SINT++: Robust Visual Tracking via Adversarial Positive Instance Generation

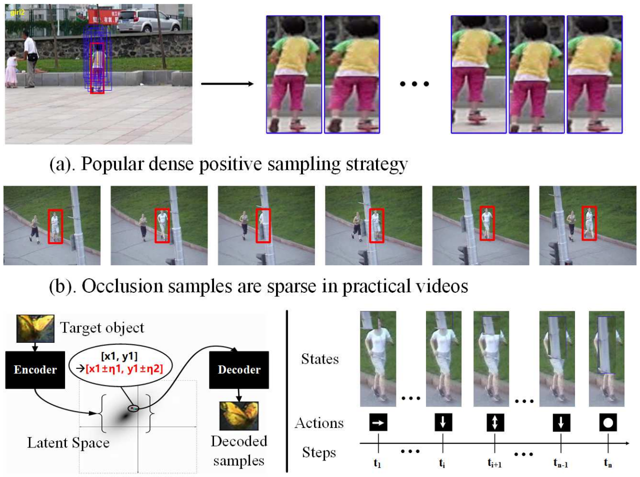
Xiao Wang, Chenglong Li, Bin Luo, Jin Tang
School of Computer Science and Technology, Anhui University, Hefei, China 230601
{wangxiaocvpr, lcl1314}@foxmail.com, {luobin, tj}@ahu.edu.cn

Abstract

Existing visual trackers are easily disturbed by occlusion, blur and large deformation. We think the performance of existing visual trackers may be limited due to the following issues: i) Adopting the dense sampling strategy to generate positive examples will make them less diverse; ii) The training data with different challenging factors are limited, even through collecting large training dataset. Collecting even larger training dataset is the most intuitive paradigm, but it may still can not cover all situations and the positive samples are still monotonous. In this paper, we propose to generate hard positive samples via adversarial learning for visual tracking. Specifically speaking, we assume the target objects all lie on a manifold, hence, we introduce the positive samples generation network (PSGN) to sampling massive diverse training data through traversing over the constructed target object manifold. The generated diverse target object images can enrich the training dataset and enhance the robustness of visual trackers. To make the tracker more robust to occlusion, we adopt the hard positive transformation network (HPTN) which can generate hard samples for tracking algorithm to recognize. We train this network with deep reinforcement learning to automatically occlude the target object with a negative patch. Based on the generated hard positive samples, we train a Siamese network for visual tracking and our experiments validate the effectiveness of the introduced algorithm. The project page of this paper can be found from the website [1].

1. Introduction

Visual tracking aims at covering the give target object using an adaptive bounding box (BBox) and moves the BBox along with target. It requires the tracker can robustly model invarances to illumination, deformation, occlusions and scale variation, *et al*. The most intuitive paradigm to handle these invariances is to collect large-scale training videos which contain video frames under different challenging factors. They hope these videos could capture all possible variations of target object and the algorithm can then effectively model invariance to them. These data indeed learning invariances to some extent, and this maybe the main reasons why CNNs have been so successful in the visual tracking.



(a). Popular dense positive sampling strategy

(b). Occlusion samples are sparse in practical videos

(c). Manifold construction with training data and occlusion samples generation with reinforcement learning

Figure 1. Due to only one target object provided in visual tracking and most existing trackers adopt the dense positive sampling strategy, which cause the positive training data lack of diversity. Moreover, we argue that occlusions and deformations follow a long-tail distribution, in another word, some occlusions and deformations are rare in training data. In this paper, we introduce two strategies to improve aforementioned situations via the generation of hard positive samples.

However, according to our observation, the target object of visual tracking could be anything, even occlusions and deformations are all following a long-tail distribution. In other word, some occlusions and deformations are so rare that they may still not occur in large-scale datasets, similar claims can be found in [43]. Hence, the learning of invariance to such rare/uncommon occlusions and deformations needs to be tackled urgently. Collecting even larger datasets is a possible scheme, but it is not likely to scale due

[1] https://sites.google.com/view/cvpr2018sintplusplus/

to the long-tail statistics. Another issue is that deep neural networks (DNNs) need massive labelled training data to obtain robust feature representation, however, only one positive sample is given in visual tracking. Hence, there exists a gap between data-driven DNNs and visual tracking. As shown in Fig. 1, most of existing deep learning based trackers adopt the dense sampling strategy, which extract positive samples according to Intersection over Union (IoU) compared with ground truth BBox. This operation may cause the positive samples not diverse enough for DNNs and hence a sub-optimal tracking performance maybe obtained. One possible solution is the data augmentation which is an essential part of the training process applied to deep learning models. The dominant data augmentation approach in the field assumes that new training samples can be obtained via random geometric or appearance transformations applied to annotated training samples, but this is a strong assumption because it is unclear if this is a reliable generative model for producing new training samples as illustrated in [41].

