Computational Visual Media
DOI 10.1007/s41095-017-0083-7 Vol. 3, No. 3, September 2017, 285–294

Research Article

Robust tracking-by-detection using a selection and completion mechanism

Ruochen Fan[1], **Fang-Lue Zhang**[2], **Min Zhang**[3] (✉), **and Ralph R. Martin**[4]

Abstract It is challenging to track a target continuously in videos with long-term occlusion, or objects which leave then re-enter a scene. Existing tracking algorithms combined with online-trained object detectors perform unreliably in complex conditions, and can only provide discontinuous trajectories with jumps in position when the object is occluded. This paper proposes a novel framework of *tracking-by-detection using selection and completion* to solve the abovementioned problems. It has two components, tracking and trajectory completion. An offline-trained object detector can localize objects in the same category as the object being tracked. The object detector is based on a highly accurate deep learning model. The object selector determines which object should be used to re-initialize a traditional tracker. As the object selector is trained online, it allows the framework to be adaptable. During completion, a predictive non-linear autoregressive neural network completes any discontinuous trajectory. The tracking component is an online real-time algorithm, and the completion part is an after-the-event mechanism. Quantitative experiments show a significant improvement in robustness over prior state-of-the-art methods.

Keywords object tracking; detection; proposal selection; trajectory completion

1 Tsinghua University, Beijing 100084, China. E-mail: frc16@mails.tsinghua.edu.cn.

2 School of Engineering and Computer Science, Victoria University of Wellington, Wellington, New Zealand. E-mail: z.fanglue@gmail.com.

3 Center of Mathematical Sciences and Applications, Harvard University, Cambridge, Massachusetts, USA. E-mail: mzhang@math.harvard.edu (✉).

4 School of Computer Science and Informatics, Cardiff University, Cardiff, Wales, UK. E-mail: ralph.martin@cs.cardiff.ac.uk.

Manuscript received: 2017-02-05; accepted: 2017-04-07

1 Introduction

Object tracking aims to acquire the moving trajectories of objects of interest in video, and is a fundamental problem in computer vision. It plays a key role in applications like surveillance analysis [1, 2] and traffic monitoring [3, 4]. Decades of research have led to tremendous progress in this field. However, there is still a long way to go to achieve satisfactory results in many challenging videos with, e.g., violent shaking, long-term occlusion, or objects which leave then re-enter the scene. Traditional tracking methods achieve high accuracy in experimental tests, but perform poorly on practical problems. In most methods, object features are extracted in each frame and used to search for the object in the subsequent frame [5, 6]. Errors can accumulate in this process. If occlusion or frame skipping occurs, tracking will fail because of the rapid change of appearance features in local windows.

Combining detection with tracking is a feasible solution to these problems [7]. In the tracking process, errors can accumulate, but a detector can be used to localize the object being tracked and re-initialize the tracker. It is evident that detection accuracy is essential, so a high decision threshold is set for the detector. This means that detection results are accurate but frequently unavailable. In recent years, deep learning has made significant strides in object detection. However, adaptive online training is still an open problem. The computational requirements of training and lack of training data make it hard to recognize a specific target amongst other objects which belong to the same category in a scene. Furthermore, most tracking frameworks can only provide a discontinuous trajectory with jumps

in position when an object is occluded. However, in application scenarios such as safety monitoring, for reliable analysis results, we must infer the missing parts of occluded trajectories.

For these reasons, this work has designed a novel framework that decomposes the tracking task into two parts: tracking and trajectory completion. During the tracking stage, three steps are invoked for every frame, including a simple tracker, a detection module, and a selection module. This allows the object to be tracked throughout the video. The tracker attempts to follow the object from one frame to another. In this process, tracking errors may accumulate, and if the object is occluded for a long term, the tracker will fail to follow the object. Thus we use the object detector and the object selector to determine the accurate location of the object to re-initialize the tracker. The object detector's job is to localize objects of the same kind as the object being tracked. The task of the object selector is to discriminate between them and determine which object should be used to re-initialize the tracker. For accuracy, we set a high decision threshold for both object detector and object selector, so the recall rate is low. Thus, the location of the object cannot be obtained in every frame by the detector and the selector. However, once the object is localized, the tracker will be re-initialized.

During the completion phase, we use a predictive neural network to complete the discontinuous trajectory. While the missing parts of the trajectory could be interpolated by a simple curve, e.g., a Hermite cubic spline, this is not a good approach as the missing trajectory may not be smooth or regular. We instead use a neural network, which is capable of learning the more complex behaviour of a real trajectory.

Our experimental results show that our method outperforms previous methods in cases in which the target objects are occasionally occluded, and can generate reliable trajectories for such objects.

