Journal of Computers Vol. 31 No. 4, 2020, pp. 77-90
doi:10.3966/199115992020083104007

Multi-Update Patterns and Validity Verification for Robust Visual Tracking

Jia Wen[1,2,3], Hong-Jun Wang[1,2,3*], Yan-Nan Li[1,2,3], Ji-Wei Zhao[1,2,3],Yu-Ying Yang[1,2,3]

[1] School of Information Science and Engineering, Yanshan University, Qinhuangdao, China

[2] The Key Laboratory for Computer Virtual Technology and System Integration of Hebei, Province

[3] The Key Laboratory of Software Engineering of Hebei Province
wjyanyuan@aliyun.com, 2642264392@qq.com

Received 2 January 2019; Revised 12 July 2019; Accepted 28 August 2019

Abstract. The Object tracking is a challenging problem in computer vision field. Now, deep learning has made outstanding achievements in feature extraction. There are already some examples of deep learning applications in visual tracking. But, they almost paid attention to precision and sacrificed efficiency. So we propose a tracker named adaptive combination tracker(ACT), based on neural network model, which gives consideration to both precision and efficiency. Our ACT tracker gets rich features offline through Stacked Denoising Autoencoders (SDAE) and transfers them to online tracking. In tracking, a dynamic particle filter algorithm is proposed to speed up tracking. We update our tracker with different patterns according to the change in appearance. In order to ensure efficiency, our tracker regulates speed adaptively. Our tracker regulates speed precisely to avoid drift caused by fast motion as much as possible, and confirm and optimize the effectiveness of the results by comparison with previous target. We evaluate our tracker on OTB2013 datasets. Extensive experiments on OTB2013 datasets demonstrate that the proposed tracker performs favorably in relation to the state-of-the-art methods.

Keywords: different update patterns, dynamic particle filter, neural network, object tracking

1 Introduction

Visual object tracking is an important topic in computer vision and plays a necessary role in numerous applications, such as video surveillance, automobile navigation, human-computer interfaces, robotics and driverless vehicle. Although substantial progress has been made in recent years, achieving Higher accuracy and speed in visual object tracking is still a thorny problem.

Many different methods have been proposed for visual object tracking in succession in recent decades [1-5]. Bertinetto et al. [6] proposed a visual target tracking method based on Siamese network model. Although this algorithm has a fast tracking speed, the tracking accuracy is not high because the target model has no update process. Wang et al. proposed a tracker named DLT [7], although there is a process of model updating during tracking, it is difficult to avoid drift and correct, when the appearance changes. For the DLT tracker, the Back Propagation algorithm (BP) tends to gradient diffusion due to the depth of the model update. The diffusion of the gradient hinders the updating of the lower layers.

To solve these problems, we use SDAE at a lower level and update the lower layers independently. It prevents the spread of gradients and improves the adaptability of the tracker. Because the appearance changes to varying degrees, the adjustment strategy should have different modes. We combine SDAE and BP to update the trackers in online tracking and use different modes in different situations. In order to achieve adaptive tracking, we use three modes.

* Corresponding Author

We use adaptive particle filte tracking in online tracking. Although trackers can accelerate tracking by reducing the number of particles, most trackers will affect the accuracy. Therefore, if the accuracy is reduced, our method will recover the number of particles and search for the target again from all particles. When the accuracy is lowered, drift due to particle reduction can be prevented by recycling the particles. The parameters of the target are determined by the particles with the highest confidence, but it increases instability. The risk of drift is reduced by a strategy of selecting from multiple candidates as compared to simply determining a target by a single particle. In our tracker, the target is determined from ten candidates and a predetermined target. In this way, the instability due to individual particles can be eliminated.

In this paper, we propose an adaptive combined tracker (ACT). Our tracker uses a large number of different types of samples for offline training. In tracking, our tracker uses different modes in different situations, applying adaptive particle filters to our model, and adjusting the number of particles according to the accuracy of the tracker. We use Denoising Autoencoders (DAE) to optimize the target. Our ACT tracker is evaluated with a challenge benchmark [8] with 50 sequences. The experiment results show that our tracker has good tracking accuracy and tracking speed, and has high tracking performance.

In this paper, a motion tracking algorithm based on the auxiliary update model and the validity check is proposed. First, in order to prevent overfitting, a large number of pictures of different objects are used as the material for offline training. In the tracking process, the tracking model uses an auxiliary update model to help the apparent model adapt to the appearance change. The tracker uses dynamic particle filtering. The number of particles is varied according to the adaptation of the tracker. In the process of target validation, this paper proposes an algorithm to optimize the target using noise reduction. This paper evaluates all the algorithms through a set of evaluation datasets with 50 challenging video sequences. The experimental results show that the algorithm of this thesis is better than the other excellent algorithms, and the learning algorithm is improved at the speed.

2 Related Work

Object tracking is a hot topic in computer vision. Many very good trackers such as [9, 10] appear every year. The current mainstream tracking data sets has OTB2013 [8]. OTB2013 contains 50 sequences and annotated sequences with 9 attributes. They represent a challenging aspect of visual tracking. Here are some excellent trackers in recent years.

Wang et al. proposed a tracker DLT [7] based on depth learning and its improved SO-DLT [11]. DLT learns generic image features with SDAE offline training and transfers knowledge from offline training to online tracking process. SO-DLT is the improvement of DLT. It explicitly adopted a structured nature in the model. FCNT [12] used a pretrained convolutional neural network (CNN) to track targets. It believed that only some neurons are related, and develops a feature map selection method to remove noise and uncorrelated feature maps. CNN-SVM [13] makes use CNN and Support Vector Machin (SVM) to construct target-specific saliency map. Saliency map visualized spatial configuration of target effectively and enable CNN-SVM to achieve pixel-level target segmentation.

