Visual Tracking via Hierarchical Deep Reinforcement Learning

Dawei Zhang[1], **Zhonglong Zheng**[1]*, **Riheng Jia**[1], **Minglu Li**[1,2]
[1] College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua, China
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
{davidzhang, zhonglong, rihengjia, mlli}@zjnu.edu.cn

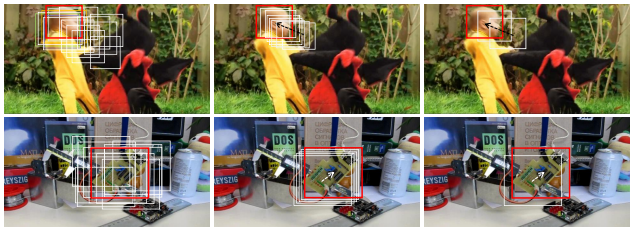
Abstract

Visual tracking has achieved great progress due to numerous different algorithms. However, deep trackers based on classification or Siamese network still have their specific limitations. In this work, we show how to teach machines to track a generic object in videos like humans, who can use a few search steps to perform tracking. By constructing a Markov decision process in Deep Reinforcement Learning (DRL), our agents can learn to determine hierarchical decisions on tracking mode and motion estimation. To be specific, our Hierarchical DRL framework is composed of a Siamese-based observation network which models the motion information of an arbitrary target, a policy network for mode switch and an actor-critic network for box regression. This tracking strategy is more in line with human behavior paradigm, and is effective and efficient to cope with fast motion, background clutter and large deformations. Extensive experiments on the GOT-10k, OTB-100, UAV-123, VOT and LaSOT tracking benchmarks, demonstrate that the proposed tracker achieves state-of-the-art performance while running in real-time.

Introduction

Object tracking is one of the fundamental vision tasks in the artificial intelligence field. Generally speaking, a model-free tracker is aimed to localize an arbitrary object in a video sequence, given its initial annotation in the first frame. Even though much effort has been made recently, there are still many challenges including illumination variations, deformation, motion blur, occlusions, and abrupt motion, to name a few. Meanwhile, many practical applications of visual tracking, such as video surveillance, autonomous driving, robotics and human-computer interaction, require a high performance tracker with a real-time constraint.

Recently, tracking algorithms (Nam and Han 2016; Danelljan et al. 2017; Li et al. 2019) base on deep learning have obviously improved the tracking performance. Generally, the pre-trained Convolutional Neural Networks (CNNs) are capable to extract rich deep features. Thus numerous deep trackers utilize the CNNs pre-trained on ImageNet (Russakovsky et al. 2015) to perform robust online tracking. Through top-ranked performance achieved by MDNet (Nam and Han 2016), it only runs 1 fps due to an inefficient search strategy that samples 256 candidates randomly like particle filter framework, and selects the best candidate by verifying each sample with the binary classification model. To alleviate this issue, an novel action-decision network (ADNet) (Yun et al. 2017) based on DRL is proposed to generate a series of discrete actions to adjust the aspect ratio and center position of the target in each frame. The searching steps of this scheme are much fewer than sliding window (Tao, Gavves, and Smeulders 2016) and random sampling (Nam and Han 2016) approaches. ADNet obtains slightly worse but three times faster performance than MDNet. Furthermore, Actor-Critic tracking framework (Chen et al. 2018) is developed to predict only one continuous action to locate the tracked object. However, these tracking methods (Chen et al. 2018; Dunnhofer et al. 2019) with one search step can not effectively capture all possible motion variations of the target of interest in some complex scenes.



(a) MDNet (b) ADNet (c) Ours

Figure 1: The different strategies of three tracking algorithms. (a) MDNet (Nam and Han 2016): classify with random sampling; (b) ADNet (Yun et al. 2017): iterative search with a series of discrete actions; and (c) our tracker: DRL-based hierarchical search with less continuous actions.

Similar to the discussion in (Ren et al. 2018), an intelligent tracking algorithm is required to understand the motion information of the target and quality of the current observation state, and make optimal decisions online at the right moments, i.e., whether to continue or stop searching or even reinitialize tracking if necessary. To this end, a straightforward approach is to learn a policy network that can select the optimal action during tracking by deep reinforcement learning. Besides, another overlooked point for visual object tracking is the limited use of the temporal information in

*Corresponding author.

videos, which provides an important clue to make the tracker resistant to changes in target appearances.

To address these issues mentioned above, we regard the tracking problem as a hierarchical decision-making process, and introduce a Hierarchical DRL (HDRL) framework consisting of a Siamese-based observation network, a policy agent to decide the tracking pattern of the current state and an actor-critic agent to conduct searching procedure in a continuous action space. Compared with existing trackers which employed RL to either estimate motion or make decision on tracking status separately, we take both into account. To the best of our knowledge, this work is the first attempt to exploit multi-agent system for visual tracking. As shown in Figure 1, our approach makes hierarchical decisions and searches the tracked object with few steps (e.g., 2 steps for the face and 1 step for the box). Thus the resulting tracking process tends to be more efficient and effective.

In general, complex tracking procedures with multiple iterations and online update are time-consuming. Recently, behavior demonstrations of an expert tracker (Dunnhofer et al. 2019) can be used to guide the tracking agent, which obtains better and faster performance. Based on this, we further take advantage of the expert tracker to simplify and guide the procedures of update and re-initialization during online tracking. Benefiting from this scheme, our HDRL framework avoids online fine-tuning or updating of the network's weights, and ensures tracking efficiency.