2 Related work

2.1 Object tracking

Object tracking is the task of estimating the trajectory of a moving target in video. Traditional tracking algorithms start from object initialization in which the target is manually specified using a bounding box or ellipse. Motion estimation is the key phase in tracking. After the object has been modeled, particle filters [8] can be used to estimate object motion. There are two kinds of object modeling approaches: global object representations and local object representations. A variety of global visual representation methods are used for object tracking. Santner et al. [9] adapted an optical-flow-based representation and built a tracker using a single template model. Hedayati et al. [10] combined optical flow with mean shift of color signature to track multiple objects. Optical flow can provide spatiotemporal features of an object, but it can not be applied to scenes with rapid changes in illumination. Zhao et al. [11] represented objects by color distribution. A differential earth mover's distance algorithm was used to calculate the distance between two distributions. Sun et al. [12] used fragment-based features, and handled occlusion by solving a two-stage optimization problem. Hu et al. [13] proposed an active contour-based visual tracking framework. Colors, shapes, and motions are combined to evolve the contour. Jepson et al. [14] used object representations based on filter responses from a steerable pyramid. Other than traditional methods, neural networks can be used to perform object tracking without depending on extracting hand-crafted features. Wang et al. [15] proposed an online training network to transfer pre-trained deep features for tracking.

In contrast to global visual representations, local visual representations based on local appearance structures can be more robust to object deformation and illumination changes. Wang et al. [16] segmented superpixel regions surrounding the target, and then represented each superpixel by a feature vector. An appearance model based on superpixels was used to distinguish the object from its background. The scale-invariant feature transform (SIFT) [17] is a widely used local feature extraction algorithm; some approaches [18–20] use it to match regions of interest between frames in a tracking framework. Static and motion saliency features [21, 22] and corner features [23] have also been commonly used in object tracking. However, local representations reply on rich texture, and they are unstable for low resolution images. Simple motion estimation tracking suffers

from error accumulation and cannot deal with object occlusion or re-entry. Thus, combining tracking with detection is meaningful.

2.2 Tracking with detection

Some work has applied object detection to tracking systems, and these approaches are most similar to the approach we take. In Ref. [24], the identify of the tracked object was verified by a validator. If verification failed, an offline-trained object detector searched the entire image exhaustively. Li et al. [25] used a probabilistic model combining conventional tracking and object detection to track objects in low frame rate (LFR) video; a cascade particle filter was used. Okuma et al. [26] focused on tracking multiple objects which can leave and enter the scene, using a combination of mixture particle filters and Adaboost. However, there is no discrimination between the objects tracked. Pedestrian detectors can also be used to improve robustness in multi-object tracking [27]. All of the detectors used in the above papers were trained offline. Although offline-trained classifiers may perform better than real-time detectors due to ample training samples and sufficient training time, they cannot distinguish between objects of the same category. For example, a detection mechanism can localize all pedestrians in a frame, but it is unable to distinguish a specific person. Thus, it is hard for the detectors in the above papers to rectify the tracker following a specific object.

Grabner and Bischof [28] used a real-time Adaboost feature selection model for object detection. This work reduced the computational complexity of Adaboost significantly, but because of the limitations of the number of weak classifiers, the accuracy of the detector was low. Babenko et al. [29] trained an online object classifier which was updated by the output of the tracker. A multiple-instance-learning approach was used to reduce ambiguities in the detection phase. Tang et al. [30] treated tracking as a foreground–background classification problem; online support vector machines were built to recognise different features using a co-training framework. Online detectors are more adaptable, and are able to track a specific target amongst many objects from the same class. However, these classifiers perform worse than offline detectors in terms of accuracy, and training data extracted from real-time video have limited reliability. In this paper, we integrate a pre-trained model with a classifier which is updated in real time, to overcome this problem.

3 Tracking by detection and selection

Our framework has two phases: tracking and trajectory completion. The former can track the target even in the presence of frequent and long-term occlusion, or object absences, while the latter can complete incomplete trajectories having missing segments. The tracking part of our *tracking-by-detection using selection and completion* (TDSC) framework can track a specific target in videos with long-term occlusion. The user should label the object to be tracked in the first frame, then our tracking algorithm produces the location of this target in every frame. If the target is occluded or goes out of sight, this algorithm outputs the location where the target last appeared. After the target reappears, this tracking algorithm can find the target and output its correct location. The TDSC framework is able to distinguish a specific object amongst others of the same kind, for example, a specific pedestrian among many people. So, even though TDSC is designed to track a single object, we can also use it to deal with the multiple object tracking problem by running multiple simultaneous instances.

A block diagram for our framework is shown in Fig. 1. In this section, we consider the tracking phase, including the object detector, object selector, and tracker. The following section will consider trajectory completion.

Both detector and tracker receive video frames as input data. The object detector can localize objects in the same object category as the target being tracked. However, as a classifier, the detector has two main shortcomings: it is inevitable that (i) it will at times return false positives, and (ii) it will fail to discriminate objects of the same kind. Thus, any objects detected are next filtered by the object selector to remove false positives and objects other than the specific desired target.

