

# Robust Visual Tracking Using Oriented Gradient Convolution Networks

Qi Xu<sup>(✉)</sup>, Huabin Wang, Jian Zhou, and Liang Tao

Key laboratory of Intelligent Computing and Signal Processing of Ministry of Education, Anhui University, Hefei 230031, China  
935821729@qq.com

**Abstract.** Convolutional networks have been successfully applied to visual tracking to extract some useful feature. However, deep networks are time-consuming to offline training and usually extract the feature from raw pixels. In this paper, we propose a two-layer convolutional network based on oriented gradient. The first layer is constructed by the convolution of the filter and an input image of oriented gradient, which is robust to the illumination variation and motion blur. Then, all of the feature maps of the simple layer are stacked to a complex feature map as the target representation. The complex feature map can encode the local structure feature which is robust to occlusion. The proposed approach is tested on nine challenging sequences in comparison with nine state-of-art trackers, and the result show that the proposed tracker achieves mean overlap rate of 0.75, which outperforms the secondary tracker by 26%.

## 1 Introduction

Visual Tracking is a major issue in computer vision with lots of applications, such as automatic supervision, human computer interaction and vehicle tracking. Given the true position of the first frame, the purpose of visual tracking is to find the position of the target in the successive frames. There are still some challenges because of occlusions, plane rotation, motion blur, illumination variation, etc.

The main influence factors of tracking include feature extractor, observation model, model update and motion model. Recently, lots of state-of-the-art algorithms are proposed, like LOT [1], Struck [2], SCM [3] and ASLA [4], and these methods focus on exploiting hand-crafted features. Although these algorithms perform well in the past, the hand-crafted features are not suitable for all generic objects.

Convolutional neural network (CNN) can exploit some useful features from raw data, so it has extensive applications, such as image classification [5] and object recognition [6] and segmentation [7]. However, traditional CNNs need too much time and samples to offline training for visual tracking. To solve these problems, Fan et al. [8] trained CNN by some auxiliary picture; Zhou et al. [9] proposed a tracking system using various CNNs; Li et al. [10, 11] used multiple image cues to design a lightweight CNN; Zhang et al. [12] exploited a CNN with lightweight structure, which is fully feed-forward and achieves fast tracking even on a CPU.

All of above-mentioned CNN tracking methods pay particular attention to raw pixels for extracting features, and ignore some inherent features like oriented gradient. Gradient mainly exist in the brim of the target and some parts whose pixels change sharply. And oriented gradient is the most change rate of image gray value, which can reflect the salience of the target. The result of the convolution of a function and unit impulse is similar to copying the function in the position of the unit impulse, so only the value of the parts with similar oriented gradient is much larger when filters extracted from the oriented gradient convolve around the input image. Based on this observation, in this paper, we propose an oriented gradient convolutional network (COG) formulated within a particle filtering framework for object tracking. To build it, we first warp the image to a fixed size and get some patches by sliding windows. Each patch consists of oriented gradient about the object. Then, a overcomplete dictionary is studied from the image patches as a filter. Motivated by the work in [12], we propose tracking system using full feed-forward convolutional network, such that convolutional network is only simple two-layer. The first layer is simple layer, which is constructed by the convolution of the filter and image patches. And the complex layer consists of complex cell feature map that is a tensor, which stacks simple cell feature maps.

## 2 Particle Filter

Particle filter is an algorithm which could estimate the posterior distribution of state based on the Bayesian sequential importance sampling technique and Monte Carlo simulation. Because of estimating the state by some samples, it is applicable for non-linear system. And it has been widely used in visual tracking [3, 4]. The process of particle filter includes two steps: predicting and updating. The predicting distribution  $p(x_t|z_{1:t-1})$  which is given by all observations  $z_{1:t-1} = \{z_1, z_2, \dots, z_{t-1}\}$  up to time  $t - 1$ , can be computed as