DSST [14] is one of the correlation filtering methods. It added more scale regression to achieve accurate scale estimation. MEEM [15] learned a discriminative tracker and makes use of a set of snapshots as expert to predict target. TGPR [16] analyzed the probability of target appearance by using Gaussian Processes Regression. The tracking results are determined by fusing decisions from two individual trackers. These ingredients together enable TGPR to alleviate the drifting issue from all various aspects. KCF [17] proposed an analytic model for datasets of thousands of translated patches. For kernel regression, they derive a new kernelized correlation filter, which unlike other kernel algorithms has the exact same complexity as its linear counterpart. This algorithm is further improved by adding the HOG feature in the KCF algorithm. João et al. proposed to use correlation filters in a kernel space with the CSK [18], which achieves the highest efficiency in OTB2013 [8].CSK algorithm based on light intensity. CN [19] proposed an adaptive low-dimensional variant of color attributes. It extends the CSK tracker with color attributes. Struck [20] presented a framework for adaptive visual object tracking based on structured output prediction. It has been highlighted in several recent studies [21-22]. By explicitly allowing the output space to express the needs of the tracker, they are able to avoid the need for an intermediate classification step. Struck uses a kernelized structured output SVM, which is learned online to provide adaptive tracking. In contrast to the generative methods, discriminative methods [23-25] consider both foreground and background information. Nam et al. [26] proposed a method for target tracking of single objects by extracting CNN features with multiple domains and more generics. The method proposed by Pu et al. [27] introduces attention mechanism to target tracking.

Our tracker is different with the other trackers in the following three aspects. Firstly, multiple patterns are used to adapt to different situations. Besides, the number of particles is flexible, which is decreased in stable situations and recovered before the precision decreases. Finally, the validity of results is confirmed by verifying candidates.

3 The Proposed Method

3.1 Offline Training

We use SDAE in offline training. SDAE is consists of DAE. The output of lower DAE forms the input of upper DAE. DAE can recover original data from corrupted input. We compel network to extract features by reducing the number of units layer by layer. The size of input is 32*32. It is the same as images of dataset. The units of top layer are 256.

In DAE, let x_i denote the original data, W and W' denote the weights of encoder and decoder, b and b' denote basis terms, z_i is the activation of the second hidden layer. DAE can extract features from images by adjusting parameters of optimization problem. The formula for the optimization problem is as follow:

$$\min_{W,W,b,b'} \sum_{i=1}^{k} \| x_i - z_i \|_2^2 + \lambda(\| W \|_F^2 + \| W' \|_F^2). \quad \textbf{(1)}$$

Here λ is the parameter of weight penalty term which balances the reconstruction loss and weight penalty terms, $\| \cdot \|_F$ denotes the L2 norm (Frobenius norm).

In order to extract more effective features, sparsity constraint is added to DAE. The sparsity constraint restricts the units to be inactive at most of time. Let ρ denote the target sparsity level, ρ usually is a number which close to 0. Let $\hat{\rho}_j$ denote average activation of hidden units *j*. So, sparsity constraint can be described as follow:

$$KL(\hat{\rho} \| \rho) = \sum_{j=1}^{m} [\rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}_j}]. \quad \textbf{(2)}$$

Due to ρ is a small value, the sparsity constraint can restrict the average activation. We train network with unsupervised learning. As too many samples, the network is trained in batches. For training sufficiently, the batch size is set to 100. The training is terminated in advance when the training is achieved or the error of reconstruct rises. The process of offline training is shown in Fig. 1.

3.2 Online Tracking

In benchmark, target is fixed in the first frame of every sequence. In the initial stages of online tracking, positive samples are collected by translating several pixels from target in the first frame. Negative samples are necessary for preventing drift. The negative samples can be collected by translating and scaling targets. Our tracker is a discriminative tracker. Online tracking is a process of discriminating targets and environment. The offline training model needs to be converted into discriminative model. In offline training, the objective of SDAE extracts abundant features. In online tracking, the features extracted by offline training model are classified by a classifier. This process can be tuned by BP simplify. The whole network is updated by BP. This has no conflict with SDAE.

The variation of appearance is unavoidable in sequences. As usually, we adapt to the variation of appearance by updating observation model. Samples collection is executed during entire object tracking. Samples are the input for model updating. As the target in first frame is given, negative samples can be collected by affine transformation near the target. In the initial stages of online tracking, the other positive samples can be collected by translating several pixels. In later tracking, each frame replaces a positive sample. Negative samples are updated when the model needs to be updated. The process of