Our work aims to build a robust framework for tracking objects with long-term occlusions. This paper does not focus on the design of the tracker. We thus simply use *compressive tracking* [6] in our implementation; it is often employed as a

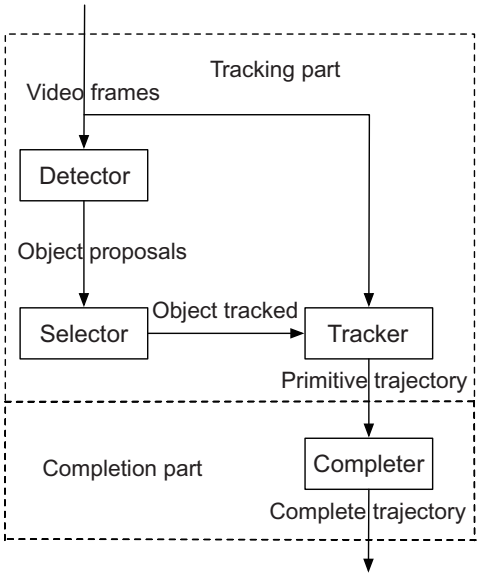

Fig. 1 Data flow between the components of TDSC.

benchmark in comparative experiments because of its effectiveness and efficiency.

A detector can produce coordinates and categories of objects. Object detection is a fundamental problem in computer vision. To obtain high accuracy, our framework uses an offline-trained detector which has been exposed to abundant training samples, without restriction on training time. In recent years, convolutional neural networks (CNNs) have become widely used in this field, as they have higher detection performance compared to methods based on low-level features such as histograms of oriented gradients (HoG) [31] or SIFT features [32]. In this paper, we employ *faster-RCNN* [33] as our object detector. Region proposal computation is a bottleneck for fast-RCNN [34]. Faster-RCNN overcomes this problem by using a region proposal network which shares convolutional features with the detection network. It achieves near real-time detection rates and detects multiple objects in specific classes with high accuracy.

The approach used by our object selector is to extract feature vectors from objects found by the detector, which we call object proposals, and use a categorization model to find positive proposals. If an object proposal is recognized as of the same category as that of the object being tracked, we call it a positive proposal. The feature vector is based on the color and shape of the object. A color histogram represents the distribution of colors in an object. We use HoG features to represent shape and contextual information. The first step in calculating the HoG descriptor is to compute image gradient values. The region of interest containing an object proposal is divided into 10×10 cells. Each pixel within a cell casts a weighted vote for an orientation-based histogram channel based on the magnitude and orientation of the gradient vector. To counter any changes in illumination over space, cells are grouped into blocks in which we locally normalize the gradient strengths. The HoG descriptor is then extracted by concatenating the normalized cell histograms from all blocks. For every block of an object proposal, the feature vector is calculated by combining the color histogram and the HoG descriptor into a vector $\boldsymbol{x}$. This is used by a classifier to assign a label y to each object, which is either $+1$ or -1 to state that it belongs to the target object and some other object respectively.

For simplicity and speed, we use a linear support vector machine (SVM) as the classifier. An SVM provides a method to calculate the hyperplane that optimally separates two high-dimensional classes of objects. The hyperplane is given by

$$\boldsymbol{\omega} \cdot \boldsymbol{x} + b = 0 \tag{1}$$

where $\boldsymbol{\omega}$ is the normal vector to the hyperplane and b is the hyperplane offset from the origin. Finding this hyperplane is a convex optimization problem. To be able to handle data which are not linearly separable, we introduce a soft margin, whereupon the objective function to be minimized is

$$\frac{1}{n}\sum_{i=1}^{n}\max(0, 1 - y_i(\boldsymbol{\omega} \cdot \boldsymbol{x} + b)) + \lambda\|\boldsymbol{\omega}\|^2 \tag{2}$$

Doing so gives a hyperplane which can be used for classifying the feature vectors. However, this does not yet take into account temporal coherence.

In the case of continuous successful detection, the current target location should be close to the previous one. If an object proposal is far from the correct target in the previous frame, this object is unlikely to be the correct proposal: distance should also be considered in our object selector. However, if the object has been absent for a while, the object is likely to be further away.

In a standard SVM, a new feature vector $\boldsymbol{x}$ is classified by computing:

$$\text{sign}(\boldsymbol{\omega} \cdot \boldsymbol{x} + b) \tag{3}$$

Taking distance as a penalty factor, given an absence of detection output for T frames, the classification

formula can now be written as

$$\text{sign}(\boldsymbol{\omega} \cdot \boldsymbol{x} + b + a_0 d^2 + a_1 d - \mu) \tag{4}$$

where

$$d = \sqrt{(x_\text{c} - x_\text{p})^2 + (y_\text{c} - y_\text{p})^2} \tag{5}$$

is the distance between the current object proposal location (x_c, y_c) and the previous target location (x_p, y_p), and constant μ is a distance threshold set by experimental experience. We can see from Eq. (4) that we use a quadratic form of distance as the penalty factor. If the current object proposal is far from the previous object, it is highly unlikely for this proposal to be the correct location. As the distance increases, the penalty factor should be dominant, so the penalty factor is made to be a quadratic function of distance.