$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})dx_{t-1} \quad (1)$$

where  $x_t$  denotes the state variable which describe the affine motion parameters at time  $t$ . When the observation  $z_t$  is available at time  $t$ , the state vector can be updated

$$p(x_t|z_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} \quad (2)$$

where  $p(z_t|x_t)$  denotes the observation likelihood, and  $p(z_k|z_{1:k-1})$  is a normalized constant which could be computed as

$$p(z_t|z_{1:t-1}) = \int p(z_t|x_t)p(x_t|z_{1:t-1})dx_t \quad (3)$$

Considering that the integrals above-mentioned are difficult to be computed, the Monte Carlo simulation and importance sampling are utilized to approximate

the  $p(z_t|z_{1:t})$  by  $N$  samples which denote  $\{x_t^i\}_{i=1}^N$ . The weights of samples denoted  $w_t^i$  are updated as

$$w_t^i = w_{t-1}^i \frac{p(z_t|x_t^i)p(x_t^i|x_{t-1}^i)}{q(x_t|x_{1:t-1}, z_{1:t})} \quad (4)$$

where  $q(x_t|x_{1:t-1}, z_{1:t})$  denotes importance distribution. Because of sample degenerating, the samples are resampled to duplicate the particles with high weight and abandon the low ones.

In this paper, the importance distribution is set to  $q(x_t|x_{1:t-1}, z_{1:t}) = p(x_t|x_{t-1})$  and the weights become the observation likelihood  $p(z_t|x_t)$ . We also warp the input image to a fixed size with  $m \times m$  pixels. Then, the state variable could be set to  $x_t = (t_x, t_y, \alpha_1, \alpha_2, \alpha_3, \alpha_4)$ , where  $\{t_x, t_y\}$  are translation parameters and  $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$  are the deformation parameters.

### 3 Oriented Gradient Convolutional Network

#### 3.1 Preprocessing

After the input image is warped to a fixed size, the gradient of each pixel can be computed as

$$\mathbf{G}_x(x, y) = \mathbf{I}^{m \times m} \otimes \mathbf{I}_o, \mathbf{G}_y(x, y) = \mathbf{I}^{m \times m} \otimes \mathbf{I}_o^T \quad (5)$$

where  $\mathbf{I}^{m \times m}$  is the wrapped image, which is preprocessed by gamma correction that is correspond to local brightness and contrast moralization, and  $\mathbf{I}_0 = [-1, 0, 1]$ . Then the oriented gradient can be formulated as

$$\alpha(x, y) = \tan^{-1} \frac{\mathbf{G}_y(x, y)}{\mathbf{G}_x(x, y)} \quad (6)$$

Therefore, a given fixed input image is constructed by the oriented gradient. We sample  $(m - n + 1) \times (m - n + 1)$  patches centered at each pixel location inside the  $\hat{\mathbf{I}}^{m \times m}$  by sliding a window with  $n \times n$ . Finally, in each patch, we divide the oriented gradient into  $R$  sections, and get a feature vector of  $R$  dimensions denoted as  $\mathbf{Y}_{K \times K}^R$ ,  $K = (m - n + 1)$  through counting the number of each section.

#### 3.2 Design Filter

Inspired by the spare representation [13]: the most likely representation of the image can be achieved by over-complete sparse representation, which also can obtain higher resolution information than traditional non adaptive methods. So we use the overcomplete dictionary to represent the target region adaptively. Then, the filter can be formulated as

$$\mathbf{F}_{K \times W} = \mathbf{Y}_{K \times K}^R \mathbf{X}_{K \times W} \quad (7)$$

where  $\mathbf{X}$  is a coefficient matrix, which should be as sparse as possible. The Eq. (7) can be rewritten as

$$\min_{\mathbf{F}, \mathbf{X}} \{ \|\mathbf{X}_i\|_0 \} \quad s.t. \quad \|\mathbf{Y} - \mathbf{F}\mathbf{X}\|_F^2 \leq \varepsilon \quad (8)$$

To solve the above Equation, supposing the dictionary  $\mathbf{F}$  is fixed, a method called Orthogonal Matching Pursuit is used to achieve sparse coding

$$\min_{\mathbf{x}_i} \{ \|\mathbf{Y}_i - \mathbf{F}\mathbf{x}_i\|_2^2 \} \quad s.t. \quad \|\mathbf{Y} - \mathbf{F}\mathbf{X}\|_F^2 \leq \varepsilon \quad (9)$$

where  $\mathbf{X}$  is a sparse parameter. Then,  $\mathbf{F}\mathbf{X}$  can be computed as

$$\mathbf{F}\mathbf{X} = \sum_{i=1}^W \mathbf{f}_i \mathbf{x}_i^T \quad (10)$$

where  $\mathbf{f}_i$  is the  $i$ -th column in the  $\mathbf{F}$  and  $\mathbf{x}_i^T$  is the  $i$ -th row in the  $\mathbf{X}$ . Then we need update the dictionary through several iterations. When the  $k$ -th column is updated, the others are supposed to be fixed. Then, the Eq. (9) can be rewritten as