In this paper, we propose a novel approach to address aforementioned issues through the generation of positive and hard training samples. Specifically speaking, our pipeline contains three main modules, *i.e.*, positive sample generation network (PSGN), hard positive transformation network (HPTN) and two streaming Siamese instance search network, as shown in Figure 2. We first propose the positive sample generation module, which is a variational auto-encoder (VAE) [16], for generating samples from encoded hidden space. The assumption of this idea is that the target objects are all lie on a manifold, hence, we can decode the samples which similar to encoded target object but with more diversity and some degree of deformation and motion blur. We can obtain massive diverse positive samples via this network and bridge the gap between data-driven deep neural networks and one target provided in visual tracking task. In addition, we also propose the hard positive transformation network (HPTN) to generate more challenging hard positve samples. The HPTN is trained via deep reinforcement learning technique which can learn to occlude the target object with background image patch. These data will address the sparse issue of challenging video frames in training data and make the visual tracker more robust to some challenging factors, such as illuminaton, occlusion.

Our contributions can be summarized as follows: Firstly, we propose a novel and general positive sample generation strategy to bridge the gap between data hunger deep neural networks and visual tracking task. We assume the target objects lie on a manifold, hence, we can decode massive diverse and some degree of deformation positive samples to training the DNNs for tracking. Secondly, we introduce the hard positive transformation network which can generate massive hard positive samples. We take this hard positive generation process as decision making problem which can be optimized by deep reinforcement learning. Finally, based on aforementioned policies, we propose the SINT++ which improves tracking performance of the two streaming Siamese network.

2. Related Works

We will give a brief review about visual tracking, deep generative models and deep reinforcement learning in this section.

Visual Tracking. We discuss the tracking methods that are most related to our work in this section, and refer the readers to a thorough review on object tracking survey [47] [24] and benchmark evaluations [38] [45]. Existing deep learning based trackers can be divided into two main streams, *i.e.* trackers that combine correlation filter and deep features, and neural network based trackers. Directly applying correlation filters on the multi-dimensional feature maps of deep Convolutional Neural Networks (CNNs) is one straight-forward way of integrating deep learning for tracking. The DCF tracker usually trained on every convolutional layer are combined by a hierarchical ensemble method [26] or an adaptive hedge algorithm [32]. Danelljan *et al.* [10] recently introduced a continuous spatial domain formulation C-COT, to enable efficient integration of multi-resolution deep features. Other CNN based methods either combine several CNN models (e.g. the DeepTrack model [23]) or establish an end-to-end CNN model, such as the GOTURN tracker [13] with no online training but offline learn a generic relationship between object motion and appearance from large number of videos. Apart from CNN models, other deep models, such as Siamese network [1] [5] [40] and Recurrent Neural Networks (RNNs), are also employed in the tracking problem. For instance, the SIAM tracker [1] learns a similarity metric offline by a Siamese network, RDM [5] proposed a template selection strategy based on a Siamese network, and the RTT tracker [6] describes a multidirectional RNN to capture long-range contextual cues.

Deep Generative Models. Recently, deep learning based generative models have been discussed widely, such as Pixel-RNN [30], Pixel-CNN [35] [31], variational autoencoder (VAE) [17] and generative adversarial networks (GANs) [12]. There are many applications utilize these generative models to achieve better performance, such as object detection [43], super-resolution [21], text to image transformation [46]. VAE [17] allows us to formalize this problem in the framework of probabilistic graphical models where we are maximizing a lower bound on the log likelihood of the data. Kihyuk Sohn *et al.* propose a deep conditional generative models (CGMs) for output representation learning and structured prediction in [39]. Jacob Walker *et al.* [42] propose a conditional variational autoencoder as a so-