In reality, the appearance of the object can change during the tracking process. For example, a pedestrian may slowly turn around. Although only the initial bounding box that the user draws is completely reliable, we cannot train our selection only using the initial data because of this appearance change. To provide greater adaptability, the object selector is trained online. In the initial stage, we should draw a rectangle to specify an object to track, and draw another rectangle as a negative sample, to initialize the SVM. Once the initial SVM model has been established, positive and negative samples are extracted in the process of selection. Afterwards, online training is carried out continuously in order to adapt to the changes in appearance of the target.

4 Trajectory completion

When the object being tracked is occluded, the tracker cannot produce correct coordinates. A sudden, significant change in object coordinates is inevitable when the object becomes visible again after occlusion. We can determine that the object has been occluded by detecting abrupt position change. This framework does not detect occlusion directly by the tracker, because the tracker should not determine that the object is occluded when it is not distinct and hard to recognize.

When occlusion has been detected, a trajectory completion mechanism is used to correct the discontinuous trajectories. Trajectory completion is a temporal extrapolation problem. Artificial neural networks are one of the most accurate and widely used forecasting models which are capable of identifying complex nonlinear relationships between input and output data. Non-linear autoregressive (NAR) neural networks have proved useful for complicated pattern data forecasting [35]. Completing the tracked object's coordinates is also a data forecasting problem. We thus employ a three-layer NAR neural network for trajectory completion. Compared with using an interpolation method such as spline fitting, an NAR neural network can produce a *predicted* trajectory without making any assumption about the type of movement the object is undergoing, and need not assume a smooth trajectory.

An NAR network has linear activation functions for the output layer and non-linear logistic activation functions for the hidden layer. Thus our network performs a non-linear functional mapping from the past object coordinates to future locations.

Let x_t be the horizontal coordinate of the target at time t. The mapping performed by the network is

$$x_t = f(x_{t-1}, \ldots, x_{t-l}, \boldsymbol{w}) \tag{6}$$

where $\boldsymbol{w}$ are the connection weights of this network and l is the maximum time delay for the input data.

We can see that this network is a non-linear autoregressive model. The structure of this autoregressive neural network is shown in Fig. 2.

In the majority of cases, trajectories of objects are continuous and smooth, such as when tracking cars and pedestrians. In order to make full use of known information and improve the continuity and smoothness of the trajectories, the forecast is carried out in two directions. The coordinates *before*

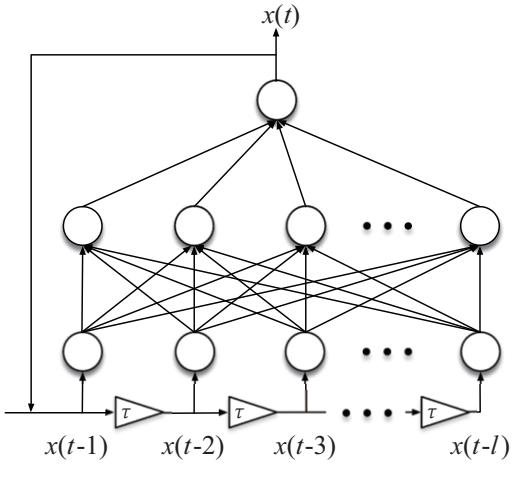

Fig. 2 Autoregressive neural network structure.

occlusion are input to the NAR neural network as training data in time sequence, and the prediction output is denoted by $\{x_t^+\}$, while the coordinates *after* occlusion are also input to the neural network as training data in the reverse direction, and the prediction output is denoted by $\{x_t^-\}$. The final forecast result $\{x_t\}$ can be calculated using:

$$x_i = \frac{t_e - t}{t_s - t_e} x_t^- + \frac{t - t_s}{t_s - t_e} x_t^+ \tag{7}$$

5 Experimental results

5.1 Dataset preparation

Several datasets exist for benchmarking visual tracking, such as VOT[①], VTB[②], and MOT[③]. However, most sequences in these datasets have no object occlusion. Even in the video samples in which objects *are* blocked, the occlusion spans are too short. In order to evaluate tracking performance in sophisticated circumstances such as long and frequent occlusion, we introduce a more challenging dataset. Two sequences in this dataset are selected from VTB benchmark, and we have captured four more difficult samples with significant occlusion.