The main contributions of this paper are summarized as:

- This work is the first attempt to exploit hierarchical deep reinforcement learning framework for visual tracking, which is modeled as a dynamic iterative search process where is performed by a Policy agent and an Actor-Critic Network in collaboration. So we named it PACNet.
- We propose a novel policy network to make decisions on the tracking status, where the rewards for different actions are dedicatedly designed according to the current target's motion information impacted by the Actor-Critic agent.
- To ensure tracking efficiency, we introduce an expert system to guide the online update and re-initialization of our model, which improves the robustness of PACNet.
- The proposed tracking method is compared with state-of-the-art trackers on five popular tracking benchmarks, and experimental results show that PACNet achieves comparable performance with a real-time speed.

Related Work

Visual Tracking

Visual tracking has continuously attracted extensive attention owing to the development of new benchmarks (Kristan et al. 2018, 2019; Fan et al. 2019; Huang, Zhao, and Huang 2019) and various improved methodologies (Li et al. 2019; Bhat et al. 2019). Generally, trackers can be classified into Correlation Filters (CF) and Convolutional Neural Networks (CNN). Both of them have shown their superior performance on various tracking benchmarks.

CF-based tracking approaches aim to learn a correlation filter in a Fourier domain with low computational load. A pioneering work is Minimum Output Sum of Squared Error (MOSSE) filter (Bolme et al. 2010), achieving a very fast tracking speed. A follow-up work, Henriques et al. (Henriques et al. 2015) proposed Kernelized Correlation Filter (KCF) tracker based on multi-channel HOG features, which provides an elegant closed-form solution to compute kernels at all cyclic shifts. Furthermore, to obtain state-of-the-art performance, (Danelljan et al. 2017) exploited deep features for correlation filters. However, they are much slower than the traditional CF-based methods.

Recently, CNN-based methods (Nam and Han 2016; Li et al. 2018) have shown outstanding performance owing to the superior representation ability. Two early attempts (Nam and Han 2016; Tao, Gavves, and Smeulders 2016) perform tracking-by-detection by binary classification and similarity measurement respectively. Through outstanding results achieved, these methodologies run at just 1 fps due to inefficient search strategies (sliding window and candidate sampling) and expensive computation. Meanwhile, (Bertinetto et al. 2016) is aimed to learn a generic similarity function by Siamese network and perform fast matching by cross-correlation. Owing to the high-speed tracking efficiency, there are many follow-up work (Li et al. 2019; Wang et al. 2019; Zhang et al. 2020). Nevertheless, since there is no considering of temporal information and online updating, these methods are insensitive to background distractors and limited to specific scenarios. Moreover, Siamese-based trackers formulated visual tracking as a cross-correlation problem are not in line with human behavior. In contrast, we utilize Long Short Temporal Memory (LSTM) to encode target's temporal relationship between adjacent frames, and offer a novel tracking paradigm to teach machines to track a generic object in videos like humans with hierarchical decisions.

In addition to the above methods, visual tracking also can be considered as a regression problem, which is also suitable for tracking. To be specific, Held et al. (Held, Thrun, and Savarese 2016) developed deep regression networks to predict the target's location. Furthermore, a recurrent regression network (Gordon, Farhadi, and Fox 2018) is proposed to incorporate temporal information into its model for robust tracking. However, these methods have difficulties to track the target in complex scenarios since no online learning procedure. What's more, these models are trained using supervised learning which optimizes parameters for just local prediction. Conversely to this, we adopt DRL-based training strategy which optimizes the model for the maximization of expected reward in future prediction.

Deep Reinforcement Learning

Reinforcement learning is one of effective machine learning paradigms, aiming to learn how to make decisions for maximizing the cumulative future rewards. Recent trends in RL field is to combine the CNNs with RL algorithms for solving high-dimensional complex problems, such as Atari games (Mnih et al. 2015), robotics (Zhu et al. 2017) and Go (Silver et al. 2016). At the same time, DRL has also been exploited in various computer vision tasks, such as object localization (Caicedo and Lazebnik 2015) and object detection (Pirinen and Sminchisescu 2018).

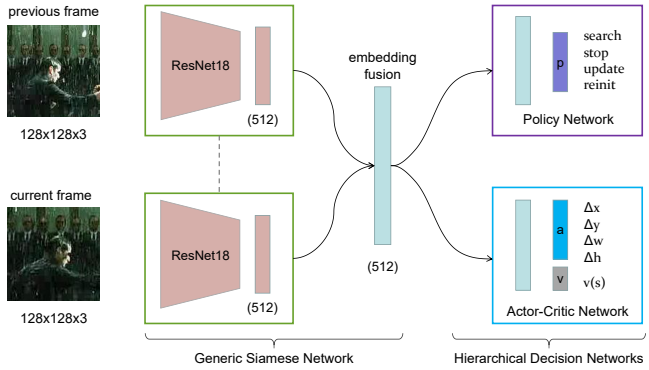


Figure 2: The overview of the proposed tracking framework. PACNet is constituted by a shared feature extractor for observation, a policy network for mode switch and an actor-critic network with an LSTM layer to perform searching.

In recent years, visual tracking also unveils the power of RL. In (Yun et al. 2017), Yun et al. proposed an action-decision network to learn a good policy for seeking the locations and size of the target. In (Huang, Lucey, and Ramanan 2017), Huang et al. exploit RL to learn an early decision policy for selecting features adaptively. Benefiting from this scheme, EAST effectively speeds up the deep tracker without losing accuracy. Furthermore, ACT (Chen et al. 2018) is aimed to predict only one continuous action to locate the tracked object in each frame, while DRL-IS (Ren et al. 2018) introduced DRL technique to learn how to make decisions (shift the current bounding box, stop the shift process, update, and re-initialize the tracker) during tracking process. Recent work (Zhang and Zheng 2020) performs two-stage tracking to pursue higher performance, while (Dunnhofer et al. 2019) claims that the value function learned offline training, can be directly used to exploit the expert demonstrations to adjust wrong tracking results. Different from above-mentioned methods, we propose a novel HDRL-based tracking framework, which can effectively make hierarchical decisions by a Policy agent and an Actor-Critic framework in collaboration.