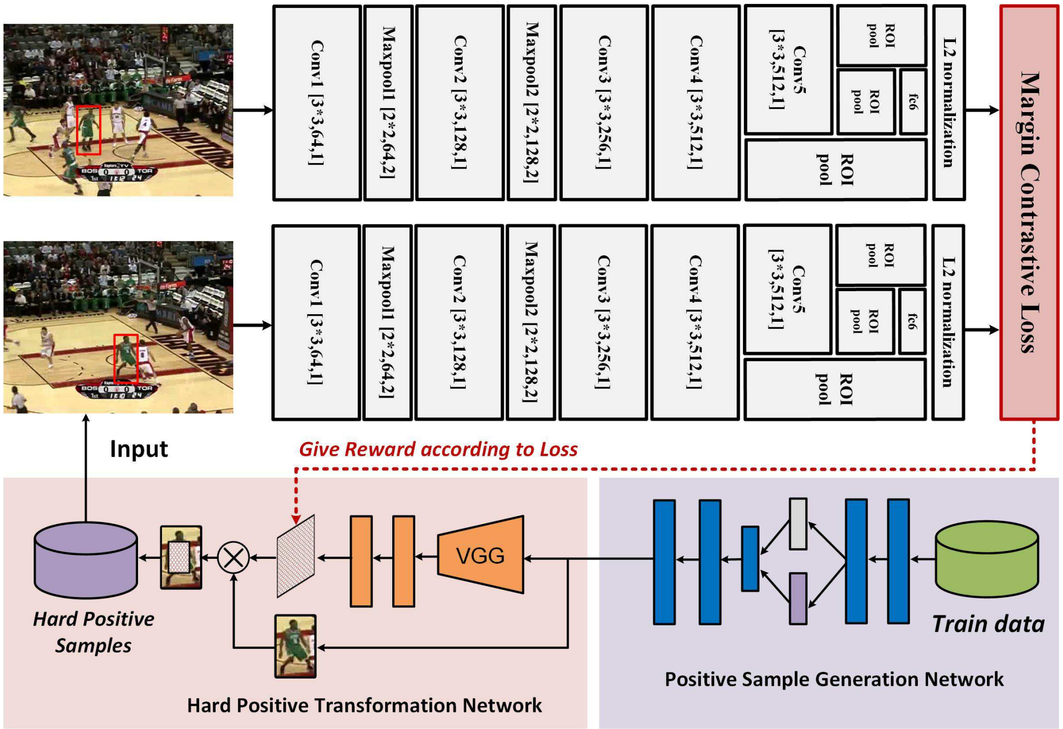


Figure 2. The pipeline of our proposed SINT++, which contain three main modules, *i.e.* positive sample generation network (PSGN), hard positive transformation network (HPTN) and two streaming Siamese network. The target object manifold is constructed by VAE, and the output can be directly input to the HPTN. The HPTN takes the reconstructed image as input, and learn to occlude the target which become hard for visual tracker to measure via deep reinforcement learning. We train SINT++ with the generated hard positive samples and improve its robustness significantly.

lution to forecasting from static images. In this framework, direct inference from the image shapes the distribution of possible trajectories while latent variables encode information that is not available in the image. Zhang *et al.* [50] propose Conditional Adversarial Auto-Encoder (CAAE) network which achieve age progression/regression on the basis of assumption that the face images are lie on a manifold. And images are clustered according to their ages and personality by a different direction. Our neural network for the positive samples generation is also based on VAE and this is also the first attempt to introduce the VAE into the visual tracking community.

Deep Reinforcement Learning. Deep reinforcement learning was first proposed by Mnih *et al.* [28] in 2013 which utilize deep neural networks, *i.e.* Deep Q-learning Networks (DQN) to parametrize an action-value function to play Atari games, reaching human-level performance. The most relevant and successful application of reinforcement learning maybe the game of Go which combined policy network and value network and beat many world-class professional player [36] [37]. Asynchronous deep reinforcement learning was also introduced in [27] to tackle the training efficiency issue by Mnih *et al.* On the aspect of computer vision applications, DQN also applied to many domains, such as object detection [3], visual tracking [49], Face Hallucination [4].

3. The Overview of SINT++

The main target of this paper is to improve the robustness of visual tracking through the generation of hard positive samples. As shown in Figure 2, our SINT++ contains three modules, *i.e.* *positive sample generation*, *hard positive transformation* and *Siamese instance search network*. We will give a brief introduction about these modules, respectively.

The goal of positive sample generation network is to generate diverse similar positive samples based on the assumption that the target objects lie on a high-dimensional manifold, on which traversing along certain direction could decode diverse positive samples. However, modeling the high-dimensional manifold is complicated, and it is difficult to directly manipulate (traversing) on the manifold. Therefore, we will learn a mapping between the manifold and a lower-dimensional space, referred to as the *latent space*, which is easier to manipulate.

On the other hand, hard positive transformation network is introduced to learn an visual tracker which is robust to different conditions, such as occlusion, deformation and illumination. Since it is impossible to cover all potential occlusions and deformations even in large-scale datasets, we actively generate examples which are hard for the visual tracker to recognize instead of relying heavily on the dataset or sifting through data to find hard examples.

With these two networks, we can obtain massive positive samples which are hard and diverse for the training of deep learning based visual trackers. The basic visual tracker used in this paper is Siamese INstance search Tracker (SINT) [40], it simply matches the initial target in the first frame with the candidates in a new frame and returns the most similar one by the learnt matching function, without updating the target, tracker combination, occlusion detection and alike,

$$\hat{x_t} = \arg_{x_{j,t}} \max m(x_{t=0}, x_{j,t}), \tag{1}$$

where $x_{j,t}$ are all the candidate boxes at frame t, m is the learned matching function, $m(x, y) = f(x)^T f(y)$. We prefer readers to check [40] to further understand the SINT tracker.

4. SINT++: Approach Details

We will first introduce the positive sample generation network, followed by describing the hard positive transformation network. In particular, we focus on generating different types of motion blur and occlusions in this paper.

4.1. The Positive Sample Generation Network

In order to maximize the number of samples for the training of deep neural networks, traditional methods usually sample positive data from a candidate region. The candidate region usually take the center of ground truth bounding box (BBox) and enlarge a little. If the intersection-over-union (IOU) between proposal and ground truth BBox larger than a pre-defined threshold, it is treated as positive sample; otherwise, negative sample. As shown in Figure 1, however, the extracted positive data are almost the same, in other words, the positive training data is not diverse. This may lead to under-fitting of learned classifier for visual tracking.

In this paper, we propose to utilize the variational auto-encoder (VAE) to learn the target object manifold. With the constructed manifold, we can decode massive samples by traversing over it. As shown in Figure 1, we can not only sample out similar target objects, but also the target object which not occured in training data, such as target object between video frames.

4.1.1 Review of Variational Auto-Encoders

We use $\mathbf{x}$ to denote the data we want to model, $\mathbf{z}$ denotes latent variable, $p(\mathbf{x})$ means the probability distribution of the data, $p(\mathbf{z})$ denotes the probability distribution of latent variables, and $p(\mathbf{x}|\mathbf{z})$ means the distribution of generating data given latent variable.