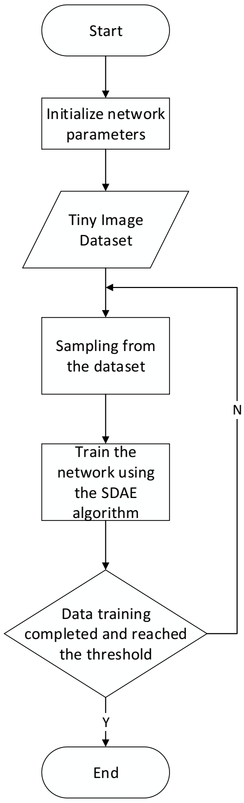


Fig. 1. Offline training flow chart

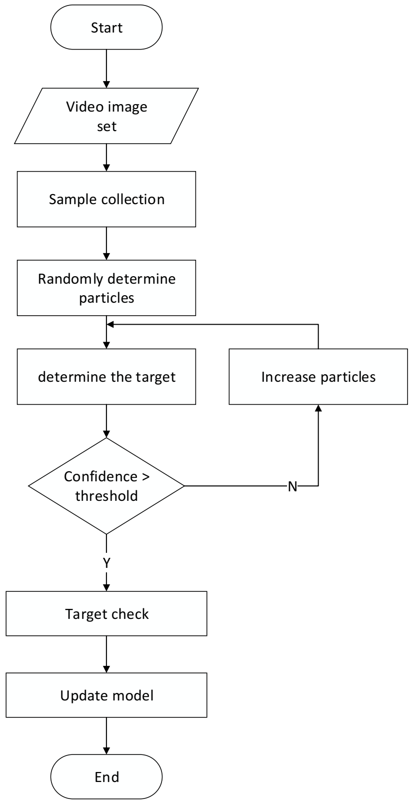


Fig. 2. Online tracking process flow chart

online tracking is shown in Fig. 2. For tracking, our tracker can be divided into apparent model and motion model. The apparent model mainly describes the appearance of the object, the motion model mainly locates the position of the target, and describes the state of the motion. The tracking process is shown in Fig. 3.

3.2.1 Observation Model

There are some drift issues because model can't adapt to changes in appearance. We take different patterns for ensuring the adaptation. We take Discriminant model and evaluate samples by confidence.

The model needn't update for a long time if the appearance variation is endurable. The variation of appearance is still existed. To prevent something unexpected, update is executed every 50 frames if the tracker behaves well. The update needn't spend too much time due to the slight variation. BP can be used to finetune whole network. When the network is updated by BP, the diffusion of gradients makes the update incomplete. The diffusion of gradients means that the error terms of deep BP network reduces layer by layer and the update of weights is relatively slow in lower layers. Considering these problems, we apply DAE in bottom layer and BP in the other layers. The combinative method is more effective compared with single BP when observation model needs to be updated on account of appearance variation. When variation is dramatical, traditional methods can't adapt to variation. To ensure precision, the model needs to be updated completely and the adaptability should be improved. Before combinative tuning, SDAE is applied to update whole network. For further update, BP can be used after combinative tuning as initialization. Considering computational complexity, the interval is at least 15 frames. The Update patterns is shown in Fig. 4.

