IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 38, No. 4, August, 2021, pp. 1209-1215
Journal homepage: http://iieta.org/journals/ts

A Visual Tracking Algorithm Based on Estimation of Regression Probability Distribution

Xu Han[1], Shang Jiang[2], Jia Yu[3*], Feng Zhang[2]

[1] Department of Military Training and Sports, Changchun University of Science and Technology, Changchun 130022, China
[2] College of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China
[3] Department of Sports Faculty, Liaoning University, Shenyang 110036, China

Corresponding Author Email: wangbin@lnu.edu.cn

https://doi.org/10.18280/ts.380432

Received: 22 April 2021
Accepted: 1 July 2021

Keywords:
target tracking, Siamese network, regression probability distribution, quality assessment

ABSTRACT

During target tracking, the target is often interfered by uncertainties like occlusion and motion blur. The interference leads to inaccurate tracking and even loss of the target. To solve the problem, this paper designs a target tracking algorithm based on the estimation of regression probability distribution (RPDE). Specifically, the uncertainty degree of the tracking frame was estimated by learning the statistical properties of regression parameters, and the quality of that frame was evaluated by fusing the predicted regression probability scores with classification scores. Next, an anchor-free regression mechanism was introduced to improve the computing speed. During network training, a simple and efficient strategy was presented for joint prediction, which jointly expresses classification scores and regression scores to eliminate the extra quality estimation branches in training and prediction. After that, the performance of our algorithm was tested on several public benchmarks, namely, OTB2015, VOT2016, GOT10k, and UAV123, and contrasted with several state-of-the-art algorithms. The results show that the proposed algorithm, named SiamRPDE for short, performed excellently on several benchmarks, and achieved the speed of 125 frames per second (FPS).

1. INTRODUCTION

Target tracking, a hot topic of computer vision, aims to predict the state and position of the target in the subsequent sequence of frames based on those in the first frame. After years of development, target tracking technology has penetrated various fields, namely, video surveillance, robotics, autonomous driving, and human-computer interaction, and become an important part of modern work and life. However, the traditional target tracking algorithms face several problems. For example, the tracking result is serious dampened by environmental changes like occlusion and deformation, as well as illumination changes. Therefore, it is urgent to improve the ability of target tracking algorithms to deal with complex scenarios.

Deep learning (DL) has obvious advantages over traditional approaches in many aspects, e.g., feature extraction, and model robustness. DL methods can significantly improve tracking accuracy, and facilitate target tracking. There are currently two types of DL-based target tracking algorithms: tracking-by-detection algorithms, and Siamese network (Siam)-based algorithms. A typical example of tracking-by-detection algorithms is multi-domain convolution neural network (MDNet) [1]. Considering target tracking as a special classification task, this type of algorithms separates the target from the background, and updates network parameters online, using tracking results. Despite their satisfactory tracking accuracy, the tracking speed is seriously affected by the time-consuming parameter update of convolutional neural network (CNN). To solve the above problem, the other type of DL-based algorithms emerges, using a Siam with consistent structure and shared weights. This type of algorithms specifies the local region of the target as a template in the first frame, and looks for the most similar region to the template in subsequent frames. The network model is obtained through offline training, and the parameters are not updated during the tracking process. Therefore, Siam-based algorithms, represented by fully convolutional Siamese network (SiamFC) [2], can achieve satisfactory speed and accuracy. Subsequently, Valmadre et al. [3] added a correlation filter (FC) layer to SiamFC, creating the CFNet for end-to-end training. Siamese regional proposal network (SiamRPN) [4] divides the tracking task into two stages: separating the foreground from the background with a classification network, and estimating the position of the bounding box with a regression network. SiamPRN greatly increases the tracking accuracy of SiamFC by improving the feature extraction subnet and bounding box generation. On this basis, He et al. [5] developed a twofold Siamese network composed of a semantic branch and an appearance branch (SA-Siam), which uses the heterogeneity of semantic features and convolutional features to achieve complementarity between them. The distractor-aware Siamese region proposal networks (DaSiamRPN) enhances the training of interference samples with semantic background, and combines contextual information to effectively cope with fully occluded target. FlowTrack [6] introduces an optical flow sub-network, which improves tracking accuracy by calculating the optical flow features between multiple templates, and incorporating the temporal features into the matching calculation. SiamRPN++ [7] evaluates the impact of deep feature extraction network on model performance, and overcomes the degradation of tracking accuracy in deep networks induced by the learning of samples with location preferences. Most of the above target tracking algorithms are grounded on SiamFC, which incorporates detection and regression. However, the computing load of the algorithms is inevitably increased by the additional layers or functional modules. Therefore, it is urgent to improve the tracking accuracy, while keeping the model simple and efficient.