The VAE is a deep directed graph model with latent variable $\mathbf{z}$. As we know, the inference of posterior $p_\theta(\mathbf{z}|\mathbf{x})$ is intractable to compute. $q_\theta(\mathbf{z}|\mathbf{x})$ is introduced in the VAE framework to approximate the true posterior by optimizing the variational lower bound. VAE map the input image into continuous latent variables $q_\theta(\mathbf{z}|\mathbf{x})$ via encoder and map latent variables to reconstructed image $p_\theta(\mathbf{x}|\mathbf{z})$ with decoder. The variational lower bound for individual sample $\mathbf{x}_i$ can be written as:

$$L(\theta, \phi, \mathbf{x}_i) = -D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}_i)||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})] \tag{2}$$

The first term in above equation is the KL divergence of the approximate from the true posterior. And the second one is expected reconstructed loss, w.r.t. the approximated posterior $q_\phi(\mathbf{z}|\mathbf{x}_i)$. We can differentiate $L(\theta, \phi, \mathbf{x}_i)$ and do gradient descent with standard back propagation algorithm to optimize the lower bound. Usually, we assume the $q_\phi(\mathbf{z}|\mathbf{x}_i)$ and $p_\theta(\mathbf{z})$ are all gaussian, hence, we can integrate KL divergence term analytically.

$$-D_{KL}(q_\phi(\mathbf{z}|\mathbf{x}_i)||p_\theta(\mathbf{z})) = \frac{1}{2}\sum_{j=1}^{J}(1 + \log(\sigma_j^2) - \mu_j^2 - \sigma_j^2) \tag{3}$$

where J is the dimension of $\mathbf{z}$. The mean μ and σ are simply outputs of encoder function of $\mathbf{x}$ and the variational parameter ϕ. Since the gradient of $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x}_i)}[\log p_\theta(\mathbf{x}_i|\mathbf{z})]$ is not straightforward for the expected reconstructed term , we can reparameterize the random variable $\mathbf{z}$ using a differentiable transformation with auxiliary noise random variable ϵ as [16] [33]. Hence, the resulting estimator for a sample $\mathbf{x}_i$ can be written as:

$$L(\theta, \phi; \mathbf{x}_i) \approx \frac{1}{2}\sum_{j=1}^{J}(1+\log(\sigma_j^2)-\mu_j^2-\sigma_j^2)+\frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{x}_i|\mathbf{z}_{i,l})) \tag{4}$$

where $\mathbf{z}_{i,l} = \mu_i + \sigma_i \odot \epsilon_l$ and $\epsilon_l \sim \mathcal{N}(0, \mathbf{I})$.

We can compute μ and σ with deterministic encoder network. Bernoulli cross-entropy loss function with deterministic decoder network can be used to compute $\log p_\theta(\mathbf{x}|\mathbf{z})$. More details can be found in [16].

4.1.2 The Network Architecture

The detailed structure of VAE network can be found in Fig. 2 (*i.e.* the Positive Sample Generation Network). We extract the target object from video frames and resize the resolution into 64 × 64. We reshape each target object into a vector and take it as the input of VAE network whose dimension is $12288 (64 \times 64 \times 3)$. The dimension of latent code and intermediate fully connected layer is 2 and 512, respectively. The output dimension is 12288 and the reconstructed image can be obtained by reshape this vector into

$64 \times 64 \times 3$. In addition, the convolutional architecture can also be a good choice for the design of VAE network.

To obtain a relatively purer manifold, we implement this part separately, in other word, we train one VAE model for each train video using extracted target object. The optimization algorithm is RMSprop [20], learning rate is 0.001, the mini-batch size is the number of target objects, and we training each model for $20,000$ epoches. We visualize some video sequence to demonstrate the effectness of the manifold construction in Fig. 3.

4.2. The Hard Positive Transformation Network

We propose the Hard Positive Transformation Network (HPTN) to create occlusions on the target objects using image patch extracted from background. There are two problems we need to consider here, *i.e.* i). which part of target object should we occlude ? ii). what do we use to occlude the target object ?

The two problems have one common properties: they all need to locate one image region (one for target object to occlude and one for background to extract). This process can be achieved by taking the selection of image patch as the sequential decision making process of a goal-directed agent interacting with a visual environment. At each point in time, the agent observes the environment (*i.e.* the observed region) and chooses one action to execute. After implementation of these sequential actions, the agent can finally locate one region by choosing *terminate action*. At each step, the agent receives a scalar reward (which depends on the actions the agent has executed and can be delayed), and the goal of the agent is to maximize the total sum of such rewards. We cast the problem as a Markov Decision Process (MDP), that provides a framework to model decision making when outcomes are partly uncertain.

4.2.1 Occlusion Region Localization as Dynamic Decision Process

In order to understand the models for the critical area localization task that we have developed, we first define how the Markov Decision Process is parameterized.

State. The state is the descriptor of current region. We resize the current region into 224×224 and extract the feature from the 8-th layer of VGG network as the state.

Actions. There are two types of possible actions: movement actions that imply a change in the current observed region, and the terminal action to indicate that the occlude location is found and terminate the search process. In this paper, we define 8 movement actions and one terminal action, as shown in Figure 4.