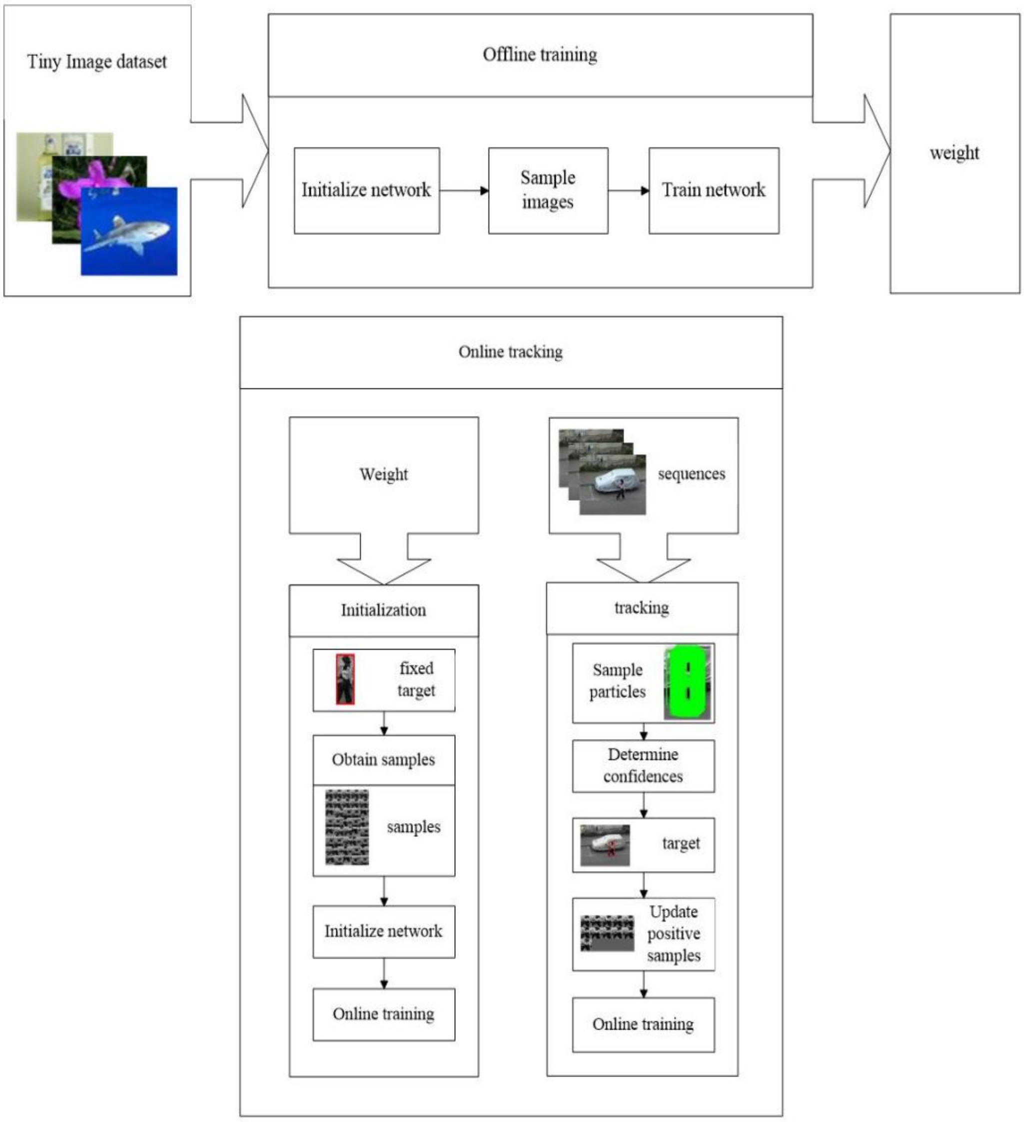


Fig. 3. The tracking process

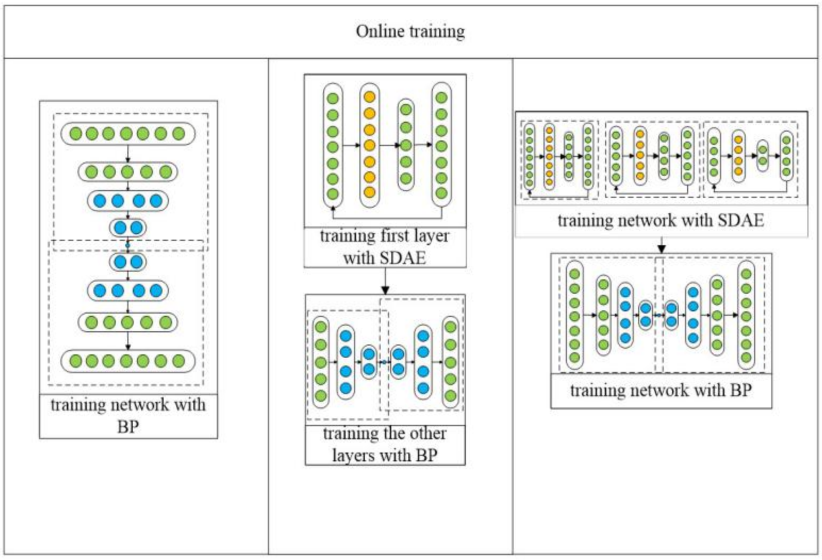


Fig. 4. Update patterns

3.2.2 Adaptive Dynamic Model

A dynamic model is used to simulate the states and state transitions of the target. Dynamic model and observation model are combined to determine target positions.

In our tracker, particle filter is used as dynamic model. Particle filter is non-linear and non-Gaussian assumption. It is easy to implement multi-object tracking. In particle filter tracker, a large number of particles are exploited to ensure precision. These particles are contributed to achieve better results in the frames which the appearance changes sharply. But in most situations, a large number of particles slow down tracking. Determining the number of particles adaptively can effectively improve tracking speed. We establish an adaptive method according to different situations.

Adaptive particle filter speeds up tracking by reducing particles. Regulating the number of particles simply according to the confidence of observation model is feasible in accelerating tracking. However, the rapidly varied appearance makes tracker drift easily for lacking of particles. For this reason, we propose a novel adaptive particle filter. The confidence reflects the state of model. Usually higher confidence makes more certain for the model. If confidence is sufficient, reducing particles properly accelerates tracking and takes less influence to precision. We reduce the number of particles according to confidence. The number of particles is divided to three levels. Reducing particles may affect precision if the confidence is less than 0.85. The number of particles is fixed at 1000. While confidence is between 0.85 to 0.9, tracker performs well. The number of particles can be reduced properly, therefore we set it to 500. If the confidence is greater than 0.9, the precision is excellent. The number of particles is fixed at 200. The parameters of particles are obtained before reducing particles. The parameters are determined randomly according to 1000 particles. Reducing particles is just reduced sampling from frames. Even if reducing particles only in proper precision, the drift problem is still happened due to lack of particles in rapidly variation. We can get confidence after determining targets in this situation. While the confidence is less than expectation, we can add particles by use of unutilized parameters. The targets which are determined by added particles and by original particles are compared to determine final targets. While 200 particles are used and the confidence is less than 0.9, 300 particles will be added. Then the target is determined from these 500 particles. 500 particles are added when we use 500 particles and confidence is less than 0.85. Regulating particles adaptively can ensure precision while accelerating tracking. The Adaptive particle filter is shown in Fig. 5.

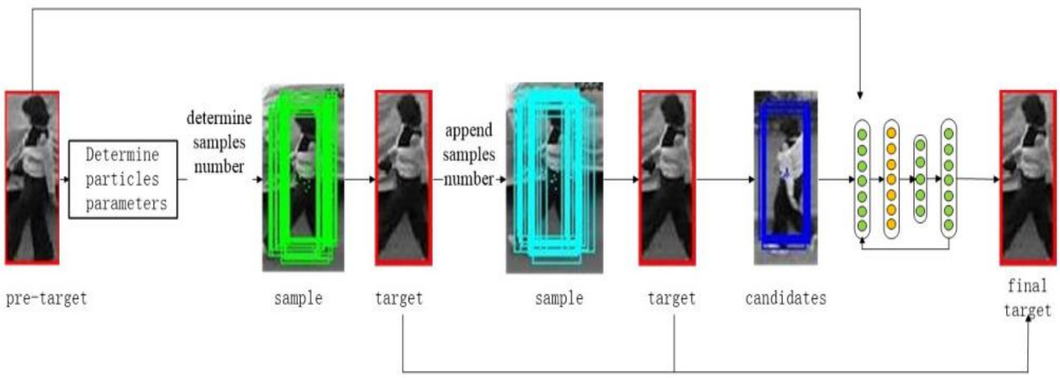


Fig. 5. Adaptive particle filter

3.2.3 Target Determination

In tracking, the targets in previous frames will influence the following tracking. The undesirable model update brings drift. The influence of drift is great for observation model in further tracking. The tracking is maintained by updating model online to adapt to appearance variation. The drift during appearance varied rapidly is extremely unfavorable for model update. Therefore, the determination of targets is important. The precision will be better if the targets are determined by candidates which are screened out from particles. The target of highest confidence maybe not the best one and the model distinguishes real target and similar surroundings occasionally.