In recent years, the first-order full convolutional network (FCOS), which is anchor-free, proposal-free, and centerness-enabled, gains much favor from scholars, because the FCOS structure saves memory by avoiding the estimation of numerous parameters associated with anchor boxes during the training. Ocean [8] designed a network structure coupling object perception network with bounding box regression: an anchorless box scheme is adopted to predict the center point and the distance from the four sides, before positioning the bounding box. Siamese box adaptive network (SiamBAN) [9] improves the delimitation scheme of positive and negative samples, utilizing an ellipse label. Siamese fully convolutional classification and regression (SiamCAR) [10] generates multiple semantic similarity maps on a deep correlation layer, and derives classification and regression information from multi-channel response maps. To improve the tracking accuracy, SiamFC++ [11] introduces the classification quality evaluation score, and applies weight to positive samples. The classification score is multiplied with the quality evaluation result as the confidence score. During model training, the network parameters are optimized by maximizing the confidence score. During prediction, the confidence score is used as the filter condition to realize non-maximum suppression (NMS). The confidence evaluation mechanism makes the regression more accurate than the four-point coordinate of the regression bounding box. But this mechanism has its own limitations. For instance, that the confidence score depends on the selection of the loss function and the generation of the regression sample labels during the training process [12]. Further exploration is necessary to determine the reasonable form of the loss function, and label regression samples correctly.

Through the above analysis, this paper proposes a target tracking algorithm based on the estimation of regression probability distribution (RPDE). To improve the tracking accuracy in complex scenes, the probability distribution of the bounding box was modeled to estimate the general regression distribution, and guide the flexible generation of bounding box. The training loss function consists of a classification branch and a regression branch, aiming to enhance the reliability of the confidence score. The learned regression distribution features were employed to guide the generation of confidence score, and the classification and regression branches were trained jointly to enhance the connection between the classification and regression information. To achieve real-time target tracking, a lightweight network structure was employed to reduce the parameters that need to be computed, such that the target tracking algorithm can operate at a high speed. Finally, the proposed algorithm, named SiamRPDE for short, was tested on different datasets. The results confirm the robustness of our algorithm for target tracking in complex scenarios.

2. ALGORITHM DESIGN

As shown in Figure 1, the proposed SiamRPDE consists of a similarity matching model and a correlation vector acquisition model. In the similarity matching model, z and x are imported to the baseline feature extraction network, which includes 5 convolutional layers and 2 pooling layers. In the first two convolutional layers, the kernel size is 11×11 and 5×5, respectively, and the step size is 2 and 1, respectively. In the last three convolutional layers, the kernel size is 3×3, and the step size is 1. The two pooling layers are located between the 1st and 2nd convolutional layers, and the 2nd and 3rd convolutional layers, respectively, using the kernel size of 3×3, and the steps size of 2. The depth correlation degree is calculated, and the correlation vectors are obtained, after obtaining the corresponding baseline features.

