disturbance and motion blur. The tracking results at frames {10, 11, 20, 167, 293} are shown in Fig. 11(c). When the runner in yellow T-shirt speeds up suddenly, FCT and DFT starts to drift at frame 10. LIAPG also loses the target and locates the number plate at frame 20 because of the similar color disturbance. During sprinting, the runner's speed and deformation are very big. CNN and 2nd-order PF both fail tracking. At last, only our algorithm keeps tracking accurately.

In *Biker* sequence, though there are scale change, pose change and fast motion, our algorithm performs well throughout the sequences. Part results at frames {39, 47, 48, 54, 78} are shown in Fig. 11(d).

(3) Occlusion. The *David3* sequence contains occlusion and background disturbance, which cause most of the trackers to drift as shown in Fig. 11(e). Part frame {31, 87, 159, 189, 194} is given. DFT and our method handle these challenges well.

In the *Lemming* sequence, there is abrupt object motion, heavy occlusion and scale changes, which lead most of the trackers to fail. Part results at frames {231, 240, 360, 380, 491} are shown in Fig. 11(f). When the target moves quickly, FCT, L1APG and CNN drift from frame 56 and lose the target at frame 240. Once the target is severely occluded



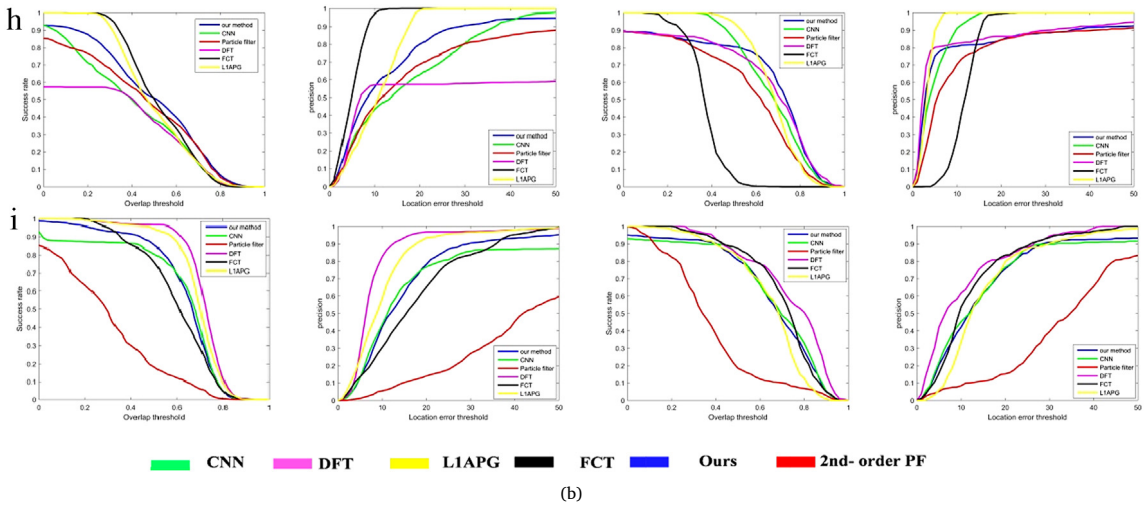
(a)

Fig. 10. Comparison of different methods with nine attributes. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

by other object from frame 360, only our proposed tracker can track the target in this sequence quite well. Other methods fail to track the object reliably.

5. Conclusions

We proposed a novel visual tracking algorithm based on a simplified CNN, where outputs from the FC layer of the pre-trained network are connected with the hand-crafted features from the color histograms in HSV color space. The generic feature and discriminative appearance feature model online PF tracking. The motion information directs the propagation of the particles and improves the adaptivity of PF. The proposed algorithm achieves substantial performance gain over the existing state-of-the-art trackers.