We propose a novel method to determine target to prevent single target rising instability of tracker. The process of determining the target is shown in Fig. 6. We get temporary target by averaging the best 10 candidate targets and determine final target by these 10 targets and temporary target. When the confidence is less than 0.8, we compare selected targets and pre-target by DAE to reflect variation of appearance. In DAE, the corrupt input takes a forward pass through the network and compared with the original input to calculate loss. We replace input by temporary target and contrast with pre-target. If the loss is less than 0.1, the distinction between two targets is smaller, so the temporary target can be regarded as final target. If the loss is beyond the threshold, we compare the candidates and pre-target to select the best one. If the loss of the best one is beyond 0.4, we give up the verification and appoint temporary target as final target. So, the formula of target determination can be described as follow:

$$\min_i \frac{1}{2} \| a^p - a_i^o \|^2 . \tag{3}$$

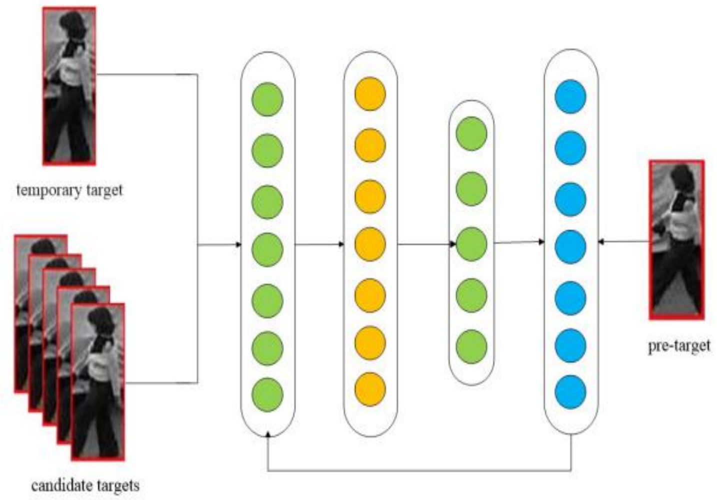


Fig. 6. The Process of determining the target

Here the a^p is the pre-target. a_i^o is the output of selected target *i*.

4 Experiments

There are three key research issues in this article. First, how to avoid the drift problem caused by the model updating speed can not keep up with the scene change speed and how to prevent the gradient diffusion in BP network when feature extraction is used. Second, how to solve the problem that particle filter requires a large number of particles and takes too long. Third, the determination of particle target.

Nowadays, computer improves the precision of image recognition with the help of deep learning. However, a large number of samples are needed during training in practice. For this reason, we make use of Tiny Image dataset [28] in offline training. Tiny Image dataset is consisted of 79 million images. The size of images is 32*32. All images are searched from Internet by Googles Image Search and other engines. We sample 1 million images randomly as the samples in offline training. We transfer images to grayscale and scale the value of pixels to the range [0, 1]. Every pixel is the input of network.

In this section, we evaluate our proposed method on the datasets OTB2013 [8]. The dataset OTB2013 has 50 different sequences and categorizes these sequences with 10 attributes, namely, fast motion (FM), background cluster (BC), motion blur (MB), deformation (DEF), illumination variation (IV), in-plane rotation (IR), low resolution (LR), occlusion (OCC), out-of-plane rotation (OR), out of view (OOV) and scale variation (SV). we also describe implementation details and compare our tracker with the other trackers.

4.1 Experimental Details

Our method is implemented in MATLAB and run at round 12 frames per second on a PC with an Intel Core-i5-4590 CPU (3.70 GHz) and 8 GB of RAM.

In offline training, the learning rate of SDAE is 0.05, momentum is 0.9, weight penalty term is 10−4, sparsity target is 0.05. In online tracking, the learning rate of SDAE is 0.08, momentum is 0.7, weight penalty term is 0.002, the learning rate of BP is 0.09, momentum is 0.7 and weight penalty term is 0.002. In initial of online tracking, the tracker needs to be updated sufficiently, we set iterations is 20 and batch size is 10. In tracking, iterations is set to 5, batch size is 110.

We show the experimental results for the OTB2013 [8]. In order to evaluate the performance of our proposed algorithm, we use three classes of evaluation indexes proposed in OTB2013 [8]: One-Pass Evaluation (OPE). OPE is a traditional evaluation method that runs trackers on each sequence just once.

After running the trackers, precision plots and success plots are applied to the present results. Precision plots mainly refers to the Euclidean distance between the predicted position center point and the center position marked in the benchmark.

With regard to the success plots, an average overlap 10 measure is the most appropriate for tracker comparison [29], as it accounts for both size and position. For this purpose, we use the typical criterion of the Pascal VOC Overlap Ratio (VOR) [30]. Given the bounding *RT* of the result and the bounding box *RG* of the ground truth, the *VOR* can be computed as:

$$VOR = \frac{area(RT \cap RG)}{area(RT \cup RG)}. \tag{4}$$

where $\cap$ and $\cup$ denote the intersection and union of two regions, respectively. Afterwards, a frame whose VOR is larger than a threshold is termed a successful frame, and the ratios of successful frames at the thresholds ranged ranging from 0 to 1 are plotted in the success plots.

For these visual tracking benchmarks, the experimental results are illustrated by precision plot (or rate) and success plot (or rate). The precision plot shows the percentage of successfully tracked frames in the whole sequence and evaluates the performance of the algorithms with Center Location Error (CEL) in pixels, which ranks the trackers as the precision score at 20 pixels. The success plot shows the percentage of successfully tracked frames using the VOR threshold (0.5 is usually taken as the threshold), while the Area Under the Curve (AUC) is used as the metric for ranking. Fig. 6 shows success rate and average center location error.

