IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 14, 2020, accepted November 23, 2020, date of publication November 30, 2020,
date of current version December 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041294

Long Short-Term Memory Networks Based on Particle Filter for Object Tracking

YANLI LIU[1,2], JINGJING CHENG[1], HENG ZHANG[1,2], HANG ZOU[3], AND NAIXUE XIONG[4,5], (Senior Member, IEEE)

[1]School of Information Engineering, East China Jiaotong University, Nanchang 330013, China
[2]School of Electronic Information, Shanghai Dianji University, Shanghai 201306, China
[3]Wuhan Research Institute of Posts and Telecommunications, Wuhan 430074, China
[4]College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
[5]Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK 74464, USA

Corresponding author: Heng Zhang (zhangheng@sdju.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61663010, Grant 61963017, and Grant 61563014; in part by the Outstanding Youth Planning Project of Jiangxi Province, China, under Grant 20192BCBL23004; and in part by the Key Research and Development Project of Jiangxi Province, China, under Grant 20192BBE50077.

ABSTRACT Due to the uncertainty of object motion, object tracking is a more difficult state estimation problem. The traditional tracking method based on particle filter has come into wide use, but it has high complexity and poor real-time performance in the process of tracking. As long as there are enough training data, the method based on deep neural network can fit any mapping well. In this paper, a structured Long Short-Term Memory Network based on Particle Filter(LSTM-PF) is proposed to learn and model video sequences with high uncertainty. This network draws on the idea of particle filter, which uses a set of weighted particles to approximate the latent variable and updates the latent state distribution through the LSTM gating structure according to Bayesian rules. We conduct a comprehensive experiment on two benchmark datasets: OTB100 and VOT2016. The experimental results show that our tracker has better performance than other trackers, which can effectively reduce the calculation redundancy and improve the tracking accuracy.

INDEX TERMS Object tracking, particle filter, deep neural network, long short-term memory.

I. INTRODUCTION

Visual tracking has a very important research value in intelligent monitoring [1], behavior recognition [2], [3], man-machine interaction [4], etc. Its main task is to obtain the position and motion trajectory of the interested object in video or image sequence. Recent advances in object detection approaches [5], [6] have promoted the development of some tracking-by-detection methods [7]. Although object tracking has accomplished great success in the area of computer vision, there are still many interference factors, such as scale variance, partial occlusion, fast motion and background noise, etc. These make robust tracking a challenging problem.

At present, there are two main types of tracking algorithms: generative model and discriminant model. The generative model regards tracking as an optimization task to find the region with the highest matching degree for the object, while the discriminant model regards tracking as a classification task and tries to distinguish the object and background in the object area [8]. The traditional tracking algorithm based on particle filter is one of the representatives of the generative model. It has the advantages of simplicity and easy implementation, and can be effectively applied to the state of uncertain systems. It shows good performance in nonlinear and non-Gaussian estimation problems [9]. However, the classic particle filter usually uses the global dynamic model, which cannot distinguish the object from the background well. When the object is partially occluded or similar to the background, the particle filter algorithm cannot track the object accurately, and such phenomena as object drift and loss will occur [10].

In order to solve these problems, more and more discriminative methods based on Deep Neural Networks (DNN) have been proposed. Different from most traditional model-based methods, DNN-based methods are optimized to learn DNN from available training data [11]. Usually, the method

The associate editor coordinating the review of this manuscript and approving it for publication was Ali Shariq Imran.

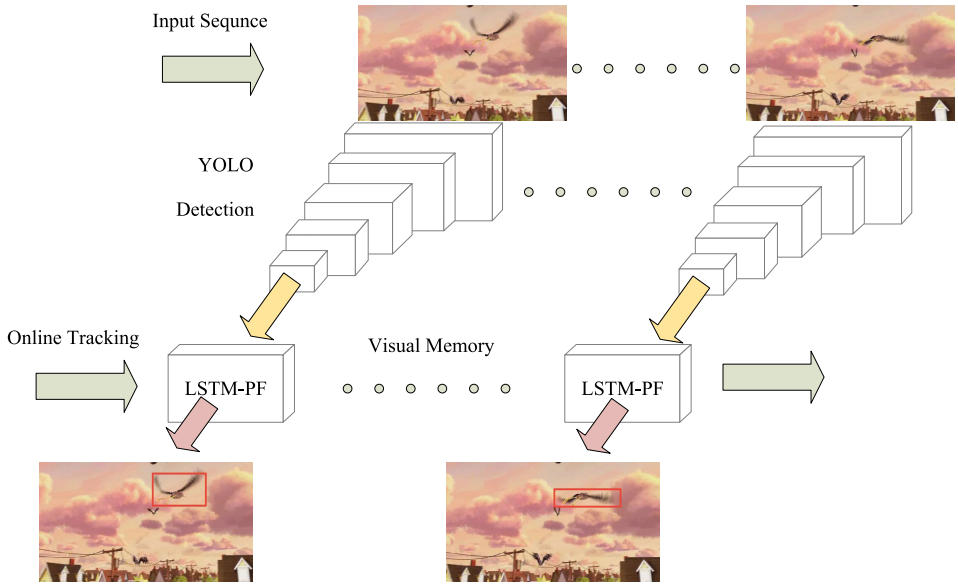


FIGURE 1. Overview of the tracking framework.

based on DNN is used as classifier to solve the problem of object tracking. For example, the Convolutional Neural Network (CNN) is used as a binary classifier when initializing the object trajectories, and distinguishs the real object trajectory from the false object track according to some hand-made features [12]. When CNN is used as a general feature extractor and trained on a general dataset, it has better performance than the hand-made features and used to model the appearance of the object [13].

However, CNN has no time correlation, and the tracking based on the CNN method only uses appearance features to build a redundant appearance model to predict the object trajectory of the next frame, which is not enough to track any moving object [14]. Some recent work has studied the correlation characteristics between video sequences in object tracking. Gan *et al.* [15] have tried to train a Recurrent Neural Network (RNN) [16] for object tracking. Ning *et al.* [17] combines You Only Look Once(YOLO) detection [18] with LSTM network [19] to process video data, and proposes a Recurrent-YOLO(ROLO) algorithm to track objects. RNN and LSTM are very successful in object tracking, because they can accurately predict the position of the object in the current frame by using the relationship between context in the video sequence. Although they have all proposed good methods to improve the accuracy of object tracking, they have not fully exploited the potential of RNN to deal with object motion uncertainty and provide better estimation accuracy. We know that neural network has a powerful expression ability, which can make it approximate to any desired accuracy with a sufficiently complex network. From the point of view of posterior state probability, the conditional density of real state of a given measurement value is meaningful in object tracking, which can be approximated by DNN. In addition, a properly selected network should be able to handle the complexity caused by motion uncertainty and noise measurements. Specifically, the ability of LSTM based structure is better than other methods in extracting sequence information. LSTM needs "memory", which summarizes and tracks information in the input sequence. Memory states are usually not observable, so the need for a latent variable, i.e., a posterior state distribution that captures the sufficient statistic of the input for the input to be predicted. However, the function of LSTM to update latent variables is deterministic, so LSTM can not model uncertainty in potential state. In order to solve this problem, the belief is represented as a set of sampling states and approximate Bayesian inference is carried out, which can effectively model the uncertainty in video sequence. Therefore, we propose a new visual tracking framework to improve the robustness and efficiency of tracking, called Deep Particle Filter Tracker (DPFT). We take advantage of the ability of particle filtering to approximate the posterior state distribution to a set of weighted particles, and use the powerful approximation capabilities of recurrent neural networks. The approximate framework is roughly shown in Figure 1. First, we use YOLO to get preliminary position inference. Then in the tracking stage, we use the LSTM-PF to obtain the final bounding box information of the object. This network explicitly models the uncertainty of its internal structure, which can better handle the measurement of motion uncertainty and noise, and improve the tracking robustness. The main contributions of this paper are as follows:

(1) The network maintains a latent state distribution, represented by a set of weighted particles, which captures all possible states of the current object movement.

(2) Different from the deterministic nonlinear update of the fully connected layer in LSTM, LSTM-PF uses a random

particle filter algorithm update strategy, which can better deal with the uncertainty of object motion during tracking.