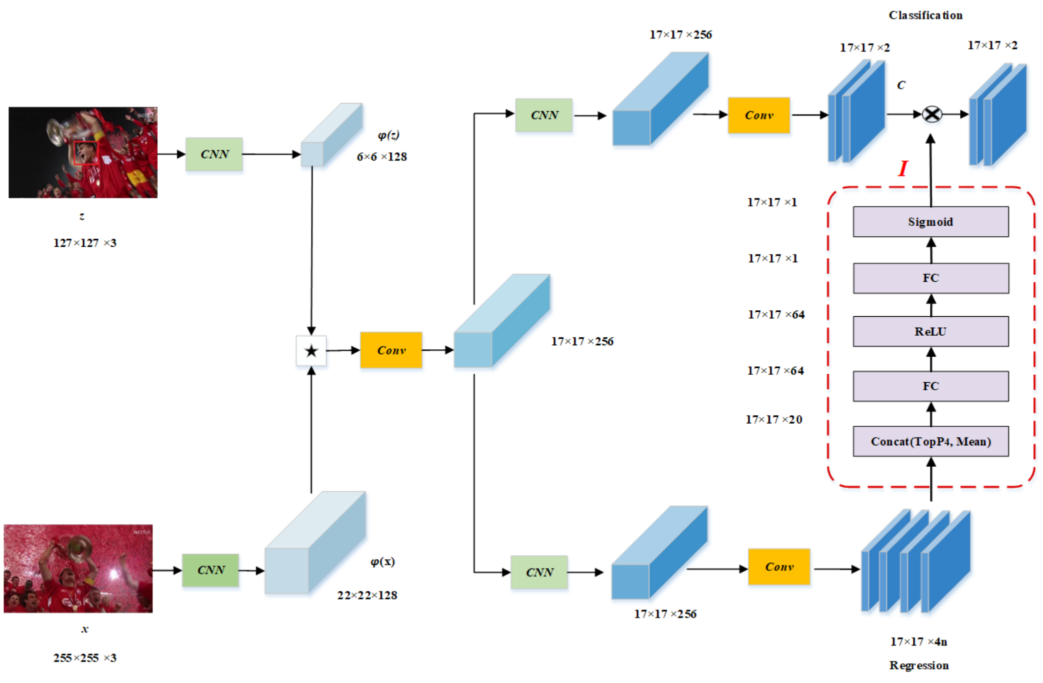


Figure 1. Algorithm structure

The correlation vector acquisition model is divided into an upper classification branch, and a lower regression branch. In the classification branch, correlation vectors go through dimensionality reduction to obtain the initial classification result, C. Then, the probability distribution of the regression parameters is estimated, and a part of the probability value is selected to be sent to be small subnet. After the processing by two fully-connected layers and two activation functions, a one-dimensional (1D) scalar value I is obtained, and multiplied with the classification result C. In this way, classification and regression are integrated into an organic whole. The product between I and C provides a criterion for NMS screening, making the tracking quality assessment more reasonable.

2.1 Siamese similarity matching

This paper presents a target tracking algorithm based on RPDE. The main model of the algorithm follows the Siamese similarity matching framework [3], which has an upper branch and a lower branch. Each branch corresponds to an input terminal. As shown in Figure 1, the input terminal of the upper branch is the local region of the target in the first frame z. This frame serves as a template for similarity matching. The lower branch corresponds to the frame x to be tracked. After z and x are imported, the baseline feature extraction network $\varphi(.)$ will output convolutional features $\varphi(z)$ and $\varphi(x)$, and calculate the similarity between the two features:

$$R=\varphi(x)\star\varphi(z) \tag{1}$$

where, $\star$ is a deep cross-correlation operation, in which the target template feature $\varphi(z)$ is correlated with the search image feature $\varphi(x)$ channel by channel.

2.2 Joint expression of classification and regression

When the tracking result is evaluated by confidence score [8], the product between the intersection over union (IoU) score and the classification score is usually taken as the confidence score. The higher the score, the more reliable the tracking result. Obviously, the confidence score depends on the IoU score and the classification score. Hence, a negative sample with a low classification score but a high IoU score may receive a high confidence score. In this case, the negative sample will be incorrectly selected as the tracking result, resulting in tracking failure. What is worse, the classification branch and the IoU branch are trained separately, without any direct connection between them. But the outputs of the two branches are directly multiplied for judgement in the prediction stage. This leads to the divergence between training and prediction, which in turn causes tracking failure.

To solve the above problems, this paper employs the loss function [13] that formally integrates classification and regression, eliminating the need for additional quality assessment branches. The joint expression of classification and regression can be expressed as:

$$L_{\mathbf{cls}}=-|y-j|^{\beta}\left[(1-y)\log(1-j)+y\log(j)\right] \tag{2}$$

where, j is the predicted classification score; y is the true class label; $-|y-j|^{\beta}$ is the mediator term of the overall function. The larger the difference between j and y, the greater the value of the mediator term, and the more the training focuses on difficult samples. The value of L_{cls} can be minimized at $j=y$. Formula (2) is a special form of focal loss function [14], which could only handle discrete label values. The special form supports supervised learning of continuous values, while retaining the excellence of focal loss function in balancing difficult and easy samples. To obtain continuous labels distributed over the interval [0, 1], the values of classification labels can be processed as follows:

$$j=\mathbf{C}\times I \tag{3}$$

where, **C** is the class label of a sample, i.e., the label of positive or negative sample; I is the IoU score, which is a scalar value to be computed via the regression branch. Formula (3) combines classification branch with regression branch to jointly express classification score and regression score. Therefore, the score of j lays a reasonable basis for evaluating the quality of classification and regression.

2.3 RPDE network

In our target tracking algorithm, the regression branch needs to estimate the positions of the top, bottom, left, and right vertices of the rectangular bounding box. In the FCOS-based target tracking algorithm [15], the bounding box regression needs to calculate the distances from the point to the four edges. The common point between the two algorithms is the need to predict four certain values.

From the perspective of probability distribution, our algorithm is equivalent to solving the Dirac distribution. In practical applications, such a distribution is an ideal state, and thus lacks generality. In some scenarios, it is difficult to represent the boundary of the bounding box with an exact value, but with a fuzzy value.