4.2 Comparative Experimental Results

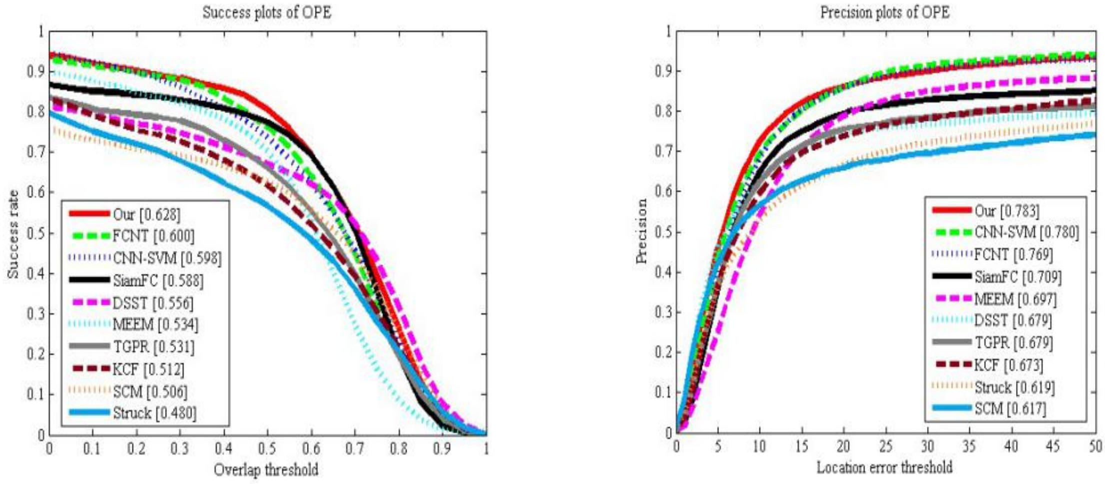
In this section, we analyze our approach on the OTB2013 [8] benchmark by demonstrating the impact of our contributions. For the OTB2013 [8] benchmark, the performances of all the tracking methods are measured by the OPE mechanisms.

We compare our method with 31 representative algorithms, which include 29 algorithms given in the OTB2013 [8] benchmark, such as FCNT [12], CNN-SVM [13], SiamFC [6] and two representative algorithms based on the correlation filter, namely, KCF [17] and DSST [14].

To make the results clearly, we only plot the top 10 ranked trackers in the precision and success plots. The curves of trackers are averaged by the results of 50 sequences. Our tracker outperforms the other trackers in overall evaluation. As shown in Fig. 5, our ACT tracker achieves top rank and the best performance with a large margin in all the tracking plots. Specifically, the proposed tracker achieves a ranking score 0.621 for the success plot and a ranking score 0.783. Compared with the KCF [17] tracker, which has a success ranking score 0.512 and a precision ranking score 0.673, our ACT tracker has obtained improvements over 11.6% and 11%, respectively. Compared with the SiamFC [6] tracker, which has a success ranking score 0.588 and a precision ranking score 0.709, our ACT tracker has obtained improvements over 4% and 7.4%, respectively. Even compared with FCNT [12], which has a success ranking score 0.600 and a precision ranking score 0.769, our tracker also has obtained improvements over 2.8% and 1.4% respectively. This demonstrates that the idea of the adaptive combination tracker for tracking is effective and promising in practice.

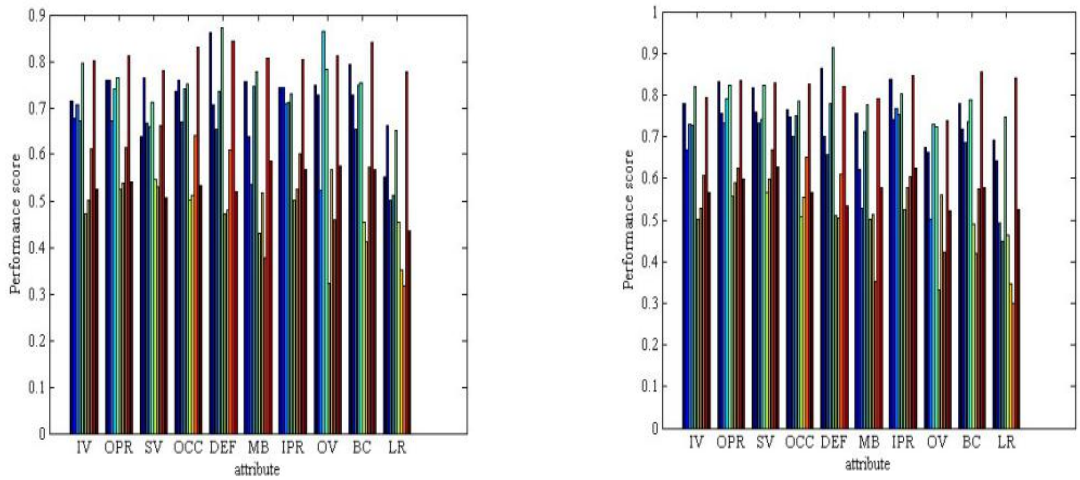
The 50 sequences are annotated with 10 attributes. The attributes of these sequences represent challenges occurred in these sequences. For more details about attributes distribution, you can refer to [2].

The average center location error and success rate of the trackers on each attribute are shown in Fig. 7, which demonstrates the performances of the top ten trackers on the 10 attributes. In Fig 8, each group represents an attribute. Fig. 8 shows that our tracker is substantially outperforms compared with trackers in benchmark in almost attributes. Obviously, our ACT tracker is clearly more accurate and robust. The tracking results of our tracker on some videos and several other trackers is shown in Fig. 9.