For quantitative evaluation, we use the following protocol similar to that used in the MOT benchmark. Tracking starts from an initial bounding box in the first frame. Both the ground truth and the tracking results are a sequence of bounding boxes. If the overlapping area between a tracking bounding box and a ground truth bounding box is larger than an overlap threshold, the tracking result in this frame is deemed successful. For every tracking framework, we plot a success rate curve against the overlap threshold. The overall performance of a tracker can be measured by the AUC (area under the curve) criterion. In order to measure the ability to handle occlusion, re-initialization is not performed in the tracking process.

5.2 Experiments on the tracking phase

We compare our proposed TDSC framework with four trackers: CT [6], KCF [36], CSK [37], IMT [38], SORT [39], and LCT [40]. All trackers were run with the same parameters using our dataset. Figure 3 shows tracking results for some test samples. From top to bottom these are: David, Jogging, Occlusion 1, Occlusion 2, Occlusion 3, and Frameskip. The first two are relatively simple scenes with short-term occlusion, and come from the VTB dataset. Occlusion 1 is a multi-object occlusion scene. Occlusion 2 includes long-term occlusion and shaking. Occlusion 3 is a long-term and multi-object occlusion scene. The last sample has missing frames in a sequence. The results reveal that while state-of-the-art tracking methods are able to handle short-term occlusion, the targets are lost with long-term occlusion. However, our TDSC framework can continue tracking through re-initialization.

Figure 4 presents the performance curves for these seven trackers on our dataset. The results indicate that the proposed tracker has better performance than the other trackers in our experiments. To provide a quantitative analysis, we give the AUC values for these seven trackers using our test dataset, for three specific conditions, in Table 1. The proposed TDSC framework shows a significant improvement over the prior state-of-the-art methods.

The proposed framework achieves real-time processing. For a resolution of 576×432, traditional tracking and selection only take 3.9 ms and less than 1 ms respectively using an Intel Core i7 CPU.

5.3 Experiments on the completion phase

Building datasets is the first difficulty in trajectory completion experiments. For a video sample in which an object is physically occluded, we are unable to acquire a complete trajectory, so it is hard to annotate ground truth. Therefore, we capture some video samples without occlusion, annotate the object trajectories as ground truth, and then draw synthetic obstacles to occlude the moving objects.

We conducted experiments using two kinds of two cases: straight trajectories and curved trajectories. Results are illustrated in Fig. 5. Blue lines are trajectories extracted by our tracking–detection–selection mechanism. Because of occlusion, these trajectories are discontinuous. Yellow lines are trajectories predicted by the completion mechanism.

Table 1 Tracking performance measured by the AUC criterion

	TDSC	KCF	CSK	CT	IMT	SORT	LCT
Frameskip	0.904	0.154	0.059	0.199	0.206	0.067	0.132
Occlusion	0.867	0.023	0.043	0.186	0.090	0.075	0.241
Shaking	0.848	0.036	0.054	0.357	0.219	0.036	0.313

① http://www.votchallenge.net/vot2016/dataset.html
② http://cvlab.hanyang.ac.kr/tracker_benchmark/index.html
③ http://motchallenge.net/data/MOT16/