As shown in Figure 2, the tracking target is an orange doll. The bounding box of the doll has fixed boundaries in the upper, left, and right directions. Yet the lower boundary is obscured by interfering objects. Hence, the probability distribution is only reasonable within a certain range. In this situation, the Dirac distribution cannot model the uncertain factors, nor estimate the fuzziness of regression labels. This calls for the estimation of a bounding box probability distribution applicable to the general case.

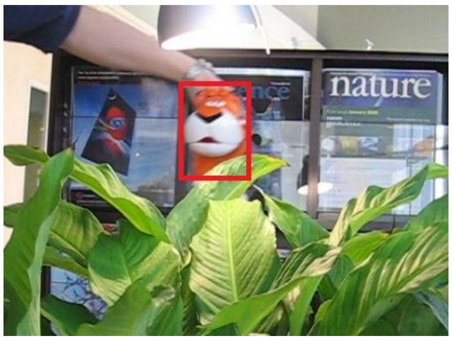


Figure 2. Bounding box uncertainty

To estimate the general distribution of the bounding box, the regression problem can be described by a probability density function:

$$\hat{y}=\int_{-\infty}^{+\infty}P(x)x\,dx=\int_{y_0}^{y_n}P(x)x\,dx \tag{4}$$

where, $\hat{y}$ is the predicted value of the regression parameter; $P(x)$ is the probability distribution of the target bounding box.

In FCOS-based regression, the target parameter is the distance from the point in the bounding box to each side, which can be represented by an interval. In this paper, the interval of regression label y is set to $[y_0, y_n]$. The integration over a continuous space cannot be solved directly and easily in a CNN. The operation needs to be solved instead with the help of discrete sampling. To estimate the probability distribution of the bounding box, the interval $[y_0, y_n]$ was uniformly divided into n intervals of the size 1: $[y_0, y_1, \ldots, y_{n-1}, y_n]$. Li et al. [13] set the distribution interval of t, b, l, and r in the regression labels to [1, 16], by counting the regression label range in the COCO dataset. The regression label can be predicted by calculating the expectation of the probability function:

$$\hat{y} = \sum_{i=0}^{n} P(y_i) y_i \tag{5}$$

where, $P(y_i)$ is the probability that the regression value is $P(y_i)$. To control $P(y_i)$ within [0, 1], the softmax function was introduced to process a total of $n+1$ values of $P(y_0), P(y_1), \ldots, P(y_n)$ to meet $\sum_{i=0}^{n} P(y_i) = 1$. However, this constraint alone may lead to multiple probability distributions, which will reduce the effectiveness of learning. Hence, the problem should be constrained as follows:

$$L_{\mathbf{reg}} = -\left(\left(y_{i+1} - y\right)\log\left(P\left(y_i\right)\right) + \left(y - y_i\right)\log\left(P\left(y_{i+1}\right)\right)\right) \tag{6}$$

Formula (6) is the regression loss function, which enables the network to converge quickly near the regression label. Thus, the point close to the target has a high probability. The loss function takes the form of a cross-entropy function, aiming to optimize the probabilities of the left and right positions (i.e., y_i and y_{i+1}) around the regression label y. In this way, the bounding box distribution can approximate the real position as much as possible.

Through the above steps, it is possible to learn a general regression probability distribution of the bounding box. This distribution is strongly correlated with the true position of the target. Normally, the peak probability in the probability distribution is linearly correlated with the true IoU value [16]. For each regression parameter, the top k probabilities ($k=4$) were taken from the corresponding distribution to reflect the main features of bounding box distribution. Then, the mean of the k probabilities was combined with the k values:

$$\mathbf{F} = \mathbf{Concat}\left(\left\{Topkm\left(\mathbf{P}^{w}\right) \middle| w \in \{t, b, l, r\}\right\}\right) \tag{7}$$

The target regression parameters, i.e., t, b, l, and r, represent the four directions. Thus, $4(k+1)$ parameters need to be stored in the eigenvector **F**. These parameters were imported to a subnet (red dashed area in Figure 1) including two fully-connected layers, a rectified linear unit (ReLU) layer, and a sigmoid layer. According to the eigenvector **F** imported to the subnet, the IoU score I can be derived:

$$I = \sigma\left(W_2 \delta\left(W_1 \mathbf{F}\right)\right) \tag{8}$$

where, σ is the ReLU function; δ is the sigmoid activation function; $W_1 \in \mathbb{R}^{p\times 4(k+1)}$; $W_2 \in \mathbb{R}^{1\times p}$; c is the channel dimension of the hidden layer. After that, the I value was substituted into formula (3), and multiplied with the classification score to obtain the joint expression of classification and regression.