$$\begin{aligned} \min_{\mathbf{F}} \|\mathbf{Y} - \mathbf{F}\mathbf{X}\|_F^2 &= \min_{\mathbf{f}_k} \|\mathbf{Y} - \sum_{i=1}^W \mathbf{f}_i \mathbf{x}_i^T\|_F^2 \\ &= \min_{\mathbf{f}_k} \|\mathbf{Y} - \sum_{i \neq k} \mathbf{f}_i \mathbf{x}_i^T - \mathbf{f}_k \mathbf{x}_k^T\|_F^2 \\ &= \min_{\mathbf{f}_k} \|\mathbf{E}_k - \mathbf{f}_k \mathbf{x}_k^T\|_F^2 \end{aligned} \quad (11)$$

where  $\mathbf{E}_k$  is a fixed error matrix. Then, two vectors should be found in  $\mathbf{E}_k$  to update  $\mathbf{f}_i$  and  $\mathbf{x}_i^T$ . In [14], author used the Singular Value Decomposition (SVD) method to solve this problem,

$$\mathbf{E}_k = \mathbf{U} \mathbf{\Lambda} \mathbf{V}^T \quad (12)$$

where  $\mathbf{U}$  and  $\mathbf{V}$  are orthogonal basis, and  $\mathbf{\Lambda}$  is a diagonal matrix. The vector of  $\mathbf{U}$  which corresponds to the maximum value of  $\mathbf{\Lambda}$  is accepted as  $\mathbf{f}_i$ . However, the sparsity of  $\mathbf{X}$  may be corrupted if we choose the corresponding vector in  $\mathbf{V}$  to update  $\mathbf{x}_i^T$ . The solution is to construct a new matrix  $\Omega_{K \times L}$  which consists of the nonzero element in  $\mathbf{x}_i^T$

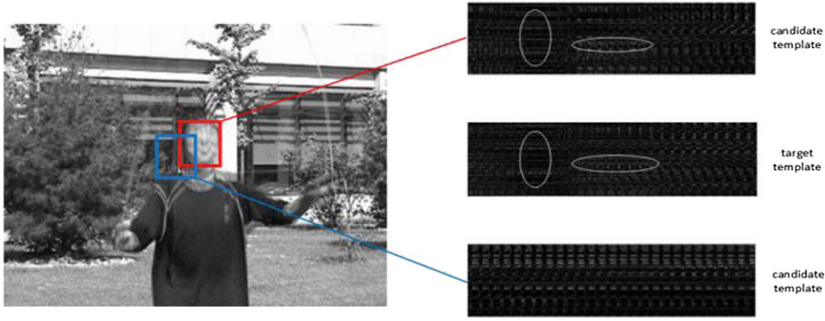
$$\hat{\mathbf{E}}_k = \mathbf{E}_K \Omega_{K \times L}, \hat{\mathbf{x}}_k = \mathbf{x}_k^T \Omega_{K \times L} \quad (13)$$

Then, a sparse  $\mathbf{x}_i^T$  can be obtained when using SVD to  $\hat{\mathbf{E}}_k$ .

### 3.3 Object Tracking

After getting the filter  $\mathbf{F}$ , the region of each particle is preprocessed, denoted as  $\mathbf{P}$ . Then, the simple layer can be defined as

$$\mathbf{S} = \mathbf{F} \otimes \mathbf{P} \quad (14)$$



**Fig. 1.** The simple cell feature map can preserve the local structure of the target.

The simple layer  $\mathcal{S}$  enhances the outline and structure of object region, which can be remained while illumination variation and motion blur. So the position of the object can be discriminated according to the geometric layout information of the simple layer. To further strengthen the effect of representation, a 3D tensor is used to construct a complex feature map in [12],

$$\mathbf{C} \in \mathbf{R}^{K \times K \times N} \tag{15}$$

which can enhance the strength of local structural. So the complex cell feature map  $\mathbf{C}$  is robust to occlusion and deemed as the candidate template. In this paper, the state transition distribution was model by a Gaussian distribution, and observation results are independent which could be computed as

$$p(z_t|x_t) = e^{-\|c_t - c_t^i\|_2^2} \tag{16}$$

where  $c_t$  and  $c_t^i$  are the target template and  $i$ -th candidate template respectively at frame  $t$ .