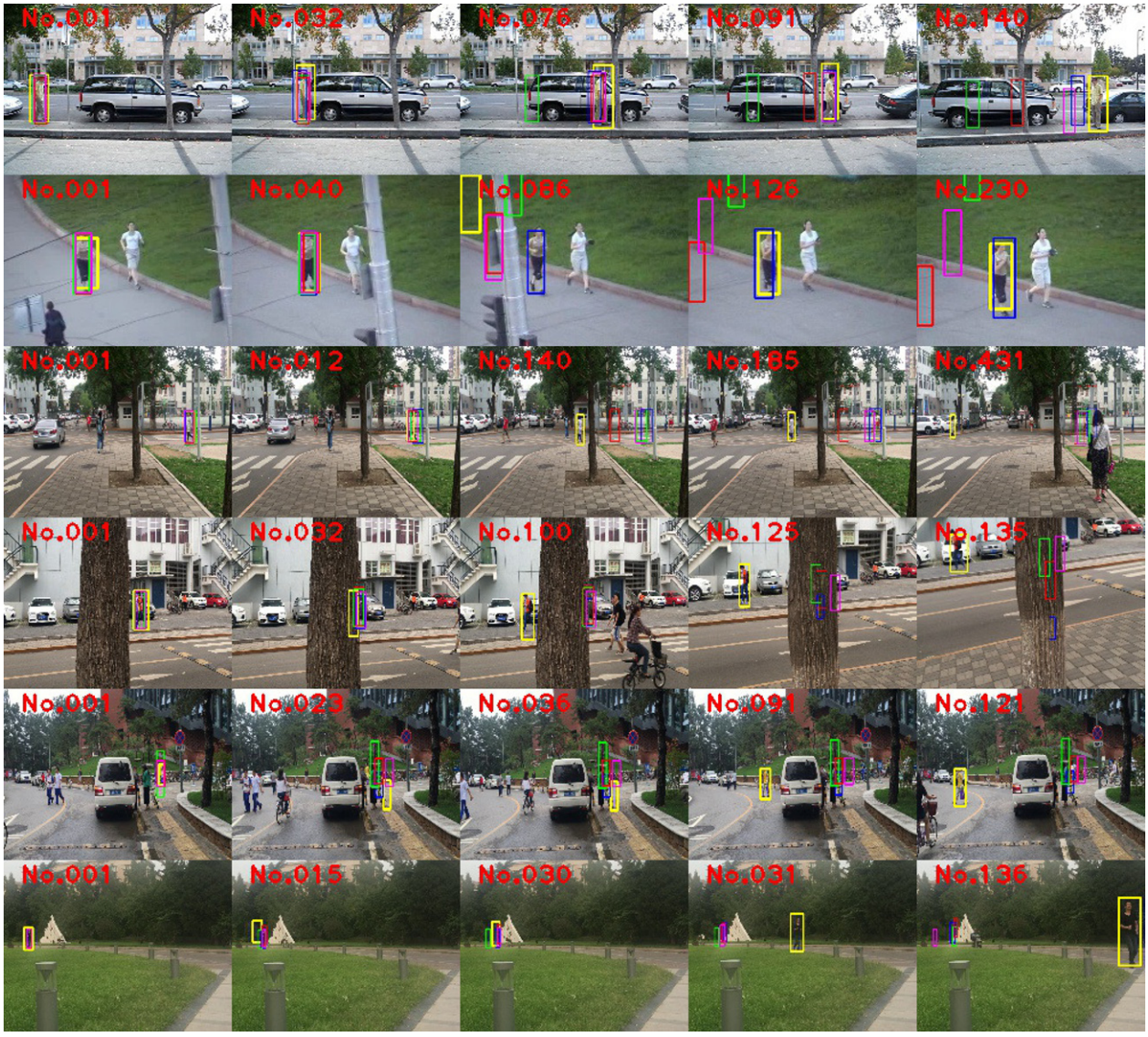

Fig. 3 Tracking results. Samples, top to bottom: David, Jogging, Occlusion 1 (multiple objects), Occlusion 2 (long-term occlusion and shaking), Occlusion 3 (long-term occlusion), and Frameskip. Yellow, blue, green, red, and purple rectangles represent tracking output of TDSC, KCF, CSK, CT, and IMT respectively.

Red crosses indicate trajectory ground truth. We use the average distance between points in predicted trajectories and ground truth trajectories for quantitative evaluation. The average distance in straight-trajectory cases is 5 pixels (3% of the target height) and in curved trajectory cases, 10 pixels (14% of the target height). Our experiments furthermore show that our TDSC framework can output continuous trajectories in cases with occlusion.

6 Conclusions

In this paper, we designed a novel framework to solve the problem of object tracking where long-term occlusion interferes with the tracking process. Continuous tracking is necessary for some realistic difficult problems, especially for safety monitoring. Our framework decomposes the task into two parts: tracking and trajectory completion. The object detector is a deep neural network model which localizes objects in the same category as the object being tracked. The object selector is based on an online-trained SVM model, and discriminates between the outputs of the object detector to determine which object should be used to re-initialize the tracker. Offline-trained and online-

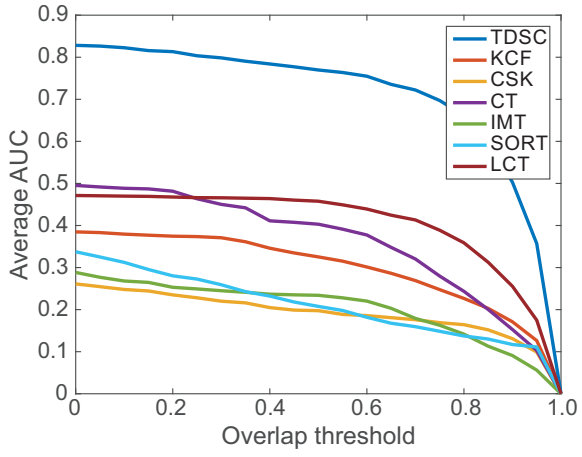

Fig. 4 Overall performance curve.