3. EXPERIMENTS AND RESULTS ANALYSIS

3.1 Experimental environment and parameter settings

Our algorithm was written in Python under the PyTorch framework, and implemented on a computer with Intel Core i7-8700k central processing unit (CPU), and NVDIA GTX1080Ti graphics processing unit (GPU). The algorithm adopts the SiamFC similarity matching framework. The modified AlexNet was selected as the baseline feature extraction network, and the network parameters were initialized as the trained values on the ImageNet dataset. Our overall model was trained on two datasets: GOT-10K and LaSOT. Two images are imported to the network for training and testing: a template image of 127×127, and a search image of 255×255.

For the experiments, our model was trained for 20 rounds. In the first 5 round, the learning rate was set to 0.001. In the last 15 rounds, the learning rate decayed from 0.005 to 0.0005. During the iterative training, the network parameters were optimized through stochastic gradient descent, with the momentum of 0.9, and the batch size of 32.

To verify its effectiveness, our algorithm was applied to five world-famous public datasets: GOT10k [17], OTB100 [18], VOT2016 [19], UAV123 [20]. The algorithm performance was evaluated with the criteria for each dataset, and compared with multiple excellent new target tracking algorithms.

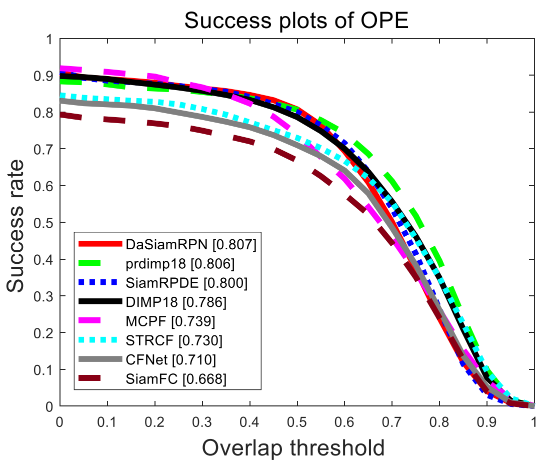
3.2 Performance on OTB100 dataset

OTB100 is an important public dataset in the field of target tracking. The dataset contains 100 sets of frame sequences. Each set is carefully labeled with 11 attributes: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane-rotation, out-plane-rotation, out-of-view, background clutters, and low resolution.

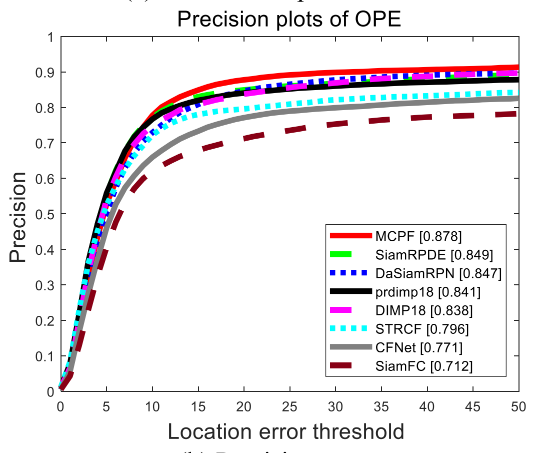
The algorithm performance on the dataset is generally evaluated against two criteria: precision and success rate. The precision refers to the proportion of frames with a target center position error below a preset threshold, while the success rate refers to the proportion of frames with an overlap rate greater than a certain threshold.

Seven excellent new target tracking algorithms were selected for comparison, namely, spatial-temporal regularized correlation filters (STRCF) [21], DaSiamRPN, SiamFC, CFNet, data integration with modeled predictions 18 (DiMP18), PrDiMP18, and multi-task correlation particle filter (MCPF) [22]. The overall performance of each algorithm on OTB100 is displayed as a precision plot and a success rate (area under the curve, AUC) plot, using the OPE (precision and success rate) criteria (Figure 3).

Our SiamRPDE algorithm ranked the third in success rate (0.807) and second in precision (0.849). In general, the performance of our algorithm is stable and better than most of the algorithms with the Siamese Network matching structure on OTB100. Compared with the baseline algorithm SiamFC, our algorithm led by 14.2% in success rate, and 13.7% in precision. Hence, the use of RPDE substantially improves the performance of the baseline similarity matching model.



(a) Success rate plots of OPE



(b) Precision curve

Figure 3. Tracking results on OTB2015

3.3 Performance on GOT10k dataset

The GOT10k dataset is a large novel dataset, with no overlap in target classes between the training and test sets. During the test on GOT10k, our algorithm strictly follows the evaluation rule of this dataset. Only 10,000 sequences of the training subset of GOT10k were used for training, and 180 sequences in the test subset were used to test the model. The evaluation criteria on this dataset are average overlap (AO) and success rate (SR). AO stands for the mean overlap between the estimated bounding box and the true bounding box; $SR_{0.5}$ stands for the proportion of frames with the area ratio of estimated bounding box to true bounding box greater than 0.5; $SR_{0.75}$ stands for the proportion with the area ratio greater than 0.75.