### 3.4 Model Update

Filter and target template  $\mathbf{c}$  should be updated incrementally to accommodate appearance changes. The target template can be update as

$$\mathbf{c}_t = (1 - \rho)\mathbf{c}_{t-1} + \rho\hat{\mathbf{c}}_{t-1} \tag{17}$$

where  $\rho$  is fixed parameter which is set to 0.05 in our experiments,  $\mathbf{c}_t$  represent the target template at frame  $t$ , and  $\hat{\mathbf{c}}_{t-1}$  is the sparse representation of the target template at frame  $t - 1$ , which can be easily achieved by a soft shrinkage function

$$\hat{\mathbf{c}} = \text{sign}(\text{vec}(\mathbf{C})) \max(0, \text{abs}(\text{vec}(\mathbf{C}) - \text{median}(\text{vec}(\mathbf{C}))) \tag{18}$$

The filter is updated as

$$\mathbf{F}_{t+1} = \lambda\mathbf{F}_o + \lambda\mathbf{F}_t + (1 - 2\lambda)\mathbf{F}_{t-1} \tag{19}$$

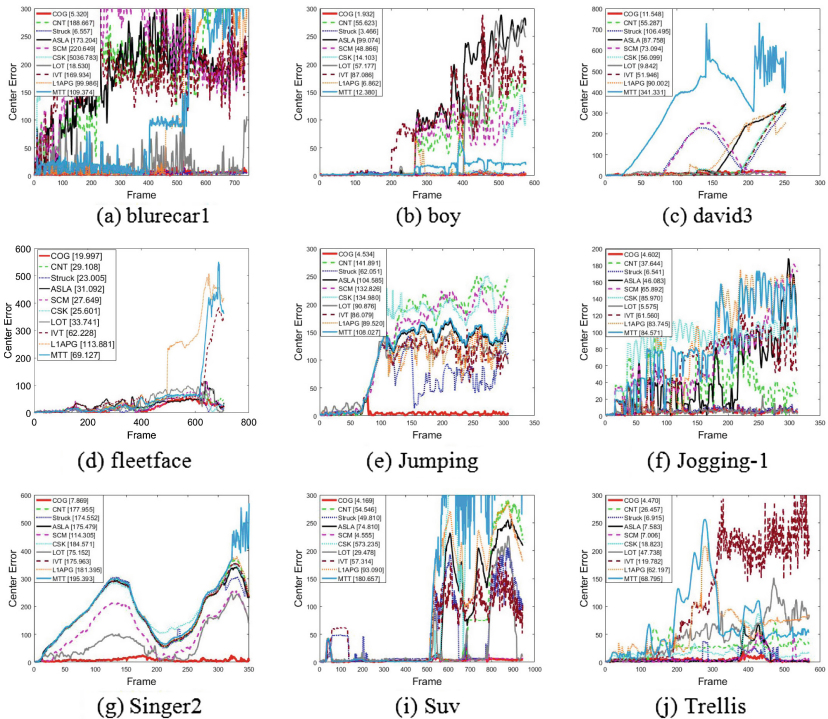
where  $\mathbf{F}_o$  is the filter in first frame,  $\lambda$  is a fixed parameter which is set to 0.15 in our experiments,  $\mathbf{F}_t$  is the filter at frame  $t$ , and  $\mathbf{F}_{t-1}$  is the filter at frame  $t - 1$ . To improve the speed, we update the filter every other 5 frames.

## 4 Experience and Discussion

### 4.1 Experimental Setup

Our tracker was implemented in matlab2016 on a PC with Intel Core i7-7700 CPU@ 3.5GHz, and run at approximately 1 frame per second. To compare the robustness of our algorithm, we use the Visual Tracking Benchmark dataset [15] and its outstanding code library, including IVT [16], L1APG [17], ASLA [4], LOT [1], MTT [18] and SCM [3]. Furthermore, we also add the CNT algorithm, which could be downloaded on authors homepage. Nine challenging sequences from visual tracking benchmark are used for comparison, including blurcar1, boy, david3, fleetface, jumping, jogging-1, singer2, suv and trellis.