Methodology

We formulate visual tracking as an iterative search problem. In this work, we attempt to perform robust deep tracking with the HDRL framework, which consists of three sub-networks (the policy network, the actor network and the critic network) with a shared generic Siamese observation network. As illustrated in Figure 2, given the previous tracking results, we can crop and resize the image patches of adjacent frames, and take them as the inputs. Then, the policy agent needs to decide the tracking status (whether or not to continue search or stop search or update or even restart tracking) according to the current observation state, while the actor agent aims to output one continuous action to search the object's location. Both can be effectively offline trained by DRL algorithm. The details of our tracking framework are presented in the following subsections.

Problem Definition

Our tracking algorithm follows the definition of a Markov Decision Process (MDP), which includes state $s \in S$, action $a \in A$, a state transition function $s' = f(s, a)$, and a reward function $r(s, a)$. In our definition, the tracker is viewed as two agents to make decisions on the tracking mode and motion estimation in collaboration. The actions are defined in two different spaces and are used to iteratively infer the accurate bounding box location and size in each frame.

Both agents are interacted with the environment through a temporal sequence of observations $s_1, s_2, ..., s_t$, actions $a_1, a_2, ..., a_t$ and rewards $r_1, r_2, ..., r_t$. In the t-th frame, the actor agent provides a continuous action a_t according to the current observation s_t. Similar to (Chen et al. 2018; Dunnhofer et al. 2019), the actor's action a_t represents the relative motion of the tracked object, i.e. indicating how its bounding box should adjust at frame t. Besides, our tracker introduces an additional policy agent to decide the current tracking mode, which makes the tracker more robust.

State. Given the bounding box $b = [x, y, w, h]$ (denote the center coordinates, width and height respectively), we can define the state s as a pair of image patches. To be specific, $s_t = \phi(b_{t-1}, \mu, F_{t-1}, F_t)$, where μ denotes a scaling factor and $\phi(.)$ is aimed to crop the image patch within the scaled box $[x_{t-1}, y_{t-1}, \mu \cdot h_{t-1}, \mu \cdot w_{t-1}]$ at consecutive frames (F_{t-1} and F_t), and resize them to a fixed size.

Actions and State Transition. For the actor agent, similar to (Dunnhofer et al. 2019), we use the action $a_t = [\Delta x_t, \Delta y_t, \Delta w_t, \Delta h_t]$ to depict the relative motion of the tracked object. Δx and Δy define the relative horizontal and vertical translations, while Δw and Δh denotes the relative scale change of width and height. Therefore, b_t can be obtained by applying the action a_t to the current b_{t-1}:

$$\begin{cases} x_t = x_{t-1} + \Delta x_t \cdot w_{t-1} \\ y_t = y_{t-1} + \Delta y_t \cdot h_{t-1} \\ w_t = w_{t-1} + \Delta w_t \cdot w_{t-1} \\ h_t = h_{t-1} + \Delta h_t \cdot h_{t-1} \end{cases} \tag{1}$$

Then, the state s_t will be transited into s_{t+1} by performing the pre-processing function $\phi(b_t, \mu, F_t, F_{t+1})$.

With the state s_{t+1}, the policy network θ^p is required to generate the action p to decide the current tracking status.

$$\mathbb{P}(p \mid s_{t+1}) = \pi(s_{t+1} \mid \theta^p), \sum_i \mathbb{P}(p_i \mid s_{t+1}) = 1 \tag{2}$$

where $p \in \{search, stop, update, reinit\}$. Action $search$ means to continue searching and predict the relative motion of the target again, while action $stop$ denotes to stop searching and go to the next frame. For the action $update$ and $reinit$, the current observation is not well or the target may be lost, so the demonstration of an expert $b^e = [x, y, w, h]$ is required to guide the search of the current frame.

Rewards. The reward function $r(s, a)$ denotes the quality of the action a taken at state s. Like (Dunnhofer et al. 2019), the reward of actor agent is based on the Intersection-over-Union (IoU) between the box b and the ground-truth g.

$$r(s, a) = \begin{cases} \omega(\text{IoU}(b, g)) & \text{if IoU}(b, g) \geq 0.5 \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

where $\omega(z) = 2 * z - 1$, controlling the range to [0, 1].

For the policy agent, we define different rewards for different actions according to their impacts. Firstly, the reward function of the action $search$ is defined based on the $\Delta_{IoU} = \text{IoU}\left(b_{t,k+1}, g_t\right) - \text{IoU}\left(b_{t,k}, g_t\right)$

$$r(s,p) = \begin{cases} 1 & \Delta_{IoU} \geq \epsilon \\ 0 & 0 \leq \Delta_{IoU} < \epsilon \\ -1 & \Delta_{IoU} < 0 \end{cases} \tag{4}$$

For the action $stop$, to stop with less iterations, the rewards are defined by the IoU and the iteration times k.

$$r(s,p) = \begin{cases} 1/k & \text{IoU}\left(b_{t,k}, g_t\right) \geq 0.7 \\ 0 & 0.5 \leq \text{IoU}\left(b_{t,k}, g_t\right) \leq 0.7 \\ -1 & \text{IoU}\left(b_{t,k}, g_t\right) \leq 0.5 \end{cases} \tag{5}$$

For the action $update$ and $reinit$, the reward is positive when the IoU is less than δ (0.5 and 0, respectively).

$$r(s,p) = \begin{cases} -1 & \text{IoU}\left(b_{t,k}, g_t\right) \geq 0.7 \\ 0 & \delta \leq \text{IoU}\left(b_{t,k}, g_t\right) \leq 0.7 \\ 1 & \text{IoU}\left(b_{t,k}, g_t\right) \leq \delta \end{cases} \tag{6}$$

Network Architecture

The backbone network is initialized by the ResNet-18 (He et al. 2016) pretrained for image classification on ImageNet. With the inputs, deep features extracted by the ResNet-18 are first linearized, then concatenated and fed into two subnetworks with similar structures for hierarchical decisions. The policy network consists of two consecutive fully connected layers with ReLU and outputs the discrete action $p = \pi(s|\theta^p)$. For the Actor-Critic network, we introduce an additional LSTM layer with 512 neurons. The output is the action $a = \pi(s|\theta^a)$ and the value of the state, i.e. $v(s|\theta^v)$.