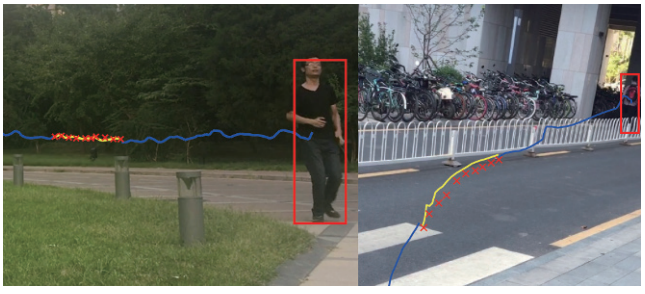

Fig. 5 Trajectory completion results.

trained classifiers are combined for accuracy and flexibility. To obtain a continuous trajectory, we utilize a non-linear autoregressive neural network to complete the missing parts of trajectories extracted by the tracking component of TDSC. Quantitative experiments show our proposed framework improves upon prior state-of-the-art tracking methods and is able to output continuous trajectories.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Project No. 61521002), the General Financial Grant from the China Postdoctoral Science Foundation (Grant No. 2015M580100), a Research Grant of Beijing Higher Institution Engineering Research Center, and an EPSRC Travel Grant.

References

[1] Collins, R. T.; Lipton, A. J.; Fujiyoshi, H.; Kanade, T. Algorithms for cooperative multisensor surveillance. *Proceedings of the IEEE* Vol. 89, No. 10, 1456–1477, 2001.

[2] Greiffenhagen, M.; Comaniciu, D.; Niemann, H.; Ramesh, V. Design, analysis, and engineering of video monitoring systems: An approach and a case study. *Proceedings of the IEEE* Vol. 89, No. 10, 1498–1517, 2001.

[3] Kanhere, N. K.; Birchfield, S. T.; Sarasua, W. A. Vision based real time traffic monitoring. U.S. Patent 8,379,926. 2013.

[4] Morris, B. T.; Tran, C.; Scora, G.; Trivedi, M. M.; Barth, M. J. Real-time video-based traffic measurement and visualization system for energy/emissions. *IEEE Transactions on Intelligent Transportation Systems* Vol. 13, No. 4, 1667–1678, 2012.

[5] Rui, Y.; Chen, Y. Better proposal distributions: Object tracking using unscented particle filter. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, II-786–II-793, 2001.

[6] Zhang, K.; Zhang, L.; Yang, M.-H. Real-time compressive tracking. In: *Computer Vision–ECCV 2012*. Fitzgibbon, A.; Lazebnik, S.; Perona, P.; Sato, Y.; Schmid, C. Eds. Springer-Verlag Berlin Heidelberg, 864–877, 2012.

[7] Li, X.; Hu, W.; Shen, C.; Zhang, Z.; Dick, A.; van den Hengel, A. A survey of appearance models in visual object tracking. *ACM Transactions on Intelligent Systems and Technology* Vol. 4, No. 4, Article No. 58, 2013.

[8] Isard, M.; Blak, A. CONDENSATION—Conditional density propagation for visual tracking. *International Journal of Computer Vision* Vol. 29, No. 1, 5–28, 1998.

[9] Santner, J.; Leistner, C.; Saffari, A.; Pock, T.; Bischof, H. PROST: Parallel robust online simple tracking. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 723–730, 2010.

[10] Hedayati, M.; Cree, M. J.; Scott, J. Combination of mean shift of colour signature and optical flow for tracking during foreground and background occlusion. In: *Image and Video Technology*. Bräunl, T.; McCane, B.; Rivera, M.; Yu, X. Eds. Springer International Publishing Switzerland, 87–98, 2016.

[11] Zhao, Q.; Yang, Z.; Tao, H. Differential earth mover's distance with its applications to visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 32, No. 2, 274–287, 2010.

[12] Sun, C.; Wang, D.; Lu, H. Occlusion-aware fragment-based tracking with spatial-temporal consistency. *IEEE Transactions on Image Processing* Vol. 25, No. 8, 3814–3825, 2016.

[13] Hu, W.; Zhou, X.; Li, W.; Luo, W.; Zhang, X.; Maybank, S. Active contour-based visual tracking by integrating colors, shapes, and motions. *IEEE Transactions on Image Processing* Vol. 22, No. 5, 1778–1792, 2013.

[14] Jepson, A. D.; Fleet, D. J.; El-Maraghi, T. F. Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 25, No. 10, 1296–1311, 2003.

[15] Wang, L.; Ouyang, W.; Wang, X.; Lu, H. STCT: Sequentially training convolutional networks for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1373–1381, 2016.

[16] Wang, S.; Lu, H.; Yang, F.; Yang, M.-H. Superpixel tracking. In: Proceedings of the IEEE International Conference on Computer Vision, 1323–1330, 2011.

[17] Lowe, D. G. Object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE International Conference on Computer Vision, Vol. 2, 1150–1157, 1999.

[18] Chen, A.-h.; Zhu, M.; Wang, Y.-h.; Xue, C. Mean shift tracking combining SIFT. In: Proceedings of the 9th International Conference on Signal Processing, 1532–1535, 2008.

[19] Fazli, S.; Pour, H. M.; Bouzari, H. Particle filter based object tracking with sift and color feature. In: Proceedings of the 2nd International Conference on Machine Vision, 89–93, 2009.