(3) The combination of offline pre-training and online tracking fine-tuning solves the problem of lack of training samples effectively.

The rest of this paper is organized as follows. In Section II, some previous works related to this study are introduced. In Section III, we introduce the relevant background of the paper. In Section IV, we describe the proposed object tracking framework in detail. The details and results of the experiment are analyzed in Section V. Finally, we conclude our study in Section VI.

II. RELATED WORK

In this section, we introduce two aspects of work related closely to our research: (i) tracking by detection; (ii) tracking by RNNs.

A. TRACKING BY DETECTION

Object tracking should first determine the position of the object to be tracked in the initial frame. Most tracking methods are manually marked, but combined with the current rapid development of object detection, automatic selection will be more convenient [20]. The tracking-by-detection methods take object tracking as a detection problem in the image of interest, and use online learning classifier to distinguish the object and background. Wang *et al.* believe that an excellent detection algorithm helps improve the accuracy of the tracking algorithm [21]. There are many detection algorithms, one type of algorithm is focused on improving accuracy, and the other type is focused on speed. Due to the real-time requirements of object tracking, YOLO v1 algorithm [18] is a good detection method. They use frame object detection as a regression problem of spatially separated bounding boxes and related class probabilities. The rapid YOLO processing speed can reach 155 FPS. The paper also uses YOLO as the object detection algorithm.

Tracking-Learning-Detection(TLD) is the first tracking method that combines both online learning and detection algorithms together. In TLD, the tracking algorithm updates the object model and learning parameters through an online mechanism to solve the problem of object deformation during the tracking process [22]. In fact, under the TLD framework, tracking and detection algorithms always cooperate with each other. Many tracking algorithms based on deep learning follow this method. Chen *et al.* [23] propose a new framework SiamBAN, which add quality branches and state regression branches to the twin network framework to avoid the super parameters and prior knowledge of the candidate frames. Danelljan *et al.* [24] propose a Deep-SRDCF algorithm, which mainly uses the feature extraction capability of CNN, but the calculation of Deep-SRDCF algorithm is complicated and the cost is high. Nam and Han [25] propose an Multi-Domain Network(MDNet) algorithm that uses CNN feature sharing layers, which iteratively trains the CNN model for each domain to obtain the common object representation in the sharing layer. Wang *et al.* [26] propose the Structured Output Deep Learning Tracker(SO-DLT) algorithm, which train a CNN-based detector offline, and then use Stochastic Gradient Descent(SGD) to learn the detection algorithm during the tracking phase. Sun *et al.* [27] propose the Deep Affinity Network(DAN), which is an end-to-end network that combines appearance features and data association algorithm. Ullah *et al.* [28] propose a new tracking-by-detection method based on Bayesian filtering, which uses HoG descriptor to model the appearance of the object. These tracking algorithms use the powerful feature representation capabilities of CNN networks to improve the stability of tracking, but do not consider the historical position information of the object in consecutive frames.

B. TRACKING BY RNNs

As we all know, RNNs perform very well in processing sequence data, because they can store memories of previous states and establish temporal connections between them. Therefore, since the video frames are sequence data, it is very appropriate to use a recurrent neural network for visual tracking. Several works have studied this direction. Cui *et al.* [29] use the multi-directionality of recurrent neural networks to model and mine reliable object parts that are useful for the whole tracking. Fan and Ling [30] propose Structure-Aware Network(SANet), which uses multiple neural networks to model object structures at different levels. Kahou *et al.* [31] use a distinguishable attention mechanism to train the RNN to locate the object. Similarly, Gan *et al.* [15] train an RNN model to obtain the absolute position of the object in each frame. Ning *et al.* [17] propose a ROLO method, which combines a convolutional neural network and a recurrent neural network to predict the position of an object. Zhong *et al.* [32] propose a robust hierarchical tracker by combining recurrent neural network with correlation filtering. With the application of deep networks, RNN's ability to learn more complex tasks will be further improved.

III. BACKGROUND

In this section, we introduce relevant background of the study: (i) particle filter algorithm; (ii) long short-term memory networks.

A. PARTICLE FILTER ALGORITHM

The tracking process is considered to be a probability problem, which matches the object by estimating the posterior probability distribution. In the particle filter method, a set of samples (particles) are used to approximate the posterior probability distribution of the system, $b(h_t)$. and then the approximate representation is used to estimate the state of the nonlinear system [10].

$$b(h_t) \approx \left\{h_t^i, w_t^i\right\}_{i=1}^{K}, \tag{1}$$

where $\sum_K w_t^k = 1$, K is the number of particles, h_t is the particle state, which is latent state sampled from the same

distribution, w_t is the weight of the particles, and t represents time. The particle set can be approximated by any distribution, such as continuous, non-linear, non-Gaussian distribution, etc. The state estimate can be calculated by weighted average, and the particles are updated regularly in a Bayesian manner. The algorithm process is detailed in Algorithm 1.

Algorithm 1 Particle Filter

Input: Previous particle set: $b(h_{t-1})$; Last action: u_t; Current observation: o_t.

Output: Current particle set: $b(h_t)$; State estimate: $\bar{h}$.

1: **for** $i = 1$ TO K **do**
2: sample $h_t^i \sim f_{tr}(h_t|u_t, h_{t-1}^i)$
3: $w_t^i = \eta f_{obs}(o_t|h_t^i)w_{t-1}^i$
4: **end for**
5: **for** $i = 1$ TO K **do**
6: draw j with probability $\propto w_{t-1}^i$
7: $h'^i_t = h_t^j$
8: reset particle as $w'^i_t = 1/K$
9: **end for**
10: return to step1
11: state estimate: $\bar{h} = \sum_{i=1}^{K} w_t^i h_t^i$

B. LONG SHORT-TERM MEMORY NETWORKS

RNNs can solve sequence prediction problems, approximate beliefs as latent state vectors, learn directly from the data, and update beliefs with deterministic nonlinear functions. Object tracking extends it to multidimensional image processing tasks. However, when the image sequence is too long, the gradient of the original RNN network disappears due to back propagation errors, which may cause the original RNN to be unable to access the remote context. At the same time, the number of associated frames is variable, which may limit the performance of RNN in complex scenarios. In contrast, Long Short-Term Memory Network (LSTM) overcomes this problem and is able to learn context-independent and context-sensitive sequence information. The main idea of LSTM is to introduce an adaptive gate mechanism to determine what information to discard and what information to store. The LSTM framework is shown in Figure 2.

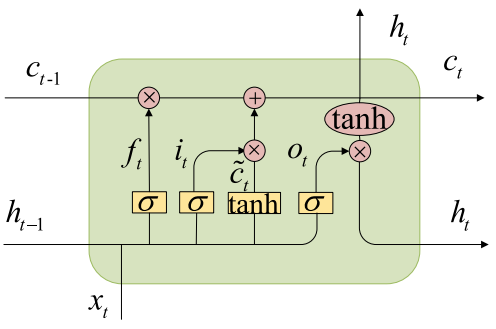


FIGURE 2. LSTM structure diagram.

A standard LSTM block consists of four parts: input gate i_t, forgotten gate f_t, output gate o_t and memory unit c_t. The memory state is updated by the deterministic function of each gate unit, and the relevant equation is derived as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \quad (2)$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \quad (3)$$
$$\tilde{c}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c), \quad (4)$$
$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t, \quad (5)$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \quad (6)$$
$$h_t = o_t * \tanh(c_t), \quad (7)$$

IV. DEEP PARTICLE FILTER TRACKER (DPFT)

The overview of our method is introduced in Section IV-A. Then, the detection module and tracking module are presented in detail in Sections IV-B and IV-C. Finally, the loss function of training the tracking network is described in Section IV-D.