Reward. The target of agent is to obtain the maximum rewards, thus, the design of reward function will be the key to success of the learned policy. We assume the predicted score of occluded target object at time t as S_t and previous score as S_{t-1}. The reward for the movement actions can be computed as:

$$R_a(s, s') = \begin{cases} +1, & if \quad S_t - S_{t-1} < 0 \\ -1, & else \end{cases} \tag{5}$$

where s and s' are current and next state, respectively.

The trigger action does not has the next state, hence, we design another reward function for the *trigger action*:

$$R_t(s, s') = \begin{cases} +\eta, & if \quad S \leq \phi \\ -\eta, & else \end{cases} \tag{6}$$

where ϕ is a pre-defined thresholding parameter. This function denotes that if the agent choose the *trigger action*, we will compute the final predicted score of positive sample by visual tracker. If the predicted score below a threshold ϕ, we give a positive reward $+\eta$ to the agent; otherwise, a negative reward η.

4.2.2 The Training of Deep Q-Network

The parameters of deep Q-network are initialized randomly. The agent is setted to interact with the environment in multiple episodes, each representing a different training image. We also take a ϵ-greedy to train the Q-network, which gradually shifts from exploration to exploitation based on the value of ϵ. When exploration, the agent selects actions randomly to observe different transitions and collects a varied set of experienes. During exploitation, the agent will choose actions according to the learned policy and learn from its own successes and mistakes.

The utilization of target network and experience replay in DQN algorithm is the key ingredient of their success. The target network with parameters θ^- is copied every τ steps from online network and kept fixed on all other steps, thus, we could have $\theta_i^- = \theta_i$. The target in DQN can be described as the following formulation:

$$Y_i^{DQN} \equiv r + \gamma max_{a'} Q(s', a'; \theta_i) \tag{7}$$

A replay memory is used to store the experiences of the past episodes, which allows one transition to be used in multiple model updates and breaks the short-time strong correlations between training samples. Each time Q-learning update is applied, and a mini-batch randomly sampled from the replay memory is used as the training samples. The update for the network weights at the i_{th} iteration θ_i given transition samples (s, a, r, s') is as follows:

$$\theta_{i+1} = \theta_i + \alpha(r + \gamma max_{a'} Q(s', a'; \theta_i) - Q(s, a; \theta_i)) \nabla_{\theta_i} Q(s, a; \theta_i). \tag{8}$$

where a' represents the actions that can be taken at state s', α is the learning rate and γ is the discount factor.

The pseudo-code for training the hard positive sample generation network can be found in Algorithm 1.

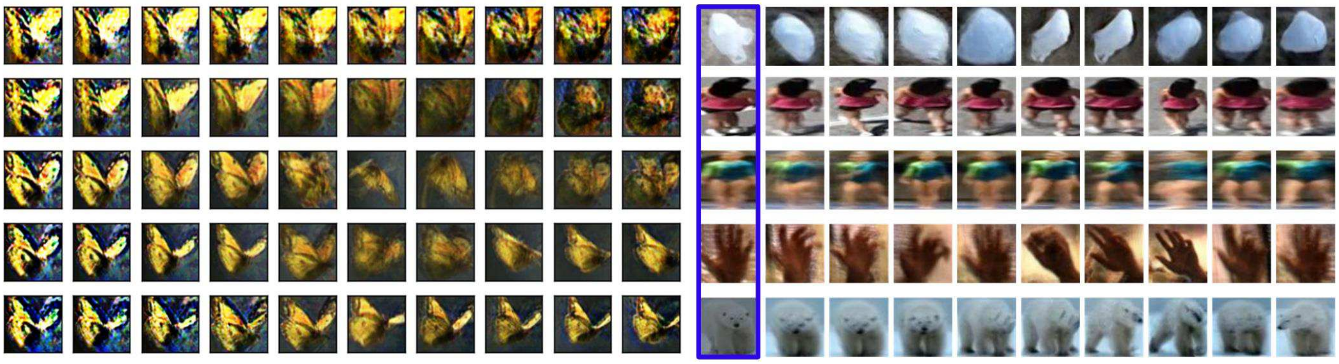


Figure 3. The targe object manifold constructed based on video *butterfly* from VOT dataset and the sampled images with our proposed methods on other videos are shown in left and right column respectively (the column with blue rectangle are target object in corresponding videos and best viewed in color).

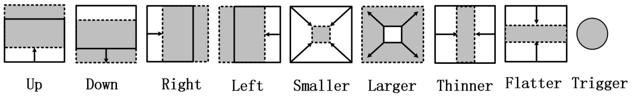


Figure 4. Illustration of the actions in the proposed MDP, giving 4 degrees of freedom to the agent for transforming boxes.

5. Experiments

In this section, we will first discuss the datasets utilized in this paper for training/testing and the evaluation criterion. We will compare our proposed algorithm with other state-of-the-art trackers on three benchmarks datasets. We also introduce the ablation studies to further analyze the proposed algorithm. Finally, we discuss the difference of our algorithm from related works.

5.1. Experimental Settings

Training Dataset. We conduct our experiments on three public visual tracking benchmarks datasets, *i.e.* OTB-50 [44], OTB-100 [45] and VOT-2014 [19]. For the evaluation of OTB-50 and OTB-100 dataset, we adopt some annotated videos from VOT-2013 [19], 2014 [19] and 2016 [18] datasets to train the introduced two streaming Siamese network. To quickly validate the effectiveness of our proposed hard positive transformation network, we only implement this experiments on 10 videos from VOT dataset , the video list can be found here [2]. It is also worthy to note that, *the videos occured in the test dataset are all removed from the training dataset for fair comparison*. For the evaluation of VOT-2014 dataset, we adopt 15 videos from the challenge competition of ImageNet Video Object Detection [34] for training.