Several parameters are used in COG tracker. The variance of particle filter is set to  $\lambda_t = 4 + (v_{t-1} + v_{t-2} + v_{t-3})/3$ , where  $v_{t-1}$ ,  $v_{t-2}$  and  $v_{t-3}$  are the targets movement speed at frame  $t$ ,  $t-1$  and  $t-2$  respectively. Deformation parameters  $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$  are set to  $\{0.02, 0.005, 0.005, 0\}$ . The size of the warped image is set to  $m = 32$ , and the size of sliding window is set to  $n = 6$ .



**Fig. 2.** Tracking results in terms of center error (in pixel). The COG is compared with 9 state-of-the-art algorithms on 9 challenging image sequences

**Table 1.** Average overlap rate of 10 algorithm. The best and second results are shown in bold-font and italic-font.

Sequences	COG	CNT	Struck	ASLA	SCM	CSK	LOT	IVT	L1APG	MTT
blurcar1	<b>0.81</b>	0.11	<i>0.79</i>	0.06	0.05	0.01	0.67	0.04	0.49	0.36
boy	<b>0.81</b>	0.40	<i>0.78</i>	0.38	0.39	0.68	0.55	0.27	0.74	0.51
david3	<b>0.73</b>	0.56	0.30	0.43	0.40	0.50	<i>0.67</i>	0.48	0.38	0.10
fleetface	<b>0.64</b>	0.58	<i>0.61</i>	0.57	0.60	0.59	0.57	0.46	0.46	0.51
jumping	<b>0.64</b>	0.08	<i>0.62</i>	0.23	0.12	0.05	0.58	0.12	0.15	0.10
jogging-1	<b>0.78</b>	<i>0.19</i>	0.17	0.18	0.18	0.18	0.09	0.18	0.17	0.18
singer2	<b>0.74</b>	0.04	0.04	0.04	0.17	0.04	<i>0.26</i>	0.04	0.04	0.04
suv	<b>0.83</b>	<i>0.80</i>	0.52	0.49	0.75	0.52	0.65	0.41	0.48	0.45
trellis	<i>0.78</i>	0.42	0.61	<b>0.80</b>	0.67	0.48	0.31	0.26	0.20	0.22
average	<b>0.75</b>	0.35	<i>0.49</i>	0.35	0.37	0.34	0.48	0.25	0.35	0.28

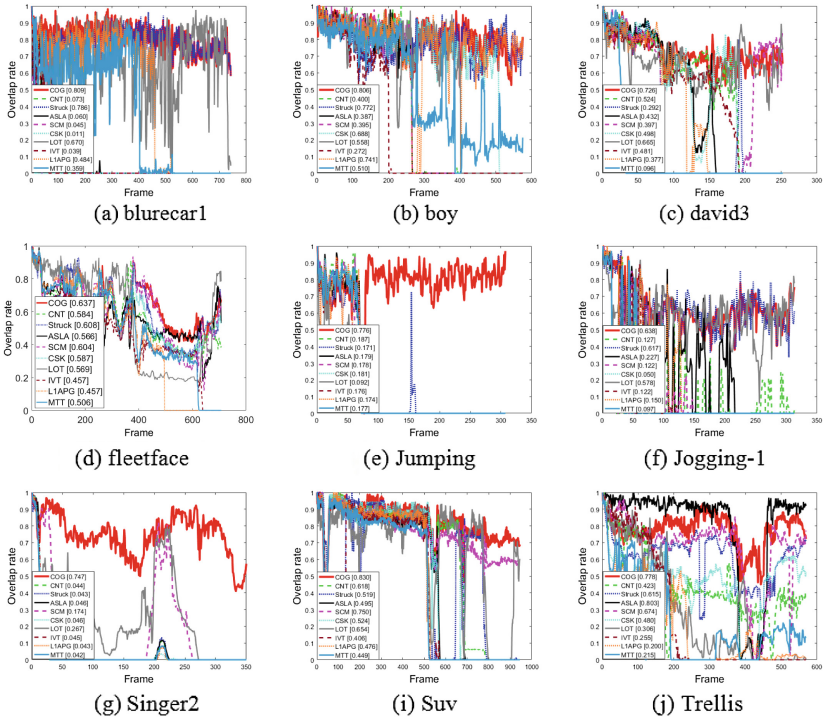
**Table 2.** Successful rate based on Fig. 3. The best and second results are shown in bold-font and italic-font.