Offline Training

In this work, we train the Actor-Critic network using the on-policy A3C (Mnih et al. 2016) RL algorithm, which exploits M parallel and independent agents that interact with their environments and update asynchronously the weights θ^a with training sample pairs collected based on the RL rule. It is fast and stable. AC algorithm take the advantage $A(s,a)$ and jointly models the policy function $\pi(a|s)$ and the value function $V^\pi(s)$. Thus, a relative measure of the importance of each action can be represented as:

$$A(s,a) = Q^\pi(a|s) - V^\pi(s) = r + \gamma V^\pi(s') - V^\pi(s) \tag{7}$$

For the loss function of the actor network and critic network can be formulated as $\mathcal{L}_{(\mathcal{A})}$ and $\mathcal{L}_{(\mathcal{V})}$:

$$\mathcal{L}_{(\mathcal{A})} = -\frac{1}{n}\sum_{i=1}^{n} A^\pi\left(s_i, a_i\right) \cdot \log \pi\left(a_i \mid s_i\right) \tag{8}$$

$$\mathcal{L}_{(\mathcal{V})} = \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=0}^{i-1}\gamma^j r_j + \gamma^i V\left(s_i'\right) - V^\pi(s)\right)^2 \tag{9}$$

Due to discrete space, we can train the policy network with PG algorithm (Sutton et al. 2000), which optimizes the action p with respect to the expected future reward J.

$$J_\pi(\theta^p) = \mathbb{E}\left[R_{1:\infty}; \pi(p \mid s; \theta^p)\right] \tag{10}$$

Algorithm 1: Hierarchical Decision Tracking

Input: Frame $\{I_t\}_1^T$, initial bounding box b_1
Output: Optimal target location $\{b_t\}_2^T$

for $t = 2 : T$ **do**
  Obatin initial state s_{t-1} according to b_{t-1} ;
  Actor agent selects the action a ;
  Obtain the state $s_{t-1,k}$;
  Policy agent selects the action p ;
  while $p == search$ **do**
    Actor agent selects the action a ;
    $k \leftarrow k + 1$;
    Obtain the state $s_{t-1,k+1}$;
    Policy agent selects action p ;
  end
  if $p \in \{update, reinit\}$ **then**
    Conduct expert demonstration b^e ;
  end
  Obtain the optimal bounding box b_t.
end

By the PG theorem, it aims to update the parameter θ^p with the gradient $\mathbb{E}\left[\nabla_{\theta^p} \log \pi(p \mid s) R(s)\right]$. Given N trajectory τ, to be an unbiased approximation, the learning process can be achieved by minimizing the following loss function:

$$\mathcal{L}_{(\mathcal{P})} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R\left(\tau^n\right) \log\left(\pi\left(p_t^n \mid s_t^n\right)\right) \tag{11}$$

The proposed HDRL-based framework can be effectively trained offline by the above methods. Therefore, the parameters of our network can be updated with the learning rate α by stochastic policy gradient or value function regression.

$$\theta^a \leftarrow \theta^a + \alpha \nabla_{\theta^a} \log \pi\left(a \mid s\right) A\left(s, a\right) \tag{12}$$

$$\theta^v \leftarrow \theta^v - \alpha \nabla_{\theta^v} \frac{1}{2}\left(R(n) - V^\pi\left(s\right)\right)^2 \tag{13}$$

$$\theta^p \leftarrow \theta^p + \alpha \nabla_{\theta^p} \log \pi\left(p \mid s\right) R\left(\tau\right) \tag{14}$$

Hierarchical Decision in Tracking

To ensure tracking efficiency, no online updating of network is performed. Benefiting from the HDRL framework, the online tracking strategy of hierarchical decision can iteratively search the tracked object and adaptively exploit the expert tracker as the demonstrator to adjust the bounding box. Therefore, our tracker effectively guarantees the performance in terms of accuracy and robustness.

For online tracking, the actor agent firstly generates a continuous action for searching the target according to the current state. After happening the state transition process, the new observation is obtained. The policy agent then decides the next tracking status until stopping. Algorithm 1 shows the tracking procedure in detail. Furthermore, Figure 3 presents an example of sampling action sequence during inference. The policy and actor agents formulate the motion estimation and tracking mode change in a unified way as taking actions in hierarchical reinforcement learning.