Evaluation Criterion. Two widely used evaluation protocols are utilized in this paper: ***success rate*** and ***precision rate***. These two criterions are all aiming at measure the percentage of successfully tracked frames. For the success rate, a frame is declared to be successfully tracked if the estimated bounding box and the groundtruth box have an interaction-over-union overlap larger than a certain threshold. For precision rate, tracking on a frame is considered successful if the distance between the center of the predicted box and the groundtruth box is under some threshold.

Implementation Details. Our SINT++ is implemented based on Caffe and Keras, and all the experiments are executed on desktop with Unbuntu 14.04, I7-6700k, 32G RAM and GPU NVIDIA GTX1080. The learning rate for the Siamese network is 0.0001, mementum is 0.9 and weight decay parameter is 0.0005. For the HPTN, the mini-batch size is 100, optimization method is Adam [15], and initial learning rate is 1e-6.

5.2. Evaluation on Tracking Benchmarks

We compare the proposed SINT++ with state-of-the-art trackers including MDNet [29], SOWP [14], DSST [8], DGT [2], ReGLe [22], CCOT [10], ECO [7], ADNet [48], CSRDCF [25], SRDCF [9], *et al.* The tracking results of these trackers can be found in Fig. 5 and Fig. 6 (sub-figure (a) and (b)). We will give a detailed observations and conclusions on these benchmarks datasets, respectively.

For the OTB-100 dataset, we can find that the proposed SINT++ (Our) achieves better tracking performance compared with baseline trackers (SINTVOT) according to our experimental results. It fully validates the effectiveness of hard positive samples generated by the proposed algorithm. Although limited by the adopted baseline (Siamese network based tracker), the overall tracking results still achieve comparable performance compared with other state-of-the-art visual tracker. From the point of view of efficiency, our visual tracker achieves similar performance compared with SINT [40]. This is because the hard positive sample generation is all implemented in the training phase, and does not increase the time cost when testing (tracking). It is also

[2] `bag,ball,ball1,ball2,bicyle,birds1,birds2,blanket,bmx,book`

Algorithm 1 The Training of Deep Q-Network for Hard Positive Samples Generation.

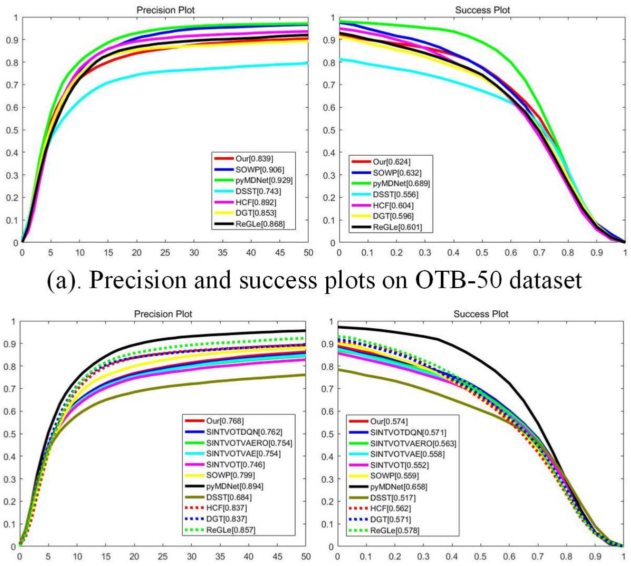
```
Input: Target objects $\{f_1, f_2, ..., f_N\}$, initial mask location $p_0$, pre-trained SINT.
Initialize replay memory $\mathcal{D}$ to capacity $\mathcal{N}$
Initialize action history $H$
Initialize action-value function $Q$ with random weights $\theta$
Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
for episode = $1, M$ do
    for target objects $f_2$ to $f_N$ do
        Initialise sequence $s_1 = (O_t, H)$ and preprocessed sequence $\phi_1 = \phi(s_1)$
        for step $t = 1, T$ do
            if random numbter $\delta < \epsilon$ then
                select a random action $a_t$
            else
                select $a_t = \arg\max_a Q(\phi(s_t); a; \theta)$
            end if
            Execute action $a_t$ to move the bbox $x_t$ and observe reward $r_t$ and new location $x_{t+1}$
            Set $s_{t+1} = s_t$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
            Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
            Sample random mini-batch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
            if episode terminates at step $j + 1$ then
                $y_j = r_j$
            else
                $y_j = r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-)$
            end if
            Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the network parameters $\theta$
            reset $\hat{Q} = Q$ for every C steps
        end for
    end for
end for
```

worthy to note that this hard positive sample generation algorithm can also combined with other strong visual trackers, such as MDNet [29]. We leave this for our future works.

For the tracking results on OTB-50 and VOT-2014 datasets, we can draw similar conclusions as mentioned above, and detailed results can be found from Fig. 5 (sub-figure (a)) and Fig. 6 (sub-figure (b)), respectively.