Sequences	COG	CNT	Struck	ASLA	SCM	CSK	LOT	IVT	L1APG	MTT
blurcar1	<b>1</b>	0.05	<i>0.98</i>	0.05	0.03	0.01	0.78	0.04	0.61	0.49
boy	<b>1</b>	0.45	<i>1</i>	0.45	0.45	0.88	0.68	0.34	0.91	0.50
david3	<b>1</b>	0.68	0.33	0.51	0.48	0.62	<i>0.93</i>	0.63	0.45	0.11
fleetface	<b>0.91</b>	0.69	0.69	0.63	<i>0.77</i>	0.68	0.60	0.53	0.62	0.62
jumping	<b>0.91</b>	0.11	<i>0.80</i>	0.16	0.12	0.05	0.78	0.09	0.12	0.09
jogging-1	<b>0.98</b>	<i>0.22</i>	0.22	0.22	0.21	0.22	0.06	0.22	0.22	0.22
singer2	<b>0.95</b>	0.04	0.04	0.04	<i>0.17</i>	0.04	0.16	0.04	0.04	0.04
suv	<b>0.98</b>	0.70	0.57	0.58	<i>0.97</i>	0.57	0.78	0.44	0.53	0.53
trellis	<b>0.98</b>	0.24	0.78	<i>0.86</i>	0.85	0.59	0.31	0.30	0.15	0.19
average	<b>0.97</b>	0.35	<i>0.60</i>	0.39	0.45	0.40	0.56	0.29	0.40	0.31

## 4.2 Qualitative Comparisons

**Motion Blur:** Target region would become blurred due to the motion of target or camera. In blurcar1 sequences (Fig. 4(a)), camera moves so swift that target is blurred. Only the COG and Struck perform well in the entire sequence. As show in Fig. 4(e), The motion blur caused by drastic appearance change in the jumping sequence that only COG, Stuck and LOT is not failed after frame 16. The main reason that proposed COG successfully keeps track of the target with motion blur is that the simple feature map is based on oriented gradient, which is robust to motion blur (see. Fig. 1).

**Plane Rotation:** The target may be deformable while rotating in or out of the image plane. It also led to motion blur if its speed of rotation is too high. For the



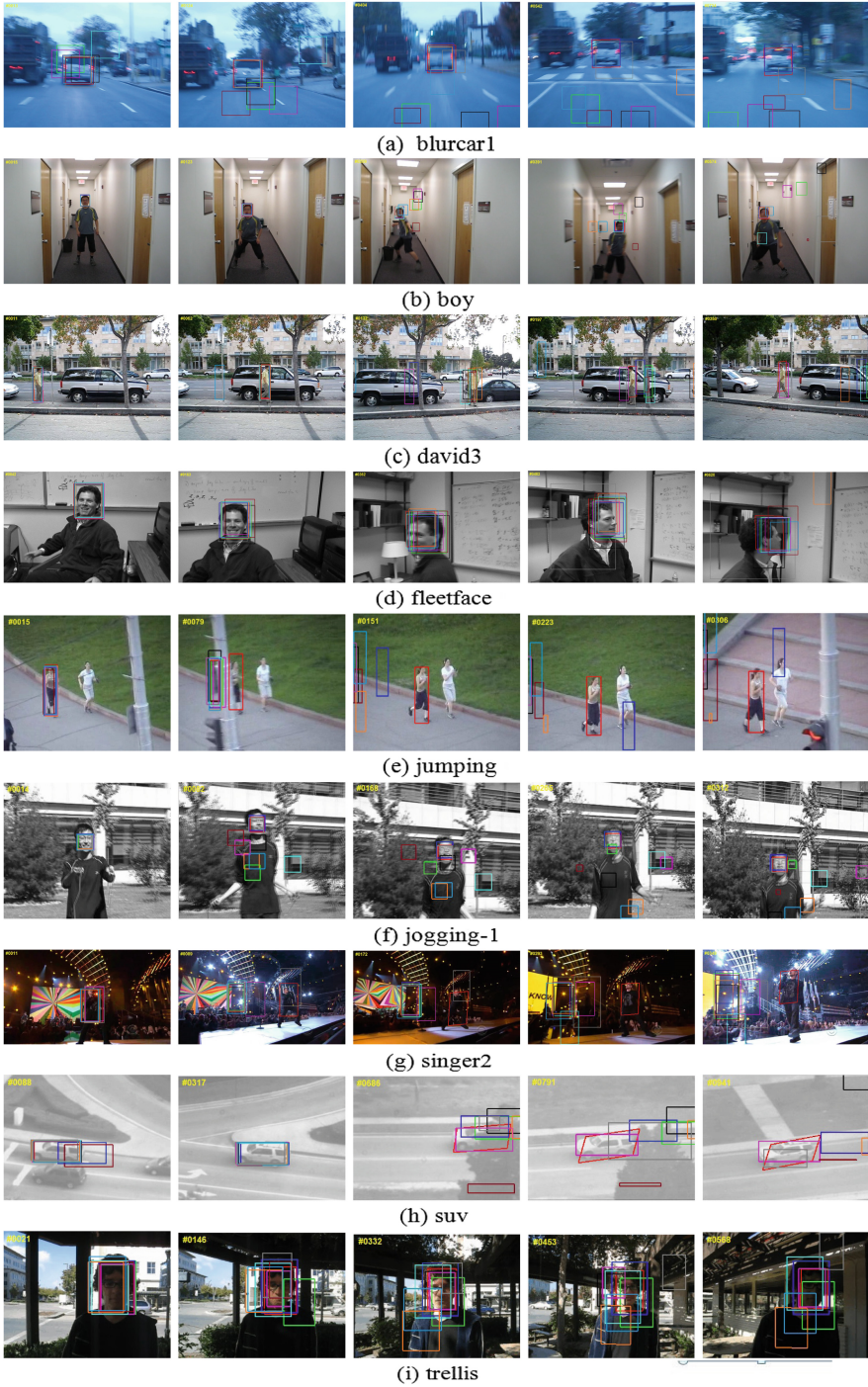
**Fig. 3.** Tracking results in terms of center error (in pixel). The COG is compared with 9 state-of-the-art algorithms on 9 challenging image sequences