(b)

Fig. 10. (*continued*)

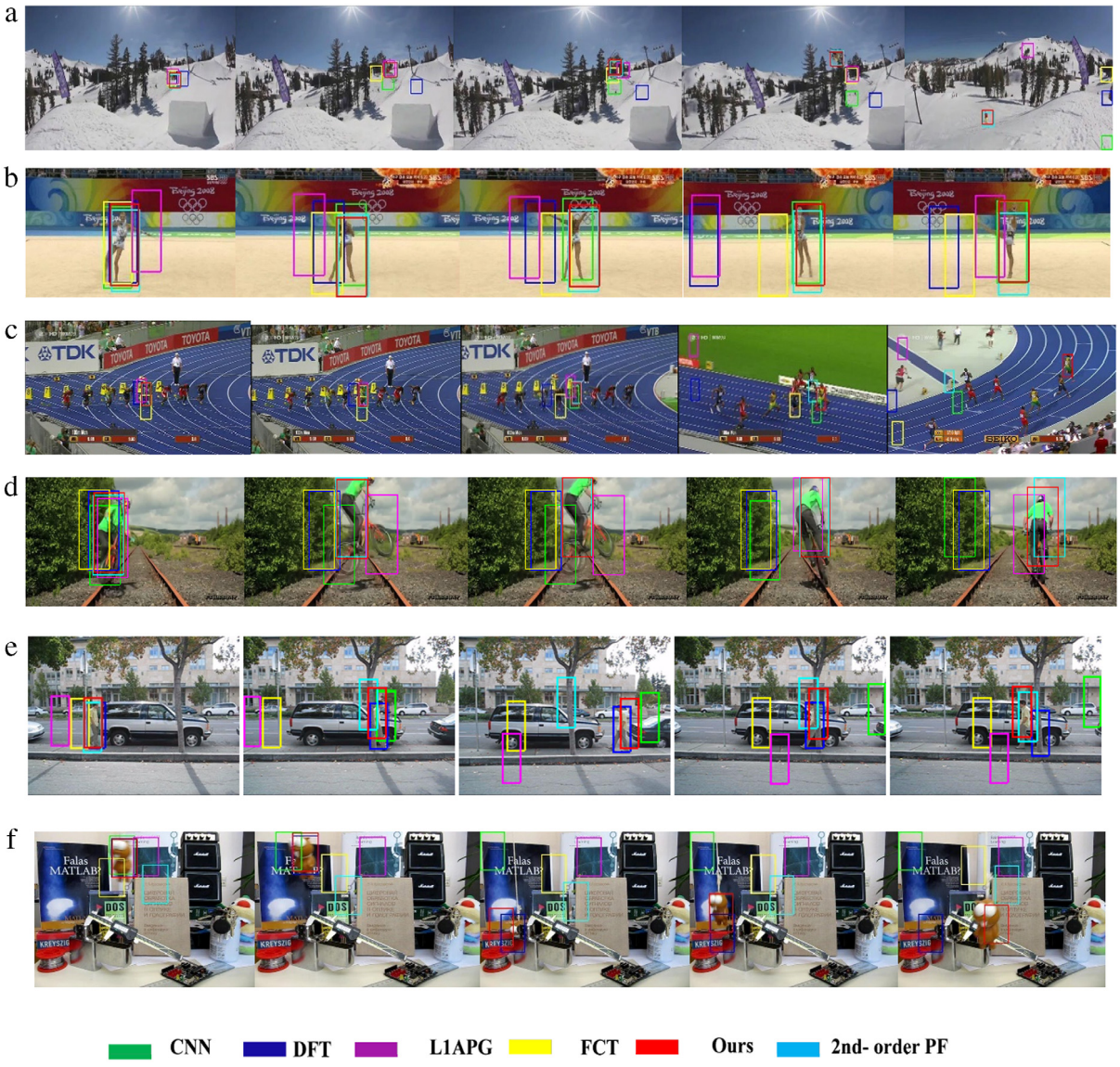


Fig. 11. Qualitative results for selected sequences: (from top to down) *Skiing, Gym, Bolt, Biker, David3* and *Lemming*. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 4
Average overlap rate scores on individual attributes. Red: best, blue: worst.