A. OVERVIEW

Inspired by particle filter to approximate the posterior state distribution to a set of weighted particles, we propose a new network named LSTM-PF, which effectively improves its ability to process spatio-temporal information and infer regional position, and can solve the disappearance of short-term object problems, can better deal with the complexity of the real world. The architecture of our tracker DPFT is shown in Figure 3. Specifically, we first use a traditional CNN for regular feature learning training. CNN takes video frames as its input to generate a feature map of the whole image. Secondly, we use YOLO as the detection module, and use the visual features obtained by CNN as its input to get a preliminary location inference. Finally, the feature vector spliced from the feature vector and the frame position is used as the input of the LSTM-PF, and the latent state is updated by the Bayesian update strategy to obtain the final position of the object to be tracked. The resampling step is micronized, so that the gradient of the network can be back-propagated during training. The tracker is a combination of offline pre-training and online fine-tuning, which can solve the problem of less training data.

B. DETECTION MODULE

The TLD framework proves that the performance of a tracker depends in part on its excellent appearance characteristics [20]. In each frame of the video sequence, we choose YOLO to generate the detection results of the object. The first is to use traditional CNN to collect rich and robust image features. The convolution weights are pre-trained and learned through 1000 classes of ImageNet data. So that the network will have a good generalization ability. During training, the output of the first fully connected layer is a feature vector with a size of 4096, which is a dense representation of the visual features of the middle layer. In theory, feature vectors can be input into any classification tool (such as SVM or CNN), and good classification results can be obtained after training. After pre-training the CNN, on the convolutional

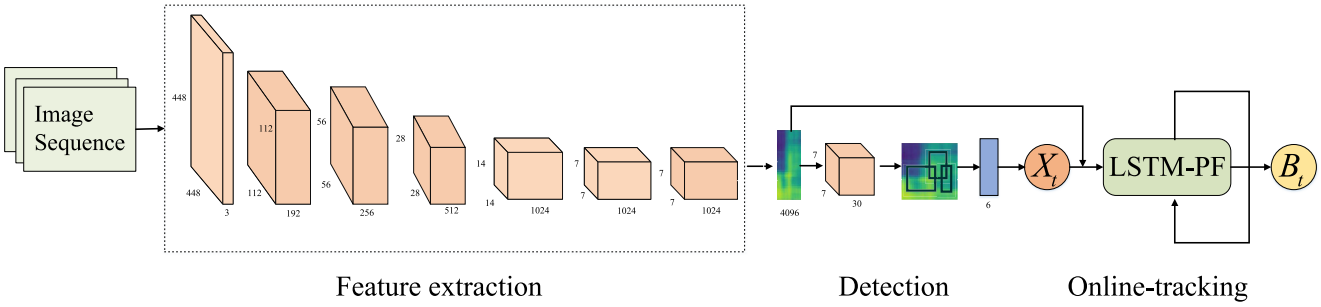


FIGURE 3. Framework of the proposed method.

layer, YOLO uses a fully connected layer to return the feature vector to the region border prediction, and these predictions are encoded into a tensor of $S \times S \times B \times 5 + C$ dimension. It indicates that the image is segmented into $S \times S$ grids, and each segmentation line has predicted B bounding boxes, which are represented by its five position parameters, x, y, w, h, and confidence c. A single thermal feature vector of length C is also predicted, representing the class label of each bounding box. In our framework, we follow the YOLO architecture and set $S = 7, B = 2, C = 20$. Each bounding box is initially composed of 6 predictions: x, y, w, h, class labels and confidence. For single object visual tracking tasks, class labels and confidence have little effect on it. Therefore, we eliminated the visual tracking category and confidence, and only locations are included in the evaluation.

$$B_t = (0, x, y, w, h, 0), \tag{8}$$

where (x, y) represents the coordinates of the center of the bounding box relative to the width and height of the image, (w, h) represents the width and height of the bounding box relative to the width and height of the image. For better regression, we should normalize all the output, which is $(x, y, w, h) \in [0, 1]$.

In a video frame, there may be multiple detection frames of the object generated by YOLO. When we want to assign the detection frame of the object tracked to the tracking network, an allocation cost matrix is used here. The matrix assignment is based on the IOU distance between the mean of the current detection and the short-term history verified detection results. The detection of the first frame is determined by the distance between the current detection and the ground truth IOU. When the IOU is less than a certain threshold value, the allocation is rejected, that is, it is not initialized. In the experiment, we set the threshold to 0.7. Finally, the correctly allocated object detection bounding box is connected with the appearance features generated by CNN, and these spliced feature vectors are input into LSTM-PF, and the object tracking problem is regarded as the regression problem of the object position coordinates.

C. TRACKING MODULE

The network overview of our tracking module is introduced in Section IV-C1. Then, the network architecture is shown in detail in Sections IV-C2. Finally, the online tracking algorithm is described in Section IV-C3.

1) NETWORK OVERVIEW

We regard the object tracking as a problem which predicts the position of the object to be tracked in the sequences of video. The ordinary sequence prediction problem is to predict the corresponding output sequence $y_1, y_2, \ldots, y_t$ given input sequence $x_1, x_2, \ldots, x_t$. The memory state of the standard LSTM is composed of the cell state c_t and the hidden state h_t. The latent state h_t is updated by deterministic nonlinear function learned from the data, which captures the historical information of input sequence. The predicted output y_t is another nonlinear function of the latent state h_t, which is also learned from the data. The memory state of our proposed network is composed of a set of weighted particles $\{h_t^i, c_t^i, w_t^i\}_{t=1}^K$, and is updated by the particle filter algorithm.In addition, the parameters of each particle are the same in LSTM-PF, so the number of particles does not influence the number of LSTM-PF network parameters. LSTM-PF uses the learning function to update the potential states, and the final prediction output y_t is obtained by using the average particle state: $y_t = f_{out}(\bar{h}_t)$ where $h_t = \sum_{i=1}^K w_t^i h_t^i$, and f_{out} is the prediction function of object tracking, which h_t will be mapped to the position of the object to be tracked in a video frame. In order to use the gradient method for effective training, we use a completely differentiable particle filter algorithm. The comparison between LSTM and LSTM-PF is shown in Figure 4. LSTM approximates the posterior probability distribution as a potential vector and uses a deterministic nonlinear function to update it. LSTM-PF approximates belief as a group of weighted particles and uses a random particle filter algorithm to update it.

2) NETWORK ARCHITECTURE

In order to improve the accuracy of object tracking, the network of our tracking module adopts the improved LSTM-PF, and its network structure is shown in Figure 5. There are two steps in the traditional particle filter to update the posterior state distribution: the transition update $\tilde{b}(h_t) = f_{tr}(b(h_{t-1}), u_t)$ for control u_t and the observation update $b(h_t) = f_{obs}(b(h_t), o_t)$ for observation o_t However, in the problem of video sequence prediction, u_t and o_t are not

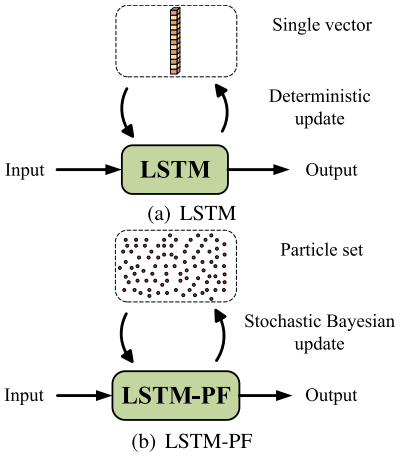


FIGURE 4. Comparison of LSTM and LSTM-PF.

separated from prior knowledge. LSTM-PF uses the input x_t in both f_{tr} and f_{obs} function, and extracts potential control vector u_t and measured value o_t from input vector x_t by task oriented discriminant training. This method is more simple and makes full use of the data-driven technology of neural networks.