[20] Zhou, H.; Yuan, Y.; Shi, C. Object tracking using SIFT features and mean shift. *Computer Vision and Image Understanding* Vol. 113, No. 3, 345–352, 2009.

[21] Mahapatra, D.; Saini, M. K.; Sun, Y. Illumination invariant tracking in office environments using neurobiology-saliency based particle filter. In: Proceedings of the IEEE International Conference on Multimedia and Expo, 953–956, 2008.

[22] Zhang, G.; Yuan, Z.; Zheng, N.; Sheng, X.; Liu, T. Visual saliency based object tracking. In: *Computer Vision–ACCV 2009*. Zha, H.; Taniguchi, R.; Maybank, S. Eds. Springer-Verlag Berlin Heidelberg, 193–203, 2010.

[23] Kim, Z. W. Real time object tracking based on dynamic feature grouping with background subtraction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1–8, 2008.

[24] Williams, O.; Blake, A.; Cipolla, R. Sparse Bayesian learning for efficient visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 27, No. 8, 1292–1304, 2005.

[25] Li, Y.; Ai, H.; Yamashita, T.; Lao, S.; Kawade, M. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 30, No. 10, 1728–1740, 2008.

[26] Okuma, K.; Taleghani, A.; de Freitas, N.; Little, J. J.; Lowe, D. G. A boosted particle filter: Multitarget detection and tracking. In: *Computer Vision–ECCV 2004*. Pajdla, T.; Matas J. Eds. Springer-Verlag Berlin Heidelberg, 28–39, 2004.

[27] Leibe, B.; Schindler, K.; van Gool, L. Coupled detection and trajectory estimation for multi-object tracking. In: Proceedings of the IEEE 11th International Conference on Computer Vision, 1–8, 2007.

[28] Grabner, H.; Bischof, H. On-line boosting and vision. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 260–267, 2006.

[29] Babenko, B.; Yang, M.-H.; Belongie, S. Visual tracking with online multiple instance learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 983–990, 2009.

[30] Tang, F.; Brennan, S.; Zhao, Q.; Tao, H. Co-tracking using semi-supervised support vector machines. In: Proceedings of the IEEE 11th International Conference on Computer Vision, 1–8, 2007.

[31] Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 886–893, 2005.

[32] Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 38, No. 1, 142–158, 2016.

[33] Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. In: Proceedings of the Advances in Neural Information Processing Systems 28, 91–99, 2015.

[34] Girshick, R. Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, 1440–1448, 2015.

[35] Chow, T. W. S.; Leung, C. T. Nonlinear autoregressive integrated neural network model for short-term load forecasting. *IEE Proceedings-Generation, Transmission and Distribution* Vol. 143, No. 5, 500–506, 1996.

[36] Henriques, J. F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 37, No. 3, 583–596, 2015.

[37] Henriques, J. F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In: *Computer Vision–ECCV 2012*. Fitzgibbon, A.; Lazebnik, S.; Perona, P.; Sato, Y.; Schmid, C. Eds. Springer-Verlag Berlin Heidelberg, 702–715, 2012.

[38] Yoon, J. H.; Yang, M. H.; Yoon, K. J. Interacting multiview tracker. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 38, No. 5, 903–917, 2016.

[39] Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and realtime tracking. In: Proceedings

of the IEEE International Conference on Image Processing, 3464–3468, 2016.

[40] Ma, C.; Yang, X.; Zhang, C.; Yang, M.-H. Long-term correlation tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 5388–5396, 2015.

Ruochen Fan is a master candidate in the Department of Computer Science and Technology, Tsinghua University. He received his bachelor degree from Beijing University of Posts and Telecommunications in 2016. His research interest is computer vision.

Fang-Lue Zhang is a lecturer in Victoria University of Wellington. He received his doctor degree from Tsinghua University in 2015 and bachelor degree from Zhejiang University in 2009. His research interests include image and video editing, computer vision, and computer graphics.

Min Zhang is a postdoctoral fellow in the Center of Mathematical Sciences and Applications, Harvard University. She received her Ph.D. degree in computer science from Stony Brook University and another Ph.D. degree in mathematics from Zhejiang University. She is an expert in the fields of geometric modeling, medical imaging, graphics, visualization, machine learning, 3D technologies, etc.

Ralph R. Martin is a professor in Cardiff University. He obtained his Ph.D. degree from Cambridge University in 1983. He has published more than 300 papers and 15 books, covering such topics as solid and surface modeling, intelligent sketch input, geometric reasoning, reverse engineering, and computer graphics. He is a Fellow of the Learned Society of Wales, the Institute of Mathematics and its Applications, and the British Computer Society. He has served on the editorial boards of *Computer-Aided Design*, *Computer Aided Geometric Design*, and *Geometric Models*. He was recently awarded a Friendship Award, China's highest honor for foreigners.