boy sequence (Fig. 4(b)), the boy rotates both in and out of plane. Only COG and Stuck perform well in all sequence. In the fleetface sequence (Fig. 4(d)), there are both in-plane and out-of-plane rotations after frame 375. Except for COG and Stuck algorithms, the others all drift. The COG deals with plane rotation well because its online updated scheme is suit for appearance variation.

**Illumination Variation:** The reasons for illumination variation include different colour and varying levels of light. In singer2 sequence (Fig. 4(g)), the colour of back-ground illumination change so drastic that only the COG algorithm can track the target stably in the entire sequence. In trellis sequences (Fig. 4(i)), only the COG and ASLA algorithms perform well while the strength of illumination is varied. The proposed COG algorithm well handles the situation with illumination variations as the feature is extracted from the normalized local filters with gamma correction and local brightness normalization.

**Occlusion:** The target may be partially or fully occluded by other object. In david3 sequence (Fig. 4(c)), only the COG and LOT algorithm perform well when the target is partially occluded by the tree (e.g., frame 84, frame 190). In jogging-1 sequences (Fig. 4(f)), the target is completely occluded by a lamppost (e.g., frame 74, frame 78). Only the COG is able to detect the person when the





**Fig. 4.** Tracking results in terms of center error (in pixel). The COG is compared with 9 state-of-the-art algorithms on 9 challenging image sequences (Color figure online)

target appears again in the screen (e.g., frame 80). Our COG tracker achieves stable performance for occlusion due to employing the local features in complex feature maps.

### 4.3 Quantitative Comparisons

For quantitative comparison, the center position error plot and overlap rate plot are employed. The center position error is defined as

$$e = \sqrt{(x_t - x_s)^2 + (y_t - y_s)^2} \quad (20)$$

where  $x_t$  with  $y_t$  is the center position of the tracking result, and  $x_s$  with  $y_s$  is the ones of the true state. The overlap rate is defined as

$$S = \frac{Area(R_t \cap R_s)}{Area(R_t \cup R_s)} \quad (21)$$

where  $R_t$  represent the area of tracking bounding box, and  $R_s$  represent the area of true state. The tracking result is successful if the value of overlap rate is greater than 0.5 in current frame. And the successful rate is defined as the number of success frames divided by the number of all frames.

Figure 2 illustrates the center position error of each frame in terms of the results of 10 algorithms, while Fig. 3 shows the corresponding overlap rate plot. The results indicate that the center position errors of COG keep the value at relatively low level and the corresponding overlap rate keep higher level in all sequence. Table 1 shows the average overlap rate of each algorithm. Overall, the proposed COG achieves mean overlap rate of 0.75, which outperforms LOT by 26%. Meanwhile, in the success rate which shows in Table 2, its score is 0.97, outperforms significantly Struck by 37%.