Random memory update: To help LSTM-PF effectively track potential particle beliefs in the long-term history of the data, we make two changes to the memory update equation. One is to add randomness:

$$\tilde{c}_t^i = W_c \cdot [h_{t-1}^i, x_t] + b_c + \xi_t^i, \xi_t^i \sim N(0, \sum_t^i), \quad (9)$$

$$\sum_t^i = W_\Sigma \cdot [h_{t-1}^i, x_t] + b_\Sigma, \quad (10)$$

where x_t is the joint vector of the bounding box coordinate position and image features obtained by the detection module, ξ_t^i is a learning noise. From the perspective of RNN, ξ_t^i capture potential stochastic transformation dynamics. From the point of view of particle filtering, ξ_t^i increases particle diversity and alleviates particle degeneracy after resampling. Another change is inspired by LiGRU [33], using ReLU activation and batch normalization [34] instead of LSTM hyperbolic tangent activation,

$$c_t^i = f_t^i * c_{t-1}^i + i_t^i * ReLU(BN(\tilde{c}_t^i)), \quad (11)$$

where $i = 1, 2, \cdots, K$. The back propagation algorithm is usually used to calculate the gradient, but back propagation can only be achieved after the RNN is deployed in time. Therefore, the back propagation algorithm used to train the RNN is often called the Back Propagation Through Time (BPTT) [35]. Gradient truncation has a great influence on the training of LSTM-PF, because the network explicitly retains the latent state, and may require a long input sequence to approximate the beliefs well. As shown in [33], ReLU activation combined with batch normalization has good numerical characteristics and can be back-propagated through multiple time steps. Therefore, when the sequence length is relatively large, the truncated BPTT algorithm can be used. The update of h_t^i can be calculated according to Equation 7.

Particle weight update: We use Bayesian method to recursively update the weight of particle h_t^i. In the particle filtering algorithm, the particle weights are updated by the likelihood probability $p(o_t|h_t^i)$ as a generated distribution. In LSTM-PF, we directly approximate it to a learning function $f_{obs}(o_t|h_t^i)$ to update particle weights.

$$w_t^i = \eta f_{obs}(o_t|h_t^i) w_{t-1}^i, \quad (12)$$

where $\eta^{-1} = \sum_{i=1:K} f_t^i w_{t-1}^i$ is a normalization factor. First use the joint vector of the boundary frame coordinate position of the previous video frame and the feature representation as input to obtain the predicted position of the current frame object, and then use the detection frame position of the current frame as the measurement result to correct the predicted object position.

Soft resampling: After updating the particles and their weights, we perform micro-sampling to get a new set of particles. In the particle filtering algorithm, re-sampling is needed to avoid particle degeneracy, because after iteration, the weight of most particles is close to zero. However, resampling is not differentiable, which prevents the use of back propagation to train LSTM-PF. In order to make our latent belief update differentiable, we use soft resampling [36]. Instead of resampling particles based on p, we resample from q.

$$q(i) = \beta w_t^i + (1 - \beta)(1/K), \quad (13)$$

where $\beta \in (0, 1]$. The new weight calculation is based on the importance sampling equation, which leads to an unbiased estimated belief,

$$w_t^i = \frac{p(i = a^j)}{q(i = a^j)} = \frac{w_t^{a^j}}{\beta w_t^{a^j} + (1 - \beta)(1/K)}, \quad (14)$$

Soft resampling provides a non-zero gradient $\beta > 0$. In our experiments, we use $\beta = 0.5$.

3) ONLINE TRACKING ALGORITHM

Our DPFT framework adopts a simple online tracking strategy, which encodes the historical data extracted from the video frame into the memory unit, and encodes the hidden state into the estimated state when passing through the network. The online tracking algorithm is detailed in the Algorithm 2. Because the training data of object tracking is very limited. Firstly, we use the auxiliary ImageNet dataset for pre-training to obtain the general representation of the object features. In the tracking process, we use the limited sample information of the current tracking object to fine-tune the LSTM-PF network to improve the performance of the tracking algorithm. This idea of migration learning greatly reduces the demand for training samples of tracking objects [37].

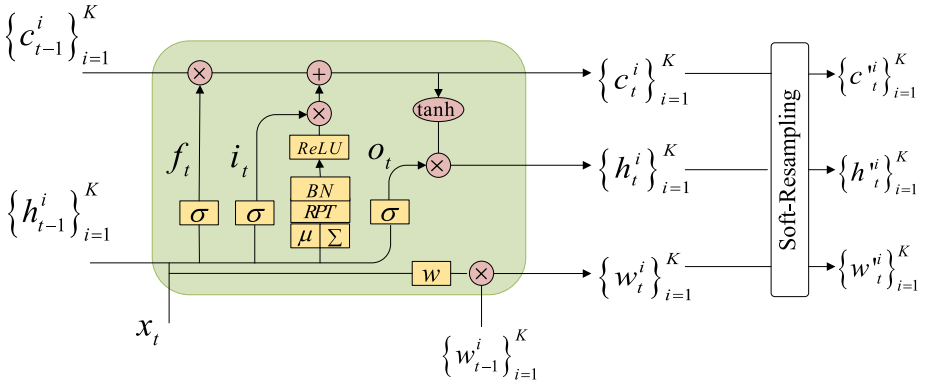


FIGURE 5. LSTM-PF network structure diagram.Notation (1) RPT:reparameterization trick (2) BN:batch normalization.

Algorithm 2 Online Tracking Algorithm With LSTM-PF

Input: Joint vector of feature representation and bounding box position: x_t; Status information of the previous frame: h_{t-1}.

Output: Predict the bounding box of the object.

1: predict the status information of the current frame
2: update the weight of the LSTM-PF memory unit according to the detection information of the current frame
3: use soft resampling to get a new set of particles
4: continue tracking and return to step 1
5: input the final current frame state information h_t into the softmax layer for regression
6: output the corresponding bounding box coordinates

D. LOSS FUNCTION

The tracking module uses a LSTM-PF network. There are two types of incoming data streams, one is the joint vector X_t of the feature representation of the convolutional layer and the detection information of the fully connected layer, and the other is the state output of previous time step h_{t-1}. The final output is the predicted object location B_t which is a real value [17]. We use the Mean Square Error (MSE) to train the network:

$$Loss = \frac{1}{n}\sum_{i=1}^{n} ||B_{true} - B_{pred}||_2^2, \tag{15}$$

where n is the number of training samples, B_{pred} is the prediction of the model, B_{true} is the object ground truth value, $||\cdot||$ is the square of the Euclidean norm, and we use the Adam method for random optimization. For all models, we perform a standard grid search on training parameters: learning rate, batch size, and gradient clipping value.

V. EXPERIMENTAL RESULTS

In this section, our tracker is compared with other state-of-the-art trackers. Experiments are conducted on OTB-100 and VOT-2016 two datasets, and the relevant introduction of the datasets are shown in Section V-A. Experimental metric and implementation details are presented in Sections V-B and V-C. The experimental results on OTB benchmarks and VOT challenge are described in Sections V-D and V-E. Ablation study and runtime about the proposed tracker are shown in Sections V-F and V-G.

A. DATASETS

In order to verify the versatility and effectiveness of the algorithm, we select OTB-100 [38] and VOT-2016 [39] two benchmark datasets for extensive experiments, these two datasets cover many challenging scenarios, Such as object deformation, local occlusion, scale change and rapid movement. The OTB-100 dataset contains video sequences of 100 different objects, such as animals, cars, toys, and people, etc., and the attributes of object deformation, local occlusion, and fast motion are manually marked in the sequence. We select 30 video sequences with special visual challenges from the OTB-100 benchmark to evaluate the performance of the proposed framework. VOT-2016 is composed of 60 challenging short video sequences of different scenes. Some examples of these two datasets are shown in Figure 6.