(a) Success plot (b) Precision plot

Fig. 7. Success plot and precision plot



(a) Average center location error (b) Success rate

Fig. 8. Average center location error and success rate

Fig. 9. Tracking comparison of each tracker

Compared with the other deep learning trackers, our tracker has advantage in efficiency too. SO-DLT [11] is another deep learning tracker based on CNN, the speed is about 4 to 5 frames per second on a desktop PC with a 3.40GHz CPU and a K40 GPU. The average speed of our tracker is over 12 frames per second with 3.3GHz CPU without GPU accelerating. The comparison of speed between our algorithm and the other three algorithms is shown in Fig. 10, which demonstrates our tracker is better than those algorithms in speed.

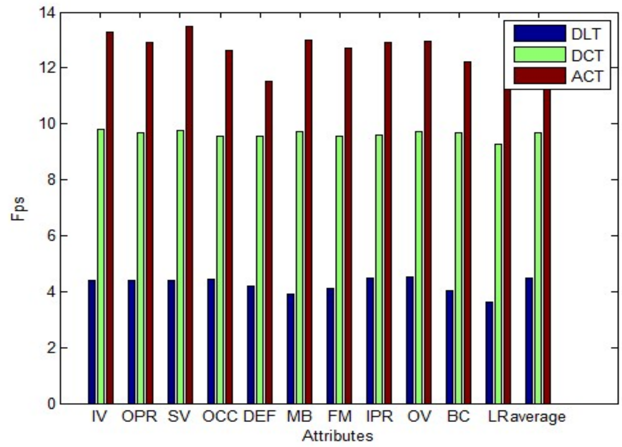


Fig. 10. Speed plot

In Table 1 and Table 2, the difference represents the percentage of the accuracy difference between the tracker in this paper and the tracker with the second score.

Table 1. Precision comparison

Tracker	Our	TLD	SCM	DLT	Struck	D-value
IV	0.6620	0.5256	0.5887	0.4844	0.5495	7.33%
OPR	0.7046	0.5862	0.6122	0.5442	0.5869	9.24%
SV	0.7252	0.5976	0.6680	0.5646	0.6263	5.72%
OCC	0.6942	0.5506	0.6335	0.4939	0.5523	6.07%
DEF	0.7128	0.5001	0.5837	0.4890	0.5136	12.91%
MB	0.6208	0.5082	0.3337	0.4684	0.5443	7.65%
IPR	0.6967	0.5732	0.5897	0.5117	0.6090	8.77%
OV	0.6802	0.5603	0.4236	0.3318	0.5212	11.99%
BC	0.6945	0.4202	0.5738	0.4898	0.5761	11.84%
LR	0.6808	0.3462	0.3008	0.4643	0.5239	15.69%

Table 2. Comparison of success rates

Tracker	Our	TLD	SCM	DLT	Struck	D-value
IV	0.6475	0.4996	0.5929	0.4586	0.5125	5.46%
OPR	0.6973	0.5369	0.6031	0.5138	0.5318	9.42%
SV	0.7054	0.5295	0.6627	0.5456	0.5070	4.27%
OCC	0.6911	0.5104	0.6227	0.4880	0.5202	2.84%
DEF	0.7045	0.4798	0.5848	0.4533	0.5030	11.97%
MB	0.6158	0.5137	0.3575	0.4038	0.5536	6.22%
IPR	0.6746	0.5245	0.5879	0.4887	0.5549	8.67%
OV	0.7166	0.5674	0.4596	0.3221	0.5749	14.17%
BC	0.6674	0.4115	0.5731	0.4531	0.5665	9.43%
LR	0.5522	0.3515	0.3161	0.4540	0.4353	9.82%

4.3 Experimental analysis

In this paper, an auxiliary update model is proposed to solve the problem of feature extraction, which makes the update more diversified. The use of DAE algorithm in the bottom layer can reduce the dimension and prevent the gradient diffusion in BP networks. First, SDAE is used to update the whole auxiliary update model, and then the conventional update method is used to update the whole model more thoroughly. The auxiliary update model enables the tracker to avoid drift to a certain extent when it encounters changes, so as to improve the accuracy of tracking. Then, aiming at the problem that particle filter needs a large number of particles, which leads to too long time, a dynamic particle filter is proposed. The algorithm is based on the apparent The confidence of the model adjusts the number of particles.

Because it can replenish particles in time, it can prevent the tracking accuracy from being affected by the number of particles. The dynamic adjustment of particles improves the tracking speed and avoids the drift caused by the decrease of particles. Finally, aiming at the determination of particle target, the validity of the target is proposed. The algorithm selects the particles with the highest confidence through the apparent model. And compare with each other to remove particles with excessive errors. With the help of auxiliary update model, the first layer and the front target of the network are compared, and the target with the smallest error is taken as the target. The validity of the target prevents drift to a certain extent.

This chapter introduces the platform and OOTB dataset used in the experiment, which contains 50 real video sequences. The accuracy of the proposed algorithm is verified by comparing the existing algorithm DLT, TLD, SCM, Struck with the motion tracking algorithm based on auxiliary update model and validity test in IV,OPR,SV,OCC,DEF,MB and so on. After that, the speed difference of the algorithm is verified, and the efficiency of the proposed algorithm is further proved.