5.3. Ablation Study

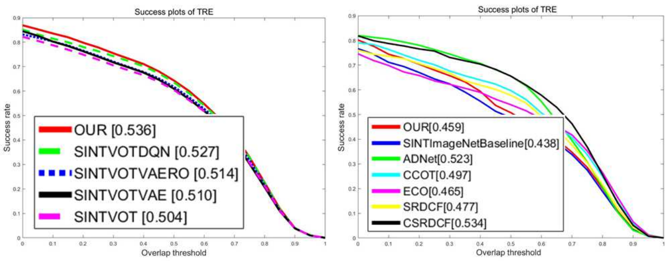
To better demonstrate the effectiveness of proposed hard positive sample generation network, we give a detailed analysis about the PSGN and HPTN in the following subsections.



(a). Precision and success plots on OTB-50 dataset

(b). Precision and success plots on OTB-100 dataset

Figure 5. Tracking Results on OTB-50 and OTB-100 dataset.



(a). TRE (Success Plots) on OTB-100

(b). OPE (Success Plots) on VOT-2014

Figure 6. TRE results on OTB-100 dataset (sub-figure(a)) and OPE results on VOT-2014 dataset (sub-figure(b)).

The Analysis of PSGN. To illustrate the effectiveness of this network, we remove the HPTN, that is to say, only the PSGN remained to check the tracking performance.

As shown in Fig. 3, we construct the manifold according to given training video frames and sample massive diverse positive samples to training the neural network. This figure fully demonstrates the effectiveness of the manifold constructed by the HPGN from the angle of qualitative analysis. On the other hand, we also perform experiment with the generated positive samples to validate the effectiveness of this network from the view of quantitative analysis. As illustrated in Fig. 5 and Fig. 6, the visual tracker (SINTVOTVAE) achieves better tracking performance compared with baseline tracking results SINTVOT on the used tracking benchmark datasets. Hence, it is an interesting and effective way to introduce the positive sample generation mechanism to bridge the gap between data-hunger deep neural networks and the scarcity of diverse positive samples in practical tracking process.

The Analysis of HPTN. To validate the effective of HPTN, we train the SINT with data generated from PSGN and HPTN. The occluded positive samples generated by HPTN will further increase the robustness of visual tracker. Because the occlusion is sparse in normal tracking videos but acturally dangerous issue which often lead to tracking failure. As shown in Fig. 5 and Fig. 6 (sub-figure (a)), we can find that the tracking performance indeed improves by this policy (The overall performance 0.768/0.574 is better than SINTVOTVAE 0.754/0.558).

In addition, we also compare our HPGN with one naive method, *i.e.* randomly selecting the occlusion region. According to our experimental results in Figure 5 and Figure 6 (sub-figure (a)), we can find this naive method (SINTVOTVAERO) achieves improvement compared with baseline tracking result, but worse than our method (Our). In one word, the proposed hard positive generation network is efficient and effective to improve the robustness of visual trackers.

5.4. Comparisons with Adversarial Networks

Our method is related with A-Fast-RCNN [43], which is a state-of-the-art object detection algorithm trained with adversarial hard positive samples. They use adversarial spatial dropout network and spatial transformer network to improve the detection performance of Fast-RCNN [11]. Our method has the following advantages compared with theirs: Firstly, we can sample more positive data through the traversing on the constructed manifold. These sampled data may not exist in the training dataset at all. Due to the nature of VAE, the reconstructed samples are blur compared with original target which makes the tracker robust to motion blur. They utilize spatial transformation network (STN) to generate hard samples, however, it is still unclear if this is a reliable generative model for producing new training samples as illustrated in [41]. Secondly, we treat the hard sample generation as desion-making problem optimized with deep reinforcement learning technique which could generate various occlusion shapes. This makes our algorithm could handling different shapes of target object in practical tracking process. They directly drop out the features in the occlusion region, which does not fit natural occlusion. The features after drop out and features extracted from occluded images are different as the input to the convolutional filters is different. Hence, our occlusion method is closer to the natural occlusion case.

To demonstrate the advantages of our method, we conduct extra experiment on adversarial network (STN + Random Occlusion), *i.e.*, we train SINT with hard positive samples generated from this network. As shown in Table 1, we can find that our method achieved better results compared with adversarial network. In short, our algorithm could achieve better tracking performance than normal data augmentation.

Table 1. Tracking performance of Adversarial-Network (Adv-Net for short) and the proposed method on public tracking benchmark OTB50 and OTB100 dataset. The precision plot and success plot are lie on the left and right of oblique line.

Methods	OTB-50 (%)	OTB-100 (%)
Adv-Net	83.7/62.2	76.0/56.7
Our	83.9/62.4	76.8/57.4

6. Conclusion

In this paper, we propose an novel and efficient hard positive sample generation network to bridge the gap between data-hunger deep neural networks and scanty of positive samples in practical tracking process. Specifically speaking, our proposed algorithm contains three main subnetworks, *i.e.* positive sample generation network (PSGN), hard positive transformation network (HPTN) and a two streaming Siamese network. In the training phase of the proposed algorithm, PSGN is used to construct manifold of each tracking video sequence and massive positive samples could be sampled from this manifold. This method will greatly enrich the positive samples from the angle of sample diversity compared with traditional dense sampling method. HPGN is utilize to generate hard occluded samples, we treat it as decision-making problem and optimize this network by deep reinforcement learning technique. Together with these hard positve samples generation strategy, we train the Siamese network to implement the tracking process based on the framework of *tracking-by-detection*. Extensive experiments on public visual tracking benchmark datasets demonstrate the effectiveness of proposed algorithm.