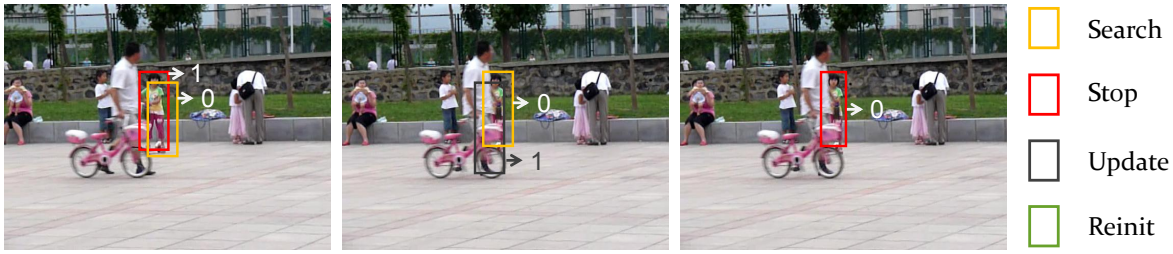


Figure 3: An illustrative example of the tracking mode switch controlled by the policy agent: 1) at frame t, the object is readily located by one $search$ step; 2) at frame $t+1$, firstly, a $search$ step tends to a distractor nearby, than the policy agent realizes this and takes an $update$ action to conduct expert demonstration; 3) the current tracking result is updated by the expert tracker.

Experiments

In this section, we present the results of our tracker on five tracking benchmarks, with comparisons to the state-of-the-art algorithms. Experimental analysis is also provided to evaluate the effect of each component in our approach.

Implementation Details

Training. We initialize our backbone network with the parameters pre-trained on ImageNet (Russakovsky et al. 2015). The proposed tracker is trained on the training set of the GOT-10k (Huang, Zhao, and Huang 2019), which is a large-scale dataset including 9335 training sequences, 180 validation and other 180 testing videos for evaluation. We crop the image patch within the bounding box scaled by $\mu = 1.5$ and resize it to $(128 \times 128 \times 3)$ for fitting the input size of the network. During training, four GPUs are used, and a total number of $M = 12$ training agents is set. The discount factor γ is set to 1. In each iteration of the reinforcement learning, we randomly select the sequence of length $L = 5$ for the tracking simulation. We apply the Adam optimizer to train the model for 40000 episodes until convergence. The learning rate of both agents is set to 10^{-6}, while the weight decay coefficient is set to 10^{-4}.

Tracking. For inference, our model can directly perform robust tracking without any online updating of the network. To ensure tracking efficiency, the maximal number k of policy's actions is set to 3 for each frame. For the action $update$ and $reinit$, we select the demonstration of an expert tracker as the tracking result. To this end, the role of expert tracker is assigned to SiamRPN++ (Li et al. 2019) or DiMP-50 (Bhat et al. 2019). This choice is motivated by the fact that both trackers have achieved significant performance and shown great balance between the accuracy and speed.

Our tracker is implemented in Python with the Pytorch 1.2 framework, which runs about at 40 fps on a PC with Intel(R) Xeon(R) CPU E5-2683 @2.10 GHz with 64G RAM and a NVIDIA GeForce GTX 2080 Ti GPU.

Evaluation Datasets and Metrics. We use five tracking benchmarks including VOT-2019, OTB-100, UAV123, GOT-10k and LaSOT for tracking performance evaluation. For VOT-2019, we take the accuracy (A), robustness (R) and Expected Average Overlap (EAO) into account to evaluate trackers. For OTB-100, UAV123 and LaSOT, we adopt one-pass evaluation (OPE) with distance precision and overlap metrics. The center location error threshold is set to 20 pixels, and the area-under-curve (AUC) score of overlap success plots is computed to evaluate the overall performance. The trackers are also evaluated using an online server on a test set of GOT-10k, which employs the average overlap (AO) and success rate (SR) as performance indicators.

Ablation Analysis

Self-comparison. To verify the effectiveness of each component in our algorithm, we conduct several variants of our tracker and evaluate them using OTB-100. These variants include: 1) A3CT: a baseline tracker which only adopts the actor agent to perform tracking; 2) 'PACNet-search' is a hierarchical decision-based model which guided with only two action types: $search$ and $stop$; 3) 'PACNet-expert' is a hierarchical decision-based model without the action $search$; 4) PACNet is our final model which is guided with full action types: $search$, $stop$, $update$ and $reinit$.

Table 1 reports the distance precision and overlap success rates of these variations. Obviously, A3CT performs not well since the model can only search by one-step and do not take expert tracker into account. By conducting the policy agent with the action $search$, 'PACNet-search' achieves 18.4% and 14.8% improvement in terms of precision and overlap metrics, compared to the baseline. When conducting the policy agent without the action $search$, 'PACNet-expert' further gained 9.3% and 7.5% additional improvements to 'PACNet-search'. This result shows that the tracking performance can be guaranteed by exploiting the expert tracker to guide the actor agent by behavior demonstrations. Moreover, PACNet incorporates all actions into the policy agent, and achieves 3.1% and 2.8% performance gains. These self-comparison results strongly demonstrate that the proposed hierarchical decision framework could effectively learn a good policy in collaboration for robust tracking.

Variants	A3CT	P-search	P-expert	PACNet
Prec.(20px)	0.568	0.752	0.845	0.876
IOU(AUC)	0.419	0.567	0.642	0.670

Table 1: The self-comparison results on OTB-100 dataset.

Tracker	OTB-100		UAV-123		LaSOT	
	Pre.	AUC	Pre.	AUC	Pre.	AUC
GOTURN	0.534	0.395	0.548	0.389	0.175	0.214
Re3	0.582	0.464	0.667	0.514	0.301	0.325
ADNet	0.880	0.646	0.724	0.483	-	-
ACT	0.859	0.625	0.636	0.415	-	-
DRL-IS	0.909	0.671	-	-	-	-
A3CT	0.568	0.419	0.622	0.471	0.246	0.306
A3CTD	0.717	0.535	0.754	0.565	0.368	0.415
PACNet	0.876	0.670	0.827	0.620	0.546	0.553

Table 2: The comparisons of relevant tracking methods of PACNet on the OTB-100, UAV-123 and LaSOT datasets.