5 Conclusion

In this paper, we proposed the Adaptive Combination Tracker (ACT). Under different situations, SDAE is combined with BP different patterns to improve adaptability. In target determination, candidate targets and pre-target are contrasted with DAE to get rid of instability. Adaptive particle filter regulates the number of particles according to confidence. It recovers particles while confidence has decreased. This algorithm speeds up tracking and guarantees precision. The results of experiments show the superiority of our tracker. In the process of target confirmation, this paper proposes an algorithm to optimize the target by using automatic noise reduction coding. This method has the accuracy and high speed for tracking. The extensive experimental results show the competitiveness of our ACT tracker compared with several trackers, which are widely used in the performance evaluation of tracking algorithm. Although the key problems are studied and some achievements have been made in this paper, the tracking speed can still be improved in the direction of the combination of depth learning and motion tracking, such as speeding up the tracking speed by distributed method. On the basis of the existing tracker, we can establish a more suitable depth learning apparent model and a motion model which can more cooperate with the apparent model, and even combine the two models to establish a new method. The follow-up work of this paper will study other in-depth learning models to make efficiency more efficient.

Acknowledgements

This work is supported by the research project of young teacher autonomy research program of Yanshan University (Grant No. 15LGB014).

References

[1] T. Zhang, S. Liu, C. Xu, B. Liu, M.-H. Yang, Correlation Particle filter for visual tracking, IEEE Transactions on Image Processing 27(6)(2018) 2676-2687.

[2] H. Fan, H. Ling, Parallel tracking and verifying: a framework for real-time and high accuracy visual tracking, in: Proc. 2017 IEEE International Conference on Computer Vision, 2017.

[3] J. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 37(3)(2015) 583-596.

[4] M. Danelljan, G. Bhat, F. Shahbaz Khan, M. Felsberg, Eco: Efficient convolution operators for tracking, in: Proc. 2017 30th IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[5] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, S. Wang, Learning dynamic siamese network for visual object tracking, in: Proc. 2017 IEEE International Conference on Computer Vision, 2017.

[6] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, P. H. S. Torr, Fully-convolutional siamese networks for object tracking, in: Proc. 2016 the 14th European Conference on Computer Vision, 2016.

[7] N. Wang, D.-Y. Yeung, Learning a deep compact image representation for visual tracking, neural information processing systems, in: Proc. 26th International Conference on Neural Information Processing Systems, 2013.

[8] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: A benchmark, in: Proc. 2013 IEEE Conference on Computer Vision and Pattern Recognition, 2013.

[9] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, A.-V.-D. Hengel, A survey of appearance models in visual object tracking, ACM transactions on Intelligent Systems and Technology 4(4)(2013) 58.

[10] H. Yang, L. Shao, F. Zheng, L. Wang, Z. Song, Recent advances and trends in visual tracking: A review, Neurocomputing 74(18)(2011) 3823-3831.

[11] N. Wang, S. Li, A. Gupta, D.Y. Yeung, Transferring rich feature hierarchies for robust visual tracking. <https://arxiv.org/abs/1501.04587>, 2015.

[12] L. Wang, W. Ouyang, X. Wang, H. Lu, Visual tracking with fully convolutional networks, in: Proc. 2015 IEEE International Conference on Computer Vision, 2015.

[13] S. Hong, T. You, S. Kwak, B. Han, Online tracking by learning discriminative saliency map with convolutional neural network, in: Proc. 2015 International Conference on Machine Learning, 2015.

[14] M. Danelljan, G. Hager, F. Khan, M. Felsberg, Accurate scale estimation for robust visual tracking, in: Proc. 2014 British Machine Vision Conference, 2014.

[15] J. Zhang, S. Ma, S. Sclaroff, MEEM: robust tracking via multiple experts using entropy minimization, in: Proc. 2014 European Conference on Computer Vision, 2014.

[16] J. Gao, H. Ling, W. Hu, J. Xing, Transfer learning based visual tracking with Gaussian processes regression, in: Proc. 2014 European Conference on Computer Vision, 2014.

[17] J. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 37(3)(2015) 583-596.

[18] J. F. Henriques, R. Caseiro, P. Martins, J. Batista, Exploiting the circulant structure of tracking-by-detection with kernels, in: Proc. European Conference on Computer Vision, 2012.

[19] M. Danelljan, F. S. Khan, M. Felsberg, J. Van de Weijer, Adaptive color attributes for real-time visual tracking, in: Proc. 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014.

[20] S. Hare, A. Saffari, P.H. Torr, Struck: structured output tracking with kernels, in: Proc. 2011 International Conference on Computer Vision, 2011.

[21] Y. Pang, H. Ling, Finding the best from the second bests- inhibiting subjective bias in evaluation of visual tracking algorithms, in: Proc. 2013 IEEE International Conference on Computer Vision, 2013.

[22] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, M. Shah, Visual tracking: An experimental survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 36(7)(2014) 1442-1468.

[23] S. Avidan, Support vector tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 26(8)(2004) 1064-1072.

[24] S. Avidan, Ensemble tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 29(2)(2007) 261-271.

[25] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, IEEE Transactions on Pattern Analysis and Machine Intelligence 33(8)(2011) 1619-1632.

[26] H. Nam, B. Han, Learning multi-domain convolutional neural networks for visual tracking, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[27] S. Pu, Y. Song, C. Ma, H. Zhang, M.-H. Yang, Deep attentive tracking via reciprocative learning, in: Proc. Neural Information Processing Systems, 2018.

[28] A. Torralba, R. Fergus, W.-T. Freeman, 80 million tiny images: a large data set for nonparametric object and scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 30(11)(2008) 1958-1970.

[29] L. Cehovin, M. Kristan, A. Leonardis, Is my new tracker really better than yours, in: Proc. Applications of Computer Vision, 2014.

[30] M. Everingham, J. Winn, The pascal visual object classes challenge 2010 (voc2010) development kit contents, in: Proc. International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, 2011.