7. Acknowledgments

This work was in part supported by the National Natural Science Foundation of China under Grants 61702002, 61602006 and 61472002, in part by the Natural Science Foundation of Anhui Higher Education Institution of China under Grants KJ2017A017.

References

[1] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr. *Fully-Convolutional Siamese Networks for Object Tracking*. Springer International Publishing, 2016. 2

[2] Z. Cai, L. Wen, J. Yang, Z. Lei, and S. Z. Li. Structured visual tracking with dynamic graph. In *Asian Conference on Computer Vision*, pages 86–97. Springer, 2012. 6

[3] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. pages 2488–2496, 2015. 3

[4] Q. Cao, L. Lin, Y. Shi, X. Liang, and G. Li. Attention-aware face hallucination via deep reinforcement learning. *arXiv preprint arXiv:1708.03132*, 2017. 3

[5] J. Choi, J. Kwon, and K. M. Lee. Visual tracking by reinforced decision making. *arXiv preprint arXiv:1702.06291*, 2017. 2

[6] Z. Cui, S. Xiao, J. Feng, and S. Yan. Recurrently target-attending tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1449–1458, 2016. 2

[7] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. 2016. 6

[8] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014. 6

[9] M. Danelljan, G. HGer, F. S. Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *IEEE International Conference on Computer Vision*, pages 4310–4318, 2015. 6

[10] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488, 2016. 2, 6

[11] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 8

[12] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *International Conference on Neural Information Processing Systems*, pages 2672–2680, 2014. 2

[13] D. Held, S. Thrun, and S. Savarese. *Learning to Track at 100 FPS with Deep Regression Networks*. Springer International Publishing, 2016. 2

[14] H. U. Kim, D. Y. Lee, J. Y. Sim, and C. S. Kim. Sowp: Spatially ordered and weighted patch descriptor for visual tracking. In *IEEE International Conference on Computer Vision*, pages 3011–3019, 2016. 6

[15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 4

[17] D. P. Kingma and M. Welling. Auto-encoding variational bayes. 2013. 2

[18] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. ehovin, T. Vojr, G. Hger, A. Lukei, and G. Fernndez. The visual object tracking vot2016 challenge results. 8926:191–217, 2016. 6

[19] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, and A. Gatt. The visual object tracking vot2013 challenge results. In *IEEE International Conference on Computer Vision Workshops*, pages 98–111, 2014. 6

[20] T. Kurbiel and S. Khaleghian. Training of deep neural networks based on distance measures using rmsprop. 2017. 5

[21] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang. Photo-realistic single image super-resolution using a generative adversarial network. 2016. 2

[22] C. Li, X. Wu, Z. Bao, and J. Tang. Regle: Spatially regularized graph learning for visual tracking. pages 252–260, 2017. 6

[23] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Transactions on Image Processing*, 25(4):1834–1848, 2016. 2

[24] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel. A survey of appearance models in visual object tracking. *ACM transactions on Intelligent Systems and Technology (TIST)*, 4(4):58, 2013. 2

[25] A. Lukezic, T. Vojir, L. C. Zajc, J. Matas, and M. Kristan. Discriminative correlation filter with channel and spatial reliability. *International Journal of Computer Vision*, pages 1–18, 2016. 6

[26] C. Ma, J. Huang, X. Yang, and M. Yang. Hierarchical convolutional features for visual tracking. 2015. 2

[27] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016. 3

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–33, 2015. 3

[29] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. pages 4293–4302, 2015. 6, 7

[30] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. 2016. 2

[31] A. V. D. Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu. Conditional image generation with pixelcnn decoders. 2016. 2

[32] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang. Hedged deep tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4303–4311, 2016. 2

[33] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 4

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6

[35] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. 2017. 2

[36] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 3

[37] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017. 3

[38] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 36(7):1442–68, 2014. 2

[39] K. Sohn, X. Yan, and H. Lee. Learning structured output representation using deep conditional generative models. In *International Conference on Neural Information Processing Systems*, pages 3483–3491, 2015. 2

[40] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1420–1429, 2016. 2, 4, 6

[41] T. Tran, T. Pham, G. Carneiro, L. Palmer, and I. Reid. A bayesian data augmentation approach for learning deep models. *arXiv preprint arXiv:1710.10564*, 2017. 2, 8

[42] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision*, pages 835–851, 2016. 2

[43] X. Wang, A. Shrivastava, and A. Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. 2017. 1, 2, 8

[44] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013. 6

[45] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 2, 6

[46] X. Yan, J. Yang, K. Sohn, and H. Lee. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*, pages 776–791, 2016. 2

[47] A. Yilmaz. Object tracking: A survey. *Acm Computing Surveys*, 38(4):81?3, 2006. 2

[48] S. Yun, J. Choi, Y. Yoo, K. Yun, and Y. C. Jin. Action-decision networks for visual tracking with deep reinforcement learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1349–1358, 2017. 6

[49] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[50] Z. Zhang, Y. Song, and H. Qi. Age progression/regression by conditional adversarial autoencoder. 2017. 3