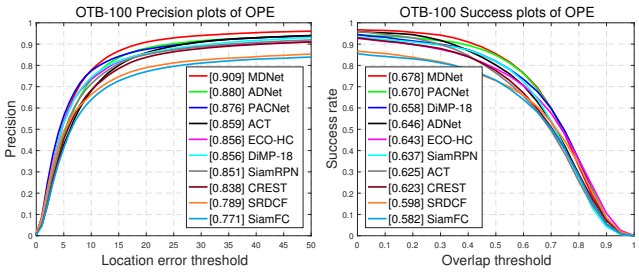


Figure 4: Precision and success plots of OPE on OTB-100.

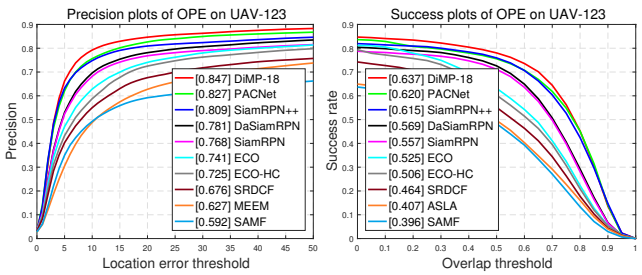


Figure 5: Precision and success plots of OPE on UAV-123.

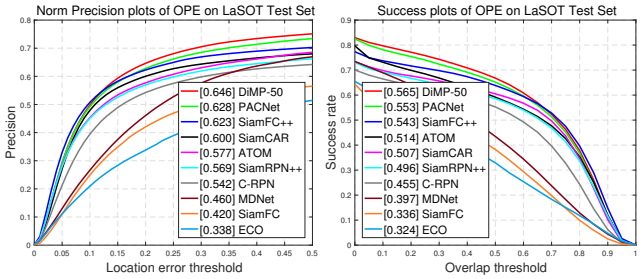


Figure 6: Normalized precision and success plots of OPE on LaSOT testing set for twelve recent tracking methods.

Comparison with the Relevant Trackers. We note that the most relevant trackers of PACNet are GOTURN, RE3, ADNet, ACT, DRL-IS and A3CTD. Both of them consider tracking as a search process but with different strategies.

The detailed comparisons are reported in Table 2. GOTURN and RE3 exploit regression networks to conduct search but with not well performance. ADNet exploits few discrete actions to conduct iterative searching, and achieves good performance but with 3 fps. ACT further speeds up it with slightly worse accuracy. DRL-IS runs about 10 fps due to time-consuming model update and a complicated restart process. We note that A3CTD runs about 50 fps but with not good performance, which means searching by one step cannot output an accurate bounding box and online update is required to adjust domain-specific objects. In contrast, we take the iterative search and Expert-based guidance into account. PACNet achieves a comparable accuracy on OTB and performs the best on UAV-123 and LaSOT. Nevertheless, our tracker can run at 40 fps. This indicates our method achieves a good trade-off between the accuracy and speed.

Comparison with State-of-the-arts

OTB-100 Dataset. OTB-100 (Wu, Lim, and Yang 2015) is a widely used classical benchmark in visual tracking with 100 videos. The results are reported within Figure 4. We can observe that our PACNet surpasses most trackers, such as DiMP-18 (Bhat et al. 2019), ACT (Chen et al. 2018), ECO-HC (Danelljan et al. 2017) and SiamRPN (Li et al. 2018). MDNet (Nam and Han 2016) performs the best in terms of accuracy but runs only 1 fps. Compared to ADNet (3 fps), we achieves a comparable precision score and a distinct improvement in AUC with 2.4%. Nevertheless, the proposed tracker is superior in terms of accuracy and speed.

UAV-123 Dataset. UAV-123 dataset includes 123 videos captured from a UAV with average sequence length of 915 frames. Besides SRDCF (Danelljan et al. 2015), ECO (Danelljan et al. 2017), SiamRPN (Li et al. 2018) and DaSiamRPN (Zhu et al. 2018), recent DiMP (Bhat et al. 2019) and SiamRPN++ (Li et al. 2019) are also added for comparisons. Figure 5 illustrates the precision and success plots of the compared trackers. Specifically, our tracker achieves an AUC score of 0.620, which outperforms DaSiamRPN (0.569), SiamRPN (0.557) and ECO (0.525) with a large margin. Compared to SiamRPN++, PACNet still obtains slight gains with 1.8% and 0.5% in precision and AUC.

LaSOT Dataset. We further evaluate PACNet on LaSOT test set consisting of 280 sequences. Compared to other datasets, LaSOT (Fan et al. 2019) has longer sequences with average 2500 frames, which is the largest benchmark in visual tracking. Following one-pass evaluation, 12 different trackers are evaluated based on normalized precision and success plots. The comparison results are shown in Figure 6. We observe that our tracker ranks second on LaSOT, second only to DiMP-50. Benefiting from hierarchical decision strategy, PACNet exceeds other 10 state-of-the-arts.

The performance of PACNet can be attributed to the expert tracker (i.e., DiMP-50) to some extent. We improve the original A3CT tracker using the policy agent which can adaptively switch the tracking status according to the observation state. When the action $update$ or $reinit$ is selected, we exploit the tracking result of the expert tracker to adjust the current bounding box. Therefore, we can claim that behavior demonstration of a more strong expert tracker can further improve the performance of our algorithm.

	A3CTD	TADT	MemDTC	SPM	SiamRPN++	SiamMask	ATOM	SiamDW	DiMP	**PACNet**
EAO ↑	0.165	0.207	0.228	0.275	0.285	0.287	0.292	0.299	0.321	0.300
A ↑	0.451	0.516	0.485	0.577	0.599	0.594	0.603	0.600	0.582	0.573
R ↓	0.933	0.677	0.587	0.507	0.482	0.461	0.411	0.467	0.371	0.401

Table 3: Comparison with ten state-of-the-art methods on the VOT-2019 in terms of EAO, Accuracy and Robustness.