Our algorithm was contrasted with six algorithms, including SiamFC, efficient convolution operators (ECO), SiamRPN++, series-parallel matching (SPM) [23], MDNet, and SiamRPN. The tracking results of all seven algorithms are shown in Table 1, where the best performance under different criteria is marked in red, and the second best is marked in blue.

The proposed SiamRPDE algorithm achieved an AO score of 50.3, an $SR_{0.5}$ score of 60.6, and an $SR_{0.75}$ score of 33.5. The overall performance of the algorithm ranked the top three among all comparative algorithms. In addition, our algorithm reached the speed of 124.5FPS, faster than any other algorithm. With fewer layers of the feature network, our algorithm obtained a good performance with a simpler and lighter network. This means regression branch information effectively underpins the evaluation of location quality.

3.4 VOT2016 dataset

This subsection evaluates the performance of multiple excellent tracking algorithms on the VOT2016 dataset, which covers 60 challenging frame sequences. The performance evaluation was carried out with the official VOT toolkit. The evaluation criteria include accuracy (A), robustness (R), and expected average overlap (EAO). Specifically, accuracy is calculated by the overlap rate, and robustness by the number of failed tracking. Here, EAO value and EAO curve are chosen to sort all algorithms. The EAO value comprehensively reflects accuracy and robustness. The higher the value, the better the performance. For the EAO curve, the abscissa is the length of the frame sequence in the specified interval of [96, 348], and the ordinate is the mean accuracy of videos of the same length.

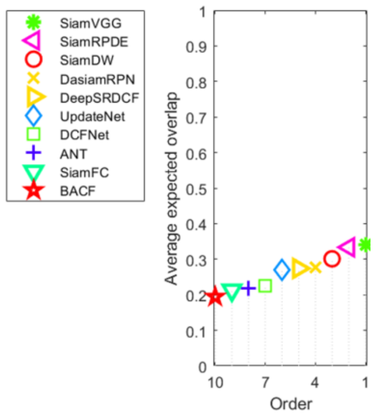
The contrastive algorithms on VOT2016 include SiamFC, SiamVGG [24], SiamDW [25], DaSiamRPN, DeepSRDCF [26], UpdateNet [21], etc. Figure 4 sums up the experimental results of all algorithms. Subgraph 4(a) shows the EAO ranking, where the abscissa represents the algorithm ranking from right to left, and the ordinate is the standardized EAO value. Our algorithm (pink triangles) ranked the second in this subgraph. Subgraph 4(b) shows the EAO curves, in which our algorithm (pink curve) ranked the second, as measured by AUC. Table 2 lists the EAO scores of each algorithm. The top 2 are in red and blue, respectively. SiamVGG achieved the highest EAO score, slightly higher (+0.76%) than that (0.3336) of our algorithm; the EAO score of our algorithm was 3.25% greater than SiamDW, and 12.07% ahead of the baseline algorithm SiamFC.

3.5 Performance on UAV123 dataset

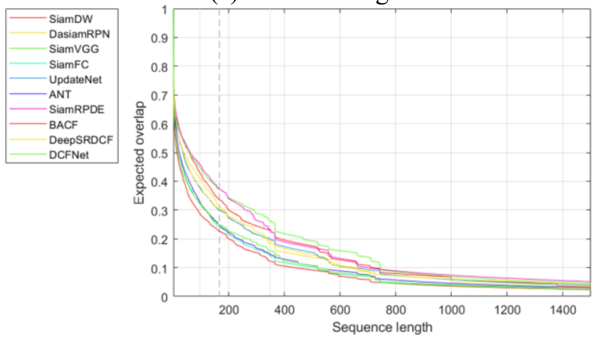
The UAV123 dataset contains 123 frame sequences, all of which are captured by drones at different heights. The mean length of the sequences is 915 frames, and the total number of frames exceeds 110k. The evaluation criteria of this dataset are consistent with those of the OTB100 dataset. That is, each algorithm was evaluated by precision and success rate. A total of six excellent algorithms were compared with our algorithm: SiamRPN++, DaSiamRPN, UPDT [27], ECO, CCOT [28], and SRDCF [29].