## 5 Conclusion

In this paper, we have proposed and demonstrated an effective and robust tracking method based on the oriented gradient convolutional networks. The tracker is constructed by a simple two-layer convolutional network and formulated within a particle filtering framework. First, we warp the input images to a fixed size and extract a set of normalized patches of oriented gradient by sliding window, which can handle illumination variation. To obtain a spare representation of the target, we exploit the overcomplete dictionary from the normalized patches as filters. Then, the first layer is constructed from the convolution of filters and input images, which is robust to motion blur. Finally, we stack all the simple feature maps to construct the complex feature map as the representation of the target which can overcome drifts and occlusions. Furthermore, both the latest observations and the original filter are considered in our update scheme, which can deal with appearance change and the drift problem well. Compared with other nine state-of-the-art algorithms, the experimental results of quantitative and qualitative comparisons on 9 challenging image sequences show the robustness of proposed tracking algorithm.

## References

1. Oron, S., Bar-Hillel, A., Levi, D., et al.: Locally orderless tracking. In: *Computer Vision and Pattern Recognition*, pp. 1940–1947. IEEE (2012)
2. Hare, S., Golodetz, S., Saffari, A., et al.: Struck: structured output tracking with kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**(10), 2096 (2016)
3. Yang, M.H., Lu, H., Zhong, W.: Robust object tracking via sparsity-based collaborative model. In: *Computer Vision and Pattern Recognition*, pp. 1838–1845. IEEE (2012)
4. Lu, H., Jia, X., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, pp. 1822–1829 (2012)
5. Wu, J., Yu, Y., Huang, C., et al.: Deep multiple instance learning for image classification and auto-annotation. In: *Computer Vision and Pattern Recognition*, pp. 3460–3469. IEEE (2015)
6. Wang, A., Lu, J., Cai, J., et al.: Large-margin multi-modal deep learning for RGB-D object recognition. *IEEE Trans. Multimed.* **17**(11), 1887–1898 (2015)
7. Pan, H., Wang, B., Jiang, H.: Deep learning for object saliency detection and image segmentation. *IEEE Trans. Neural Netw. Learn. Syst.* **27**(6), 1135–1149 (2015)
8. Fan, J., Xu, W., Wu, Y., et al.: Human tracking using convolutional neural networks. *IEEE Trans. Neural Netw.* **21**(10), 1610–1623 (2010)
9. Zhou, X., Xie, L., Zhang, P., et al.: An ensemble of deep neural networks for object tracking. In: *IEEE International Conference on Image Processing*, pp. 843–847. IEEE (2015)
10. Li, H., Li, Y., Porikli, F.: Robust online visual tracking with a single convolutional neural network. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) *ACCV 2014 Part V. LNCS*, vol. 9007, pp. 194–209. Springer, Cham (2015). [https://doi.org/10.1007/978-3-319-16814-2\\_13](https://doi.org/10.1007/978-3-319-16814-2_13)
11. Li, H., Li, Y., Porikli, F.: Deep track: learning discriminative feature representations online for robust visual tracking. *IEEE Trans. Image Process.* **25**(4), 1834–1848 (2016). A Publication of the IEEE Signal Processing Society
12. Zhang, K., Liu, Q., Wu, Y., et al.: Robust tracking via convolutional networks without learning. *IEEE Trans. Image Process.* **25**(4), 1779–1792 (2015). A Publication of the IEEE Signal Processing Society
13. Lu, J., Zhang, Q., Xu, Z., et al.: Image super-resolution by dictionary concatenation and sparse representation with approximate  $L_0$  norm minimization. *Comput. Electr. Eng.* **38**(5), 1336–1345 (2012)
14. Elad, M., Aharon, M.: Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process.* **15**(12), 3736–3745 (2006)
15. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: a benchmark. In: *Computer Vision and Pattern Recognition*, pp. 2411–2418. IEEE (2013)
16. Ross, D.A., Lim, J., Lin, R.S., et al.: Incremental learning for robust visual tracking. *Int. J. Comput. Vis.* **77**(1), 125–141 (2008)
17. Ji, H.: Real time robust L1 tracker using accelerated proximal gradient approach. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1830–1837. IEEE Computer Society (2012)
18. Zhang, T., Ghanem, B., Liu, S., et al.: Robust visual tracking via multi-task sparse learning. In: *Computer Vision and Pattern Recognition*, pp. 2042–2049. IEEE (2012)