	MDNet	ECO	GOTURN	SiamFC	A3CTD	SiamRPN++	ATOM	SiamCAR	DiMP	**PACNet**
AO	0.299	0.316	0.347	0.348	0.425	0.518	0.556	0.569	0.611	0.582
$SR_{0.5}$	0.303	0.309	0.375	0.353	0.495	0.618	0.634	0.670	0.717	0.685
$SR_{0.75}$	0.099	0.111	0.124	0.098	0.205	0.325	0.405	0.415	0.492	0.443

Table 4: Comparison with state-of-the-arts on the GOT-10K test set in terms of AO and SR at thresholds 0.5 and 0.75.

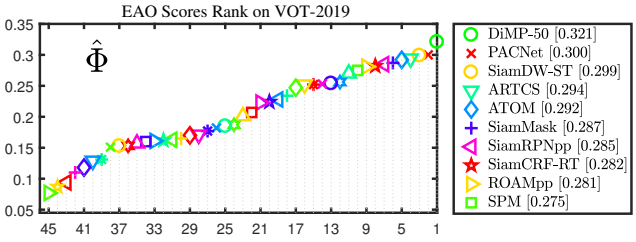


Figure 7: EAO score performance on the VOT-2019 dataset.

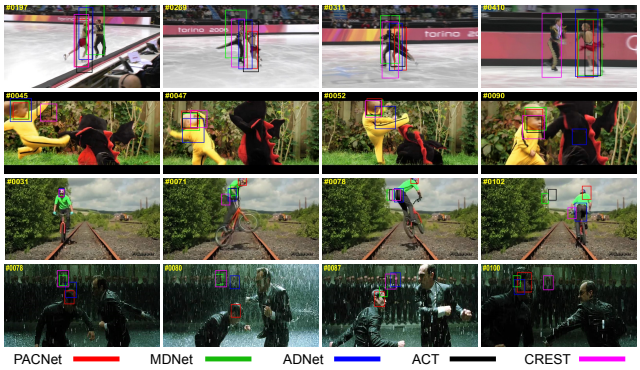


Figure 8: Qualitative results of state-of-the-art trackers.

VOT-2019 Dataset. Table 3 reports the evaluation results with the comparisons to recent prevailing trackers on VOT-2019 which provides 60 auto-annotated challenging videos. We can observe that the recent DiMP tracker achieves the best performance (0.321), while our PACNet ranks second (0.300) in terms of EAO. Our method surpasses SiamDW (Zhang and Peng 2019), ATOM (Danelljan et al. 2019), SiamRPN++ (Li et al. 2019) and SiamMask (Wang et al. 2019) by a light margin on EAO, which utilize larger training datasets than ours and perform better in terms of accuracy metric. Moreover, Figure 7 shows the comparison results of EAO in detail. For the robustness, PACNet obtains a score of 0.401 and ranks second, achieving an absolute advantage compared to Siamese based tracking methods.

GOT-10K Dataset. GOT-10k (Huang, Zhao, and Huang 2019) is a large-scale dataset containing over 10 thousand videos. To ensure a fair comparison, all trackers should use the GOT-10K training set, and be evaluated by an online server. After submitting the tracking results, the analysis can be obtained automatically by the official website. The evaluation protocol is the one-pass evaluation (OPE) using three metrics including average overlap (AO) and the success rates (SR) with overlap thresholds 0.5 and 0.75.

As shown in Table 4, we report the results of PACNet against the state-of-the-art including DiMP (Bhat et al. 2019), SiamCAR (Guo et al. 2020), ATOM (Danelljan et al. 2019), SiamRPN++ (Li et al. 2019) and other 5 tracking methods. PACNet outperforms all trackers reported on the GOT-10k. In particular, compared with A3CTD and SiamRPN++, our tracker obtains significant improvements of 15.7% and 6.4% in AO, 19% and 6.7% in $SR_{0.50}$ as well as 23.8% and 11.8% in $SR_{0.75}$. We perform worse than the expert tracker DiMP, however these results are obtained considering just part of demonstrations of the expert.

Qualitative Results

Figure 8 shows qualitative comparisons with five tracking methods including MDNet (Nam and Han 2016), ADNet (Yun et al. 2017), ACT (Chen et al. 2018), CREST (Song et al. 2017) and PACNet on four challenging sequences (*(Skating2, DragonBaby, Biker* and *Matrix*) of OTB dataset. We observe that our tracker can robustly locate the targets under severe appearance changes such as scale variation, deformation and fast motion, which confirms its effectiveness. In contrast, other tracking methods have failure cases.

Conclusion

In this paper, we present a hierarchical deep reinforcement learning (HDRL) framework with Policy and Actor-Critic Networks (PACNet) for robust visual tracking. We show that the tracking problem can be effectively solved by dynamic iterative search process and behavior guidance of an expert tracker. Our proposed PACNet is simple in structure but can track generic objects in line with human thinking paradigm. Benefiting from the hierarchical decision tracking strategy, our method achieves state-of-the-art results on five challenging tracking benchmarks. This demonstrates the generalization performance. In the future work, we will study the fast update of the parameters using meta reinforcement learning to perform domain adaptation for tracking.

Acknowledgments

This work was supported by the Natural Science Foundation of China under Grant No. 61672467, No. 61902358, and the Special Fund for Scientific Instruments of the Natural Science Foundation of China under Grant No. 61827810.

References

Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. S. 2016. Fully-Convolutional Siamese Networks for Object Tracking. In *European Conference on Computer Vision*, 850–865.