Table 1. Tacking results on GOT10k dataset

Algorithm	SiamFC	ECO	SiamRPN++	SPM	MDNet	SiamRPN	SiamRPDE
AO	34.8	31.6	51.8	51.3	29.9	48.3	50.3
$SR_{0.5}$	35.3	30.9	61.8	59.3	30.3	58.1	60.6
$SR_{0.75}$	9.8	11.1	32.9	35.9	9.9	27.0	31.5
FPS	25.8	2.6	49.8	72.3	1.5	97.5	124.5



(a) EAO ranking



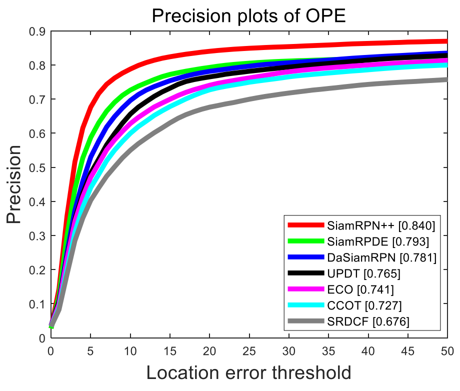
(b) EAO curves

Figure 4. Tracking results on VOT2016 dataset

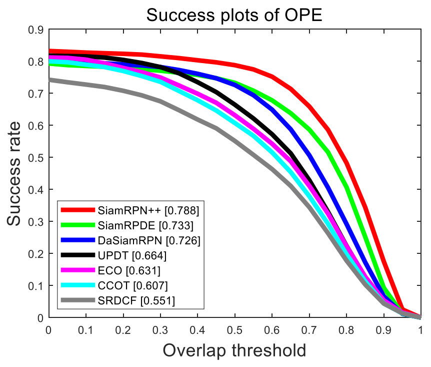
Table 2. EAO scores of each algorithm on VOT2016 dataset

Trackers	EAO
SiamVGG	0.3412
SiamRPDE	0.3336
SiamDW	0.3011
DaSiamRPN	0.2762
DeepSTRCF	0.2742
UpdateNet	0.2689
SiamFC	0.2129

Figure 5 compares the performance of these algorithms on UAV123 dataset. Our algorithm ranked the second in precision and success rate, only behind SiamRPN++. The latter achieved the best performance, thanks to its baseline feature extraction network of ResNet50. The multiple layers of the network integrate feature information at different levels, providing more features for similarity matching. Furthermore, our algorithm outperformed DL models like DaSiamRPN and UPDT, and far outshined related filtering algorithms ECO, CCOT, and SRDCF, in both precision and success rate.



(a) Precision curve



(b) Success rate curve

Figure 5. Tracking results on UAV123 dataset

4. CONCLUSIONS

This paper proposes a visual target tracking algorithm based on RPDE to pinpoint targets in complex real-world environment with multiple interferences. The experimental results on the algorithm yield the following conclusions:

(1) By modeling the true bounding box distribution, our algorithm realizes flexible generation of the bounding box, and improves the tracking accuracy in complex scenes.

(2) The confidence assessment of tracking quality and accurate prediction are realized through the joint expression of regression probabilities and classification scores. Besides, the end-to-end training eliminates the difference between training and testing for the additional quality estimation branch.

(3) For the sake of computing speed and tracking accuracy, our algorithm has a limited number of network layers and a simple model structure.

(4) The rectangular bounding box of our algorithm might include some background interferences, if the target is small. In future, the target tracking will be refined by optimizing the labeling of the bounding box.

REFERENCES

[1] Han, B., Sim, J., Adam, H. (2017). Branchout: Regularization for online ensemble tracking with convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3356-3365. https://ieeexplore.ieee.org/document/8099546

[2] Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H. (2016). Fully-convolutional Siamese networks for object tracking. In European Conference on Computer Vision, pp. 850-865. https://doi.org/10.1007/978-3-319-48881-3_56

[3] Valmadre, J., Bertinetto, L., Henriques, J., Vedaldi, A., Torr, P.H. (2017). End-to-end representation learning for correlation filter based tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2805-2813. https://ieeexplore.ieee.org/document/8100014

[4] Li, B., Yan, J., Wu, W., Zhu, Z., Hu, X. (2018). High performance visual tracking with Siamese region proposal network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8971-8980. https://ieeexplore.ieee.org/document/8579033

[5] He, A., Luo, C., Tian, X., Zeng, W. (2018). A twofold Siamese network for real-time object tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4834-4843. https://ieeexplore.ieee.org/document/8578606

[6] Zhu, Z., Wu, W., Zou, W., Yan, J. (2018). End-to-end flow correlation tracking with spatial-temporal attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 548-557. https://ieeexplore.ieee.org/document/8578162

[7] Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J. (2019). Siamrpn++: Evolution of Siamese visual tracking with very deep networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4282-4291. https://ieeexplore.ieee.org/document/8954116