	Ours	CNN	DFT	FCT	L1APG	2nd order PF
Walking1	0.6628	0.5808	0.7426	0.5923	0.6729	0.6381
David3	0.6053	0.3838	0.5476	0.1087	0.0796	0.2872
Walking2	0.5998	0.4031	0.6373	0.1022	0.3128	0.4812
Singer1	0.3674	0.3584	0.4182	0.4156	0.4116	0.4087
Lemming	0.6821	0.3347	0.5567	0.0717	0.3136	0.2979
Caviar1	0.7532	0.258	0.6741	0.2244	0.2342	0.2782
Caviar2	0.5783	0.5663	0.4049	0.3137	0.2796	0.3833
Biker	0.6486	0.2312	0.2213	0.2161	0.6448	0.4769
FaceOcc1	0.6922	0.6687	0.6475	0.4506	0.6239	0.6214
Human8	0.6268	0.2491	0.2655	0.0361	0.0267	0.4939
Crossing	0.5106	0.2916	0.2969	0.4567	0.1325	0.4342
Bolt	0.6805	0.3109	0.0224	0.0174	0.0284	0.3895
Basketball	0.7305	0.2	0.6932	0.5703	0.6284	0.6349
Jogging	0.5726	0.4761	0.1634	0.1635	0.6241	0.1626
Skiing	0.4472	0.0295	0.0268	0.0542	0.2432	0.4242
Walking	0.5263	0.3268	0.3414	0.5234	0.5131	0.403
MountainBike	0.5961	0.2444	0.6474	0.6292	0.2925	0.5096
Dog	0.4264	0.3949	0.3709	0.415	0.3789	0.2381
Gym	0.4104	0.5518	0.2629	0.3131	0.107	0.6127
FaceOcc2	0.6245	0.6321	0.6827	0.5501	0.6602	0.3707
Average	0.5871	0.3746	0.4311	0.3112	0.3602	0.42731
Deviation	0.1088	0.1633	0.2278	0.2078	0.2256	0.1354

Acknowledgments

This study is supported by the Fundamental Research Funds for the Central Universities (Grant No. NS2017040). Many thanks to the original authors for providing the TB-100 dataset that is publicly available online at http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html.