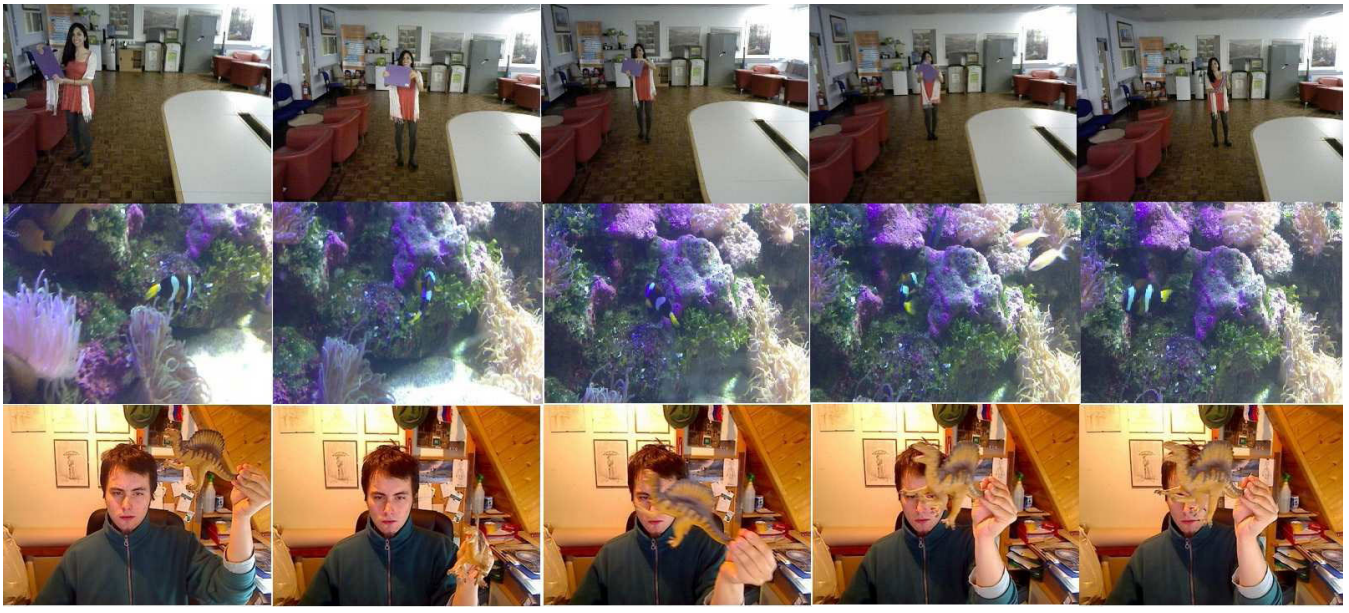
B. EXPERIMENTAL METRIC

There are usually two evaluation indicators for the OTB dataset: accuracy map and success rate. The accuracy map uses the Central Location Error (CLE) to measure the positioning accuracy. CLE is the measurement of the discrete pixel error between the predicted value and the ground truth center. The accuracy graph shows that the distance between the center point of the object position estimated by the tracker and the manually labeled object center point is less than the percentage of the video frame with the given threshold. We set the threshold to 20 pixel. The success rate is evaluated using the bounding box overlap rate. The calculation equation is as follows:

$$OR = \frac{area(B_{pre} \cap B_{true})}{area(B_{pre} \cup B_{true})}, \quad OR \in [0, 1], \tag{16}$$

where OR is the coincidence rate between the object tracking result B_{pre} and the ground truth B_{true}. When the coincidence rate of a frame is greater than the set threshold, the frame is considered to be successful. The total number of successful

(a) Samples of the OTB dataset are Dog, Woman and Car



(b) Samples of the VOT2016 dataset are Book, Fish and Dinosaur

FIGURE 6. Some examples of two datasets.

frames as a percentage of all frames is the success rate. We set this threshold to 0.5.

We use the One Pass Evaluation (OPE) to express the average success rate and accuracy. However, the tracker may be very sensitive to initialization, with different initializations at different start frames. Therefore, we also use Temporal Robustness Evaluation (TRE) and Spatial Robustness Evaluation (SRE) to represent the success rate of the object state.

There are usually three evaluation indicators for VOT dataset: accuracy, robustness, and Expected Averaged Overlap (EAO). Accuracy refers to the average overlap rate of the tracker under a single test sequence. Robustness is used to evaluate the stability of the tracker's tracking object, that is, the number of times the tracker needs to be reinitialized when the tracker loses (or drifts) the visible object. EAO is the expected value of the non-reset overlap of each tracker on a short-time image sequence. Given a video sequence of length N_s, the EAO calculation equation is:

$$\hat{\phi}_{N_s} = \frac{1}{N} \sum_{i=1}^{N_s} \phi_{N_i}, \tag{17}$$

FIGURE 7. Visual results of OTB dataset samples, Dog, Woman, Car.

where ϕ_{N_i} expressed as the overlap rate of the tracker at frame i of this video.

C. IMPLEMENTATION DETAILS

The YOLO used by our tracker is trained on ImageNet data of 1000 classes, and fine-tuning on the VOC dataset can detect 20 types of objects. We select a subset of 30 videos from the benchmark, and the objects belong to these categories. First, we follow the YOLO architecture, with a total of 32 layers. In the second stage, we combine detection with LSTM-PF network trained with benchmark OTB and VOT for fine tuning. The training is carried out as the epoch gradually increases, so when the iteration shows the best tracking performance, we need to terminate the iteration.

From the OTB-100 benchmark, we select a set of 30 sequence videos with special visual challenges, such as object deformation, lighting changes, scale changes, local occlusion, and fast motion. We use 20 sequences to train the LSTM-PF model and tested with another 10 sequences. And after our experiments, it is found that the number of sequence frames (step size) of the input network has a great impact on the overall performance and running time. We set the step size to 6 in the experiment, and the algorithm performance is the best. Our training rate is set to 0.0001, the dropout is 0.5, and the number of particles is 30. Our method is implemented in Python using TensorFlow, and test it on a computer with NVIDIA GeForce GTX 1050 Ti.

D. TRACKING RESULTS ON OTB100

In order to verify the effectiveness and robustness of our proposed method for tracking objects, we use 3 benchmark trackers and 4 advanced trackers to compare with our tracking methods, which are OAB [40] and TLD [22], STRUCK [41], SiamFC [42], ROLO [17], YOLO+SORT [43] and KCF [44] are on the OTB100. The visual tracking results are shown in Figure 7, where the object in Dog and Car video sequences has obviously changed in scale. Compared with other tracking algorithms, our tracker can accurately track the object, and the size of the tracking frame changes adaptively with the change of object scale. However, TLD and STRUCK trackers cannot update the size of the object annotation frame in time, which leads to poor object tracking effect. It can be seen that the TLD has a tracking deviation at frame 252 of the Car video sequence. In the Woman video sequence, the object is partially occluded. From the video test results, it can be seen that the TLD and STRUCK trackers can only detect the part that is not blocked at frame 117, and even the phenomenon of tracking loss occurs in TLD at frame 163. The results show that our tracker performs well on the OTB-100 dataset and has good robustness, because our algorithm extracts deep abstract features from the spatiotemporal background, which can effectively solve the occlusion problem.

We show the OPE accuracy graph and success graph in Figure 8, where our method is expressed as DPFT. The tracker is ranked according to its area under the curve (AUC) score. From the test results, AUC of our proposed tracker is slightly higher, the accuracy score is 0.821, compared with the second SiamFC score of 0.725, increased by 13.24%, the success rate score is 0.623, compared with the second place increased by 9.3%. The SRE and TRE results are shown in Figure 9 and are used for robustness evaluation. The results show that we approximate the posterior state distribution of the long

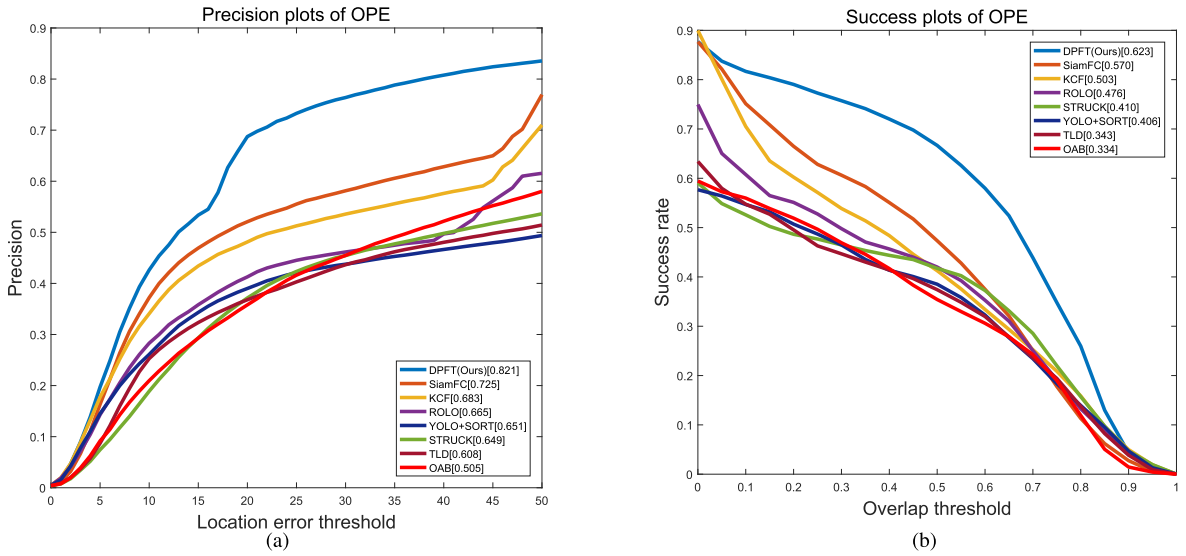


FIGURE 8. Accuracy graph and success graph of OPE (One Pass Evaluation) on the OTB benchmark.