[8] Zhang, Z., Peng, H., Fu, J., Li, B., Hu, W. (2020). Ocean: Object-aware anchor-free tracking. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow. XXI 16: 771-787. https://doi.org/10.1007/978-3-030-58589-1_46

[9] Chen, Z., Zhong, B., Li, G., Zhang, S., Ji, R. (2020). Siamese box adaptive network for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6668-6677. https://ieeexplore.ieee.org/document/9157457

[10] Guo, D., Wang, J., Cui, Y., Wang, Z., Chen, S. (2020). SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6269-6277. https://ieeexplore.ieee.org/document/9157720

[11] Xu, Y., Wang, Z., Li, Z., Yuan, Y., Yu, G. (2020). SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines. In Proceedings of the AAAI Conference on Artificial Intelligence, 34(7): 12549-12556. https://doi.org/10.1609/aaai.v34i07.6944

[12] Danelljan, M., Gool, L.V., Timofte, R. (2020). Probabilistic regression for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7183-7192. https://ieeexplore.ieee.org/document/9157124

[13] Li, X., Wang, W., Wu, L., Chen, S., Hu, X., Li, J., (2020). Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. arXiv preprint arXiv:2006.04388. https://arxiv.org/abs/2006.04388

[14] Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P. (2020). Focal loss for dense object detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(2): 318-327. https://ieeexplore.ieee.org/document/8417976

[15] Tian, Z., Shen, C., Chen, H., He, T. (2019). Fcos: Fully convolutional one-stage object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9627-9636. https://ieeexplore.ieee.org/document/9010746

[16] Li, X., Wang, W., Hu, X., Li, J., Tang, J., Yang, J. (2021). Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11632-11641. https://arxiv.org/abs/2011.12885

[17] Huang, L.H., Zhao, X., Huang, K.Q. (2021). Got-10k: A large high-diversity benchmark for generic object tracking in the wild. IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(5): 1562-1577. https://doi.org/10.1109/TPAMI.2019.2957464

[18] Y. Wu, J. Lim, and M. H. Yang. (2015). Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(9): 1834-1848. https://ieeexplore.ieee.org/document/7001050

[19] Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., ˇCehovin Zajc, L. (2016). The Visual Object Tracking VOT2016 challenge results. European Conference on Computer Vision, pp. 777-823. https://link.springer.com/chapter/10.1007/978-3-319-48881-3_54

[20] Mueller, M., Smith, N., Ghanem, B. (2016). A benchmark and simulator for UAV tracking. European conference on computer vision. Springer, pp. 445-461. https://link.springer.com/chapter/10.1007/978-3-319-46448-0_27

[21] Bhat, G., Johnander, J., Danelljan, M., Khan, F.S., Felsberg, M. (2018). Unveiling the power of deep tracking. In Proceedings of the European Conference on Computer Vision (ECCV), pp. 483-498. https://link.springer.com/chapter/10.1007/978-3-030-01216-8_30

[22] Zhang, T., Xu, C., Yang, M.H. (2017). Multi-task correlation particle filter for robust object tracking. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4335-4343. https://ieeexplore.ieee.org/document/8099995

[23] Wang, G., Luo, C., Xiong, Z., Zeng, W. (2019). Spm-tracker: Series-parallel matching for real-time visual object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3643-3652. https://ieeexplore.ieee.org/document/8953375

[24] Li, Y., Zhang, X. (2019). SiamVGG: Visual tracking using deeper Siamese networks. arXiv preprint arXiv:1902.02804. https://arxiv.org/abs/1902.02804v2

[25] Zhang, Z., Peng, H. (2019). Deeper and wider siamese networks for real-time visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4591-4600. https://ieeexplore.ieee.org/document/8953458

[26] Li, F., Tian, C., Zuo, W., Zhang, L., Yang, M.H. (2018). Learning spatial-temporal regularized correlation filters for visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4904-4913.

[27] Danelljan, M., Robinson, A., Khan, F.S., Felsberg, M. (2016). Beyond correlation filters: Learning continuous convolution operators for visual tracking. In European Conference on Computer Vision, pp. 472-488. https://doi.org/10.1007/978-3-319-46454-1_29

[28] Danelljan, M., Hager, G., Shahbaz Khan, F., Felsberg, M. (2015). Learning spatially regularized correlation filters for visual tracking. In Proceedings of the IEEE International Conference on Computer Vision, pp. 4310-4318. https://ieeexplore.ieee.org/document/7410847

[29] Danelljan, M., Bhat, G., Shahbaz Khan, F., Felsberg, M. (2017). Eco: Efficient convolution operators for tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6638-6646. https://ieeexplore.ieee.org/document/8100216