Bhat, G.; Danelljan, M.; Gool, L. V.; and Timofte, R. 2019. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, 6182–6191.

Bolme, D. S.; Beveridge, J. R.; Draper, B. A.; and Lui, Y. M. 2010. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2544–2550.

Caicedo, J. C.; and Lazebnik, S. 2015. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE international conference on computer vision*, 2488–2496.

Chen, B.; Wang, D.; Li, P.; Wang, S.; and Lu, H. 2018. Real-time 'Actor-Critic' Tracking. In *The European Conference on Computer Vision (ECCV)*, 318–334.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2019. ATOM: Accurate Tracking by Overlap Maximization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4660–4669.

Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; and Felsberg, M. 2017. Efficient Convolution Operators for Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6638–6646.

Danelljan, M.; Hager, G.; Shahbaz Khan, F.; and Felsberg, M. 2015. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, 4310–4318.

Dunnhofer, M.; Martinel, N.; Luca Foresti, G.; and Micheloni, C. 2019. Visual Tracking by Means of Deep Reinforcement Learning and an Expert Demonstrator. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*.

Fan, H.; Lin, L.; Yang, F.; Chu, P.; Deng, G.; Yu, S.; Bai, H.; Xu, Y.; Liao, C.; and Ling, H. 2019. LaSOT: A High-Quality Benchmark for Large-Scale Single Object Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5374–5383.

Gordon, D.; Farhadi, A.; and Fox, D. 2018. Re3: Real-Time Recurrent Regression Networks for Visual Tracking of Generic Objects. *IEEE Robotics and Automation Letters* 3(2): 788–795.

Guo, D.; Wang, J.; Cui, Y.; Wang, Z.; and Chen, S. 2020. SiamCAR: Siamese Fully Convolutional Classification and Regression for Visual Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6269–6277.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Held, D.; Thrun, S.; and Savarese, S. 2016. Learning to Track at 100 FPS with Deep Regression Networks. In *European Conference on Computer Vision*, 749–765.

Henriques, J. F.; Caseiro, R.; Martins, P.; and Batista, J. 2015. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(3): 583–596.

Huang, C.; Lucey, S.; and Ramanan, D. 2017. Learning Policies for Adaptive Tracking With Deep Feature Cascades. In *The IEEE International Conference on Computer Vision (ICCV)*, 105–114.

Huang, L.; Zhao, X.; and Huang, K. 2019. GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .

Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R.; et al. 2018. The sixth Visual Object Tracking VOT2018 challenge results. In *The European Conference on Computer Vision (ECCV) Workshops*.

Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; Pflugfelder, R.; Kamarainen, J.-K.; Cehovin Zajc, L.; et al. 2019. The seventh visual object tracking vot2019 challenge results. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*.

Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; and Yan, J. 2019. SiamRPN++: Evolution of Siamese Visual Tracking With Very Deep Networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Li, B.; Yan, J.; Wu, W.; Zhu, Z.; and Hu, X. 2018. High Performance Visual Tracking With Siamese Region Proposal Network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 8971–8980.

Mnih, V.; Badia, A. P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; and Kavukcuoglu, K. 2016. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, 1928–1937.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; et al. 2015. Human-level control through deep reinforcement learning. *nature* 518(7540): 529–533.

Nam, H.; and Han, B. 2016. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4293–4302.

Pirinen, A.; and Sminchisescu, C. 2018. Deep reinforcement learning of region proposal networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6945–6954.

Ren, L.; Yuan, X.; Lu, J.; Yang, M.; and Zhou, J. 2018. Deep Reinforcement Learning with Iterative Shift for Visual Tracking. In *The European Conference on Computer Vision (ECCV)*, 684–700.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; et al. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* 115(3): 211–252.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529(7587): 484–489.

Song, Y.; Ma, C.; Gong, L.; Zhang, J.; Lau, R.; and Yang, M.-H. 2017. CREST: Convolutional Residual Learning for Visual Tracking. In *IEEE International Conference on Computer Vision*, 2555 – 2564.

Sutton, R. S.; McAllester, D. A.; Singh, S. P.; and Mansour, Y. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, 1057–1063.

Tao, R.; Gavves, E.; and Smeulders, A. W. 2016. Siamese Instance Search for Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Q.; Zhang, L.; Bertinetto, L.; Hu, W.; and Torr, P. H. 2019. Fast Online Object Tracking and Segmentation: A Unifying Approach. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, Y.; Lim, J.; and Yang, M.-H. 2015. Object Tracking Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(9): 1834–1848.

Yun, S.; Choi, J.; Yoo, Y.; Yun, K.; and Jin, Y. C. 2017. Action-Decision Networks for Visual Tracking with Deep Reinforcement Learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2711–2720.

Zhang, D.; and Zheng, Z. 2020. High Performance Visual Tracking With Siamese Actor-Critic Network. In *IEEE International Conference on Image Processing (ICIP)*, 2116–2120.

Zhang, D.; Zheng, Z.; Li, M.; He, X.; Wang, T.; Chen, L.; Jia, R.; and Lin, F. 2020. Reinforced Similarity Learning: Siamese Relation Networks for Robust Object Tracking. In *Proceedings of the 28th ACM International Conference on Multimedia*, 294–303.

Zhang, Z.; and Peng, H. 2019. Deeper and Wider Siamese Networks for Real-Time Visual Tracking. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4591–4600.

Zhu, Y.; Mottaghi, R.; Kolve, E.; Lim, J. J.; Gupta, A.; Fei-Fei, L.; et al. 2017. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *IEEE International Conference on Robotics and Automation*, 3357–3364.

Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; and Hu, W. 2018. Distractor-aware Siamese Networks for Visual Object Tracking. In *The European Conference on Computer Vision (ECCV)*, 101–117.