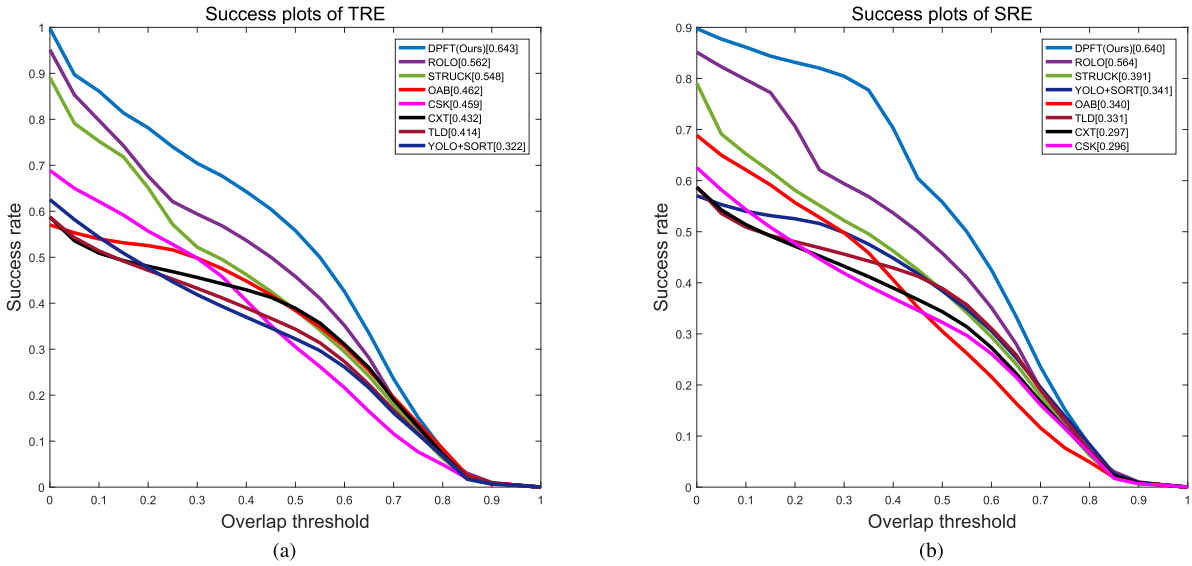


FIGURE 9. TRE (Temporal Robustness Evaluation) and SRE (Spatial Robustness Evaluation) success graphs on the OTB benchmark.

short-term memory network as a set of particles, and use random particle filtering algorithm to update the memory cells, fully exploit the potential of the cyclic neural network to deal with the uncertainty of moving objects, and improve the estimation accuracy.

E. TRACKING RESULTS ON VOT2016

We also test the versatility of our proposed method on the VOT2016 dataset. According to the VOT challenge protocol [36], when tracking failure is detected, the tracker will re-initialize. We compare our method with 9 state-of-the-art trackers, including SiamRPN [45], ASRCF [46], MAM [47], SiamFC [42], KCF [44], ROLO [17], TLD [22], RFL [48], DSST [49], the visual tracking results are shown in Figure 10. In the Book video sequence, the object mainly produces scale changes. The fish video sequence object is mainly the interference of similar background. In the Dinosaur video sequence, the object clearly appears motion blur. Experimental results show that when YOLO detection error occurs due to motion blur, the DPFT tracking results remain stable. In addition, LSTM considers historical position information, which is different from traditional time correction methods (such as Kalman filtering, whose prediction is only based on the previous position). When dealing with occlusion, the location of occluded object can be determined better by using spatio-temporal information, and the drift phenomenon can

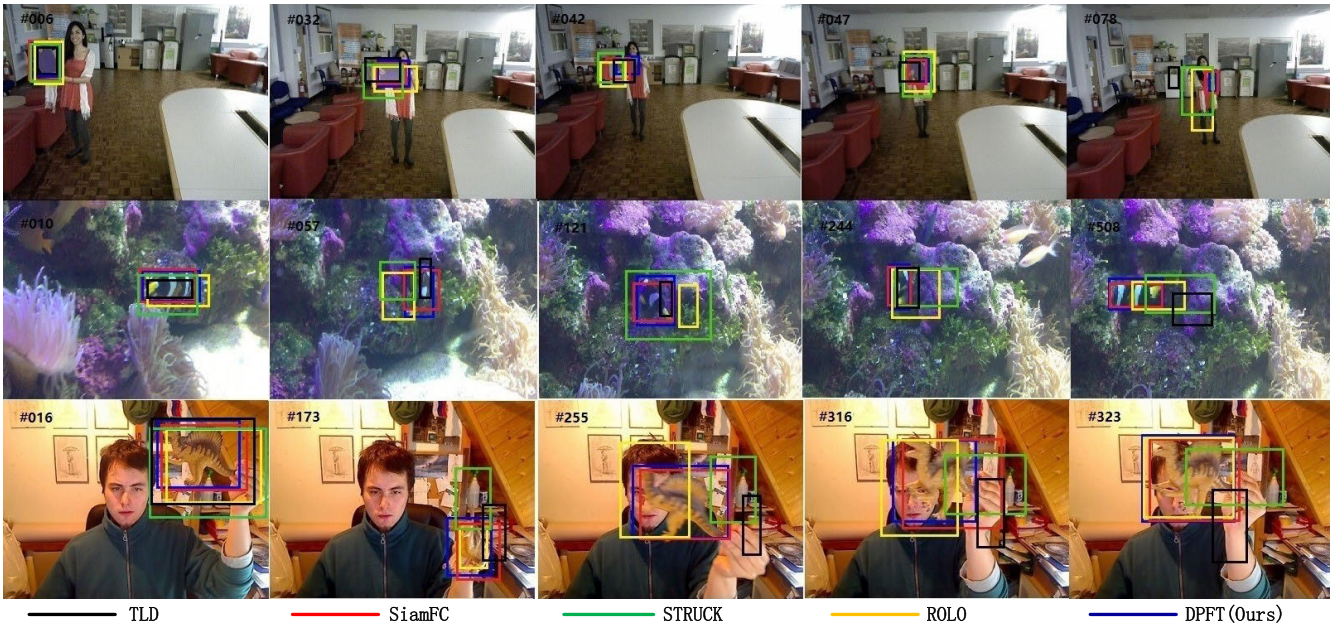


FIGURE 10. The visualization results of the VOT dataset samples, namely Book, Fish, Dinosaur.

be reduced. In the case of scale changes, motion blur, similar background and occlusion, the algorithm can obtain more robust and accurate tracking results.

On this basis, the three performance indicators of accuracy, robustness and expected average overlap (EAO) are used to evaluate the algorithm. Different trackers are sorted according to EAO criterion, we invite readers to [36] for more detailed information. We use the "VOT-toolkit" to conduct a comprehensive analysis of these results. Table 1 summarizes our analysis results. The second and third rows contain the accuracy and robustness of the entire dataset, and the last column is the EAO of each tracker. The best results are shown in red,the second best and the third best results are shown in green and blue respectively. Compared with other state-of-the-art trackers, accuracy of our proposed tracker is the highest, the accuracy score is 0.568, compared with the second ASRCF score of 0.563, increased by 0.9%. Although our robustness and EAO performance are not as good as ASRCF, our tracker has reached the best accuracy level. The closer the tracker to the upper right corner, the better the performance. This is also verified in Figure 11.