References

[1] K. Zhang, Q. Liu, Y. Wu, M.H. Yang, Robust visual tracking via convolutional networks without training, IEEE Trans. Image Process. 25 (4) (2016) 1779–1793.
[2] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. New York, NY, IEEE, USA, 2006, pp. 798–805.
[3] N. Alt, S. Hinterstoisser, N. Navab, Rapid selection of reliable templates for visual tracking, in: Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, San Francisco, CA, USA, 2010, pp. 1355–1362.
[4] S.F. He, Q.X. Yang, R.W.H. Lau, J. Wang, M.H. Yang, Visual tracking via locality sensitive histograms, in: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, Portland, OR, USA, 2013, pp. 2427–2434.
[5] D.A. Ross, J. Lim, R.S. Lin, M.H. Yang, Incremental learning for robust visual tracking, Int. J. Comput. Vis. 77 (1–3) (2008) 125–141.
[6] T.Z. Zhang, S. Liu, C.S. Xu, S.C. Yan, B. Ghanem, N. Ahuja, M.H. Yang, Structural sparse tracking, in: Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 150–158.
[7] X. Mei, H. Ling, Robust visual tracking using l1 minimization, in: IEEE International Conference on Computer Vision, 2009, pp. 1436-1443.
[8] M. Amal, B.-P. Jenny, D. Barba, Tracking of objects in video scenes with time varying content, EURASIP J Adv. Sig. Proc. 2002 (6) (2002) 582–594.
[9] Y.F. Zhou, B.-P. Jenny, N. Henri, Multi-object particle filter tracking with automatic event analysis, ARTEMIS@ACM Multimedia (2010) 21–26.
[10] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2011) 1619–1632.
[11] H. Grabner, C. Leistner, H. Bischof, Semi-supervised on-line boosting for robust tracking, in: Proceedings of the 10th European Conference on Computer Vision, Springer, Marseille, France, 2008 234–24.
[12] Y.C. Bai, M. Tang, Robust tracking via weakly supervised ranking SVM, in: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Vol. 20, IEEE, Providence, RI, USA, 2012, pp. 1854–1861.
[13] X.Y. Qian, Y.D. Wang, L. Han, An object tracking method based on guided filter for night fusion, Infrared Phys. Technol. 74 (2016) 38–43.
[14] S. Chen, A. Dehghan, M. Hahn, Deep tracking: Visual tracking using deep convolutional networks, Comput. Sci. (2015).
[15] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks. In NIPS, 2012 1,2,3.
[16] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. In ICLR, 2015, 1,2,3.
[17] M. Isard, A. Blake, Condensation-conditional density propagation for visual tracking, Int. J. Comput. Vis. 29 (1) (1998) 5–28.
[18] P. Perez, C. Hue, J. Vermaak, et al., Color-based probabilistic tracking, in: Proceeding of the European Conference on Computer Vision, Copenhagen, Denmark, 2002, pp. 661–675.
[19] S. Wang, H. Lu, F. Yang, et al., Superpixel tracking, in: Proceeding of the IEEE International Conference on Computer Vision, Barcelona, Spain, 2011, pp. 714–719.
[20] D.A. Ross, J. Lim, R.S. Lin, et al., Incremental learning for robust visual tracking, Int. J. Comput. Vis. 77 (1–3) (2008) 125–141.
[21] X. Mei, H. Ling, Robust visual tracking using L1 minimization, in: Proceedings of the IEEE International Conference on Computer Vision, Kyoto, Japan, 2009, pp. 1436–1443.
[22] X. Jia, H. Lu, M.-H. Yang, Visual tracking via adaptive structural local sparse appearance model, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Providence, USA, 2012, pp. 1822–1829.
[23] Y.X. Wang, Q.J. Zhao, Y.M. Cai, B. Wang, Tracking by auto-reconstructing particle filter trackers, Chinese J. Comput. 39 (7) (2016) 1294–1305.
[24] K. Volkan, B. Mark, W. Wenwu, K. Josef, Audio assisted robust visual tracking with adaptive particle filtering, IEEE Trans. Multimedia 17 (2) (2015) 186–200.
[25] Y. Zhou, H. Nicolas, J. Benois-Pineau, A multi-resolution particle filter tracking in a multi-camera environment, in: 2009 16th IEEE International Conference on Image Processing (ICIP), pp. 4065–4068.
[26] N. Wang, D.-Y. Yeung, Learning a deep compact image representation for visual tracking, in: Proceedings of the 2013 advances in neural information processing systems, Harrahs and Harveys, Lake Tahoe, 2013, pp. 809–817.
[27] S. Hong, T. You, S. Kwak, et al., Online tracking by learning discriminative saliency map with convolutional neural network, Comput. Sci. (2015).
[28] H. Li, Y. Li, F. Porikli, Deep track: Learning discriminative feature representations by convolutional neural networks for visual tracking, in: Proceedings of the British Machine Vision Conference, Nottingham, England, 2014, pp. 1–11.
[29] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: A benchmark, in: 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 2013, pp. 2411–2418.
[30] K. Zhang, L. Zhang, M.H. Yang, Fast compressive tracking, IEEE Trans. Pattern Anal. Mach. Intell. 36 (10) (2014) 2002–2015.
[31] L. Sevilla-Lara, E. Learned-Miller, Distribution fields for tracking, Proc. IEEE Conf. Comput. Vision Pattern Recognit. 157 (10) (2012) 1910–1917.
[32] C. Bao, Y. Wu, H. Ling, H. Ji, Real time robust L1 tracker using accelerated proximal gradient approach, Proc. IEEE Conf. Comput. Vision Pattern Recognit. 157 (10) (2012) 1830–1837.