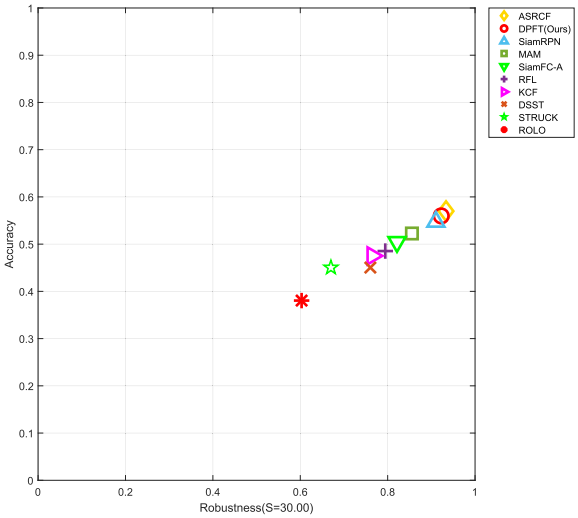


FIGURE 11. Accuracy-Robustness plot on VOT2016.

F. ABLATION STUDY

In order to verify the effectiveness of the proposed LSTM-PF in object tracking, we conduct a thorough ablation study on the OTB-100 dataset. We use ROLO as a baseline to achieve and evaluate the contributions of the following components.

Number of particles: The network employs a set of weighted particles to approximate the potential state. The number of particles directly determines the size of network memory. We set the number of particles to P1, P5, P10, P20, P30, in order to search for the best latent state size and get the best result. The result is shown in Figure 12a. The result shows that performance of our LSTM variant is better than that of the standard LSTM, and the AUC of ROLO is the lowest,which shows that LSTM-PF approximates beliefs as a set of weighted particles, and trains approximate representations from data to optimize regression performance. Secondly, we can also find that as the number of particles increases, the accuracy score increases gradually. When the number of particles is 30, the tracker has the highest accuracy with a score of 0.623.

Soft resampling: In order to illustrate the advantages of our method that the resampling steps can be miniaturized,

TABLE 1. Accuracy (A), robustness (R) and expected average overlap (EAO) scores of different trackers on VOT-2016.

Trackers	Accuracy	Robustness	EAO
DPFT(Ours)	0.568	0.283	0.341
SiamRPN	0.56	0.26	0.34
SiamFC-A	0.532	0.461	0.235
MAM	0.50	0.74	0.27
ASRCF	0.563	0.187	0.391
KCF	0.489	0.569	0.192
STRUCK	0.458	0.942	0.142
ROLO	0.412	0.923	0.131
DSST	0.533	0.704	0.181
RFL	0.503	0.801	0.222

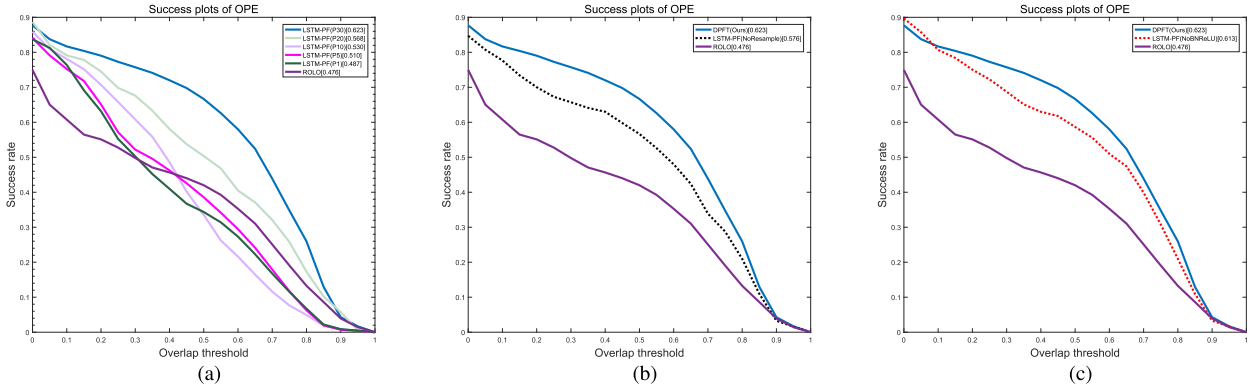


FIGURE 12. Ablation study on OTB-100.

TABLE 2. Running time.Runtime is the processing time of forward propagation of an image.The unit is fps.

Trackers	Struck	ROLO	KCF	SiamFC	TCNN	MDNet	DPFT(Ours)
Runtime(ms)	21.4	35	172	58	1.5	1	49

we designed a variation of LSTM-PF (NoResample) for comparative experiments. The results are shown in Figure 12b. It can be seen from the test results that soft resampling improves the performance of the tracker.

ReLU activation and batch normalization: To test the results of using the combination of ReLU activation and batch normalization instead of hyperbolic tangent, we implemented a comparative experiment with the LSTM-PF (NoBNReLU) proposed by Ma *et al.* [50]. The results are shown in Figure 12c. The results show that batch normalized ReLU activation is helpful for training LSTM-PF.

G. RUNTIME

A good tracker not only needs to achieve good accuracy, but also needs good tracking performance and real-time tracking speed. We compared the running time of 6 trackers, including Struck [41], ROLO [17], KCF [44], SiamFC [32], TCNN [51] and MDNet [25] are compared, and the results are shown in Table 2. As can be seen from Table 2, although the tracker proposed by us is not the fastest tracker, our tracker can not only ensure the tracking accuracy, but also track with the real-time speed. The frame rate of our improved method is 49 fps, which is 40% faster than ROLO.

VI. CONCLUSION AND FUTURE WORK

The purpose of this study is to propose a visual tracking framework for long short-term memory networks based on particle filtering, called DPFT. In essence, it is based on the tracking-by-detection methonds. The main idea is to obtain the position of the initial frame through the detection algorithm, and then use the improved LSTM for a post-processing, and use the particle filter to update the posterior state probability of the object position to make the tracking more robust. We conduct detailed experimental evaluations on two challenging large datasets. The accuracy of OTB100 is 0.821, compared with the state of the art tracker increased by 13.24%, the accuracy score on the VOT2016 dataset is 0.568, compared with the state of the art tracker increased by 0.9%. Both qualitative and quantitative experimental results prove that our proposed tracker can perform well in most cases.

In future work, we will propose improvements to these two aspects of research. One is to use a more accurate detection method instead of YOLO, and the detection tracking method is largely affected by the detection accuracy. When tracking, the accuracy of selecting a bounding box from YOLO is not high, which may lead to the wrong selection of bounding box among multiple detection frames. The second is to explore

data correlation technology and expand DPFT to the field of multi-object tracking.

REFERENCES

[1] J.-H. Huh, "PLC-based design of monitoring system for ICT-integrated vertical fish farm," *Hum.-Centric Comput. Inf. Sci.*, vol. 7, no. 1, pp. 1–19, Jul. 2017.

[2] J. Chang, S. Hong, D. Son, H. Yoo, and H. Ahn, "Development of real-time video surveillance system using the intelligent behavior recognition technique," *J. Inst. Internet, Broadcast. Commun.*, vol. 19, no. 2, pp. 161–168, Apr. 2019.

[3] Q. Zhou, B. Zhong, X. Lan, G. Sun, Y. Zhang, B. Zhang, and R. Ji, "Fine-grained spatial alignment model for person re-identification with focal triplet loss," *IEEE Trans. Image Process.*, vol. 29, pp. 7578–7589, Jun. 2020.

[4] A. Caraban, E. Karapanos, D. Gonçalves, and P. Campos, "23 ways to nudge: A review of technology-mediated nudging in human-computer interaction," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2019, pp. 1–15.

[5] Z. Wang, "SEG-YOLO: Real-time instance segmentation using YOLOv3 and fully convolutional network," M.S. thesis, School Electron. Comput. Sci., KTH Roy. Inst. Technol., Stockholm, Sweden, 2019.

[6] Q. Zhou, B. Zhong, Y. Zhang, J. Li, and Y. Fu, "Deep alignment network based multi-person tracking with occlusion and motion reasoning," *IEEE Trans. Multimedia*, vol. 21, no. 5, pp. 1183–1194, May 2019.

[7] D. Frossard and R. Urtasun, "End-to-end learning of multi-sensor 3D tracking by detection," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 635–642.

[8] S. Mojtaba Marvasti-Zadeh, L. Cheng, H. Ghanei-Yakhdan, and S. Kasaei, "Deep learning for visual tracking: A comprehensive survey," 2019, *arXiv:1912.00535*. [Online]. Available: http://arxiv.org/abs/1912.00535

[9] M. E. Yildirim, I. F. Ince, Y. B. Salman, J. K. Song, J. S. Park, and B. W. Yoon, "Direction-based modified particle filter for vehicle tracking," *ETRI J.*, vol. 38, no. 2, pp. 356–365, Apr. 2016.

[10] M. Nieto, A. Cortés, O. Otaegui, J. Arróspide, and L. Salgado, "Real-time lane tracking using rao-blackwellized particle filter," *J. Real-Time Image Process.*, vol. 11, no. 1, pp. 179–191, Jan. 2016.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[12] Y. Xie, J. Shen, X. Han, and C. Wu, "Convolutional features combining SL(3) group for visual tracking," *IEEE Access*, vol. 8, pp. 61096–61106, 2020.

[13] M. Ullah and F. A. Cheikh, "A directed sparse graphical model for multi-target tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1816–1823.

[14] K. Zhang, Q. Liu, Y. Wu, and M.-H. Yang, "Robust visual tracking via convolutional networks without training," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1779–1792, Apr. 2016.

[15] Q. Gan, Q. Guo, Z. Zhang, and K. Cho, "First step toward model free, anonymous object tracking with recurrent neural networks," 2015, *arXiv:1511.06425*. [Online]. Available: https://arxiv.org/abs/1511.06425

[16] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A critical review of recurrent neural networks for sequence learning," 2015, *arXiv:1506.00019*. [Online]. Available: https://arxiv.org/abs/1506.00019

[17] G. Ning, Z. Zhang, C. Huang, X. Ren, H. Wang, C. Cai, and Z. He, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[20] S. P. Bharati, S. Nandi, Y. Wu, Y. Sui, and G. Wang, "Fast and robust object tracking with adaptive detection," in *Proc. IEEE 28th Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2016, pp. 706–713.

[21] N. Wang, J. Shi, D.-Y. Yeung, and J. Jia, "Understanding and diagnosing visual tracking systems," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3101–3109.

[22] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.

[23] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6668–6677.

[24] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshop (ICCVW)*, Dec. 2015, pp. 58–66.

[25] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4293–4302.

[26] N. Wang, S. Li, A. Gupta, and D.-Y. Yeung, "Transferring rich feature hierarchies for robust visual tracking," 2015, *arXiv:1501.04587*. [Online]. Available: http://arxiv.org/abs/1501.04587

[27] S. Sun, N. Akhtar, H. Song, A. S. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jul. 19, 2019, doi: 10.1109/TPAMI.2019.2929520.

[28] M. Ullah, F. A. Cheikh, and A. S. Imran, "HoG based real-time multi-target tracking in Bayesian framework," in *Proc. 13th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2016, pp. 416–422.

[29] Z. Cui, S. Xiao, J. Feng, and S. Yan, "Recurrently target-attending tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1449–1458.

[30] H. Fan and H. Ling, "SANet: Structure-aware network for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 42–49.

[31] S. E. Kahou, V. Michalski, R. Memisevic, C. Pal, and P. Vincent, "RATM: Recurrent attentive tracking model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1613–1622.

[32] B. Zhong, B. Bai, J. Li, Y. Zhang, and Y. Fu, "Hierarchical tracking by reinforcement learning-based searching and coarse-to-fine verifying," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2331–2341, May 2019.

[33] M. Ravanelli, P. Brakel, M. Omologo, and Y. Bengio, "Light gated recurrent units for speech recognition," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 2, pp. 92–102, Apr. 2018.

[34] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2015, pp. 448–456.

[35] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[36] P. Karkus, D. Hsu, and W. S. Lee, "Particle filter networks with application to visual localization," in *Proc. Conf. Robot Learn.*, Oct. 2018, pp. 169–178.

[37] N. Wang and D. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. Adv. Neural Inf. Process. Syst*, 2013, pp. 809–817.

[38] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.

[39] S. J. Hadfield, R. Bowden, and K. Lebeda, "The visual object tracking VOT2016 challenge results," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 9914, 2016, pp. 777–823.

[40] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. Brit. Mach. Vis. Conf.*, 2006, p. 6.

[41] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. S. Torr, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.

[42] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis.* Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 850–865.

[43] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3464–3468.

[44] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[45] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8971–8980.

[46] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4670–4679.

[47] B. Chen, P. Li, C. Sun, D. Wang, G. Yang, and H. Lu, "Multi attention module for visual tracking," *Pattern Recognit.*, vol. 87, pp. 80–93, Mar. 2019.

[48] T. Yang and A. B. Chan, "Recurrent filter learning for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 2010–2019.

[49] M. Danelljan, G. Häger, F. Shahbaz Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–5.
[50] X. Ma, P. Karkus, D. Hsu, and W. S. Lee, "PF-LSTM: Belief state particle filter for LSTM," in *Proc. NeurIPS RLPO Workshop*, 2018. [Online]. Available: https://yusufma03.github.io/publication/ma-2018-pflstm
[51] H. Nam, M. Baek, and B. Han, "Modeling and propagating CNNs in a tree structure for visual tracking," 2016, *arXiv:1608.07242*. [Online]. Available: http://arxiv.org/abs/1608.07242

YANLI LIU was born in 1979. She received the M.S. degree in computer application technology and the Ph.D. degree from Central South University, China, in 2005 and 2014, respectively. She is currently a Professor with Shanghai Dianji University. She is the author or coauthor of more than 40 articles. Her research interests include computer vision, mobile robot autonomous navigation, and cloud robotics.

JINGJING CHENG was born in 1996. She is currently pursuing the master's degree under the supervision of Prof. Y. Liu. Her research interests include image processing, deep learning, and intelligent robotics.

HENG ZHANG was born in 1979. He received the B.S. degree in computer science and technology and the Ph.D. degree from Central South University, China, in 2001 and 2007, respectively. He is currently a Professor with Shanghai Dianji University. He is the author or coauthor of more than 50 articles. His research interests include mobile robot autonomous navigation, computer vision, and cloud robotics.

HANG ZOU was born in 1995. He is currently pursuing the master's degree with the Wuhan Research Institute of Posts and Telecommunications. His research interests include image processing, deep learning, and intelligent robotics.

NAIXUE XIONG (Senior Member, IEEE) received the Ph.D. degree in software engineering from Wuhan University and the Ph.D. degree in dependable networks from the Japan Advanced Institute of Science and Technology. He was a Full Professor with Colorado Technical University for a period of four years, an Assistant Professor with the Wentworth Technology Institution for a period of one year, and an Assistant Professor and held a postdoctoral position with Georgia State University for a period of four years. He is currently with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK, USA. He published over 100 international journal articles and over 100 international conference papers. Some of his works were published in the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE or ACM TRANSACTIONS, ACM Sigcomm Workshop, the IEEE INFOCOM, ICDCS, and IPDPS. His research interests include cloud computing, business networks, security and dependability, parallel and distributed computing, and optimization theory. He has been a PC member, an OC member, a General Chair, a Program Chair, and a Publicity Chair of over 100 international conferences. He is a Senior Member of the IEEE Computer Society. He has received the Best Paper Award in the 10th IEEE International Conference on High Performance Computing and Communications 2008 and the Best Student Paper Award in the 28th North American Fuzzy Information Processing Society Annual Conference 2009. He is the Chair of Trusted Cloud Computing Task Force, the IEEE Computational Intelligence Society, and the Industry System Applications Technical Committee. He was serving as the Editor-in-Chief, an Associate Editor, or an Editor member for over 10 international journals, including an Associate Editor for the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS SYSTEMS, an Associate Editor for *Information Science*, an Editor-in-Chief for the *Journal of Internet Technology*, and an Editor-in-Chief for the *Journal of Parallel and Cloud Computing*, and a Guest Editor for over ten international journals, including the IEEE WIRELESS COMMUNICATIONS, the IEEE SENSORS JOURNAL, *WINET*, and *MONET*.