Structured Siamese Network for Real-Time Visual Tracking

Yunhua Zhang[0000−0003−3567−215X], Lijun Wang[0000−0003−2538−8358], Jinqing Qi[0000−0002−3777−2405], Dong Wang[0000−0002−6976−4004], Mengyang Feng[0000−0002−7112−4655], and Huchuan Lu[0000−0002−6668−9758]

School of Information and Communication Engineering, Dalian University of Technology, China
{zhangyunhua,wlj,mengyang_feng}@mail.dlut.edu.cn
{wdice,jinqing,lhchuan}@dlut.edu.cn

Abstract. Local structures of target objects are essential for robust tracking. However, existing methods based on deep neural networks mostly describe the target appearance from the global view, leading to high sensitivity to non-rigid appearance change and partial occlusion. In this paper, we circumvent this issue by proposing a local structure learning method, which simultaneously considers the local patterns of the target and their structural relationships for more accurate target tracking. To this end, a local pattern detection module is designed to automatically identify discriminative regions of the target objects. The detection results are further refined by a message passing module, which enforces the structural context among local patterns to construct local structures. We show that the message passing module can be formulated as the inference process of a conditional random field (CRF) and implemented by differentiable operations, allowing the entire model to be trained in an end-to-end manner. By considering various combinations of the local structures, our tracker is able to form various types of structure patterns. Target tracking is finally achieved by a matching procedure of the structure patterns between target template and candidates. Extensive evaluations on three benchmark data sets demonstrate that the proposed tracking algorithm performs favorably against state-of-the-art methods while running at a highly efficient speed of 45 fps.

Keywords: Tracking, deep learning, siamese network

1 Introduction

Single object tracking is a fundamental problem in computer vision, where the target object is identified in the first video frame and successively tracked in subsequent frames. Although much progress has been made in the past decades, tremendous challenges still exist in designing a robust tracker that can well handle significant appearance changes, pose variations, severe occlusions, and background clutters with real-time speed.

A survey [18] has investigated recent deep neural networks (DNNs) for visual tracking by either finetuning pre-trained deep models online [34, 35, 28, 32] or directly utilizing pre-trained deep features to characterize the targets [10, 7, 21, 30, 29]. Though promising performance has been reported, these methods only exploit the holistic model for target representation and ignore detailed information.

The above issues are mostly handled by part-based models in traditional methods [19, 20]. Rather than describing the entire object with a single global model, the part-based approaches divide the target region into a number of fixed rectangular patches and are capable of capturing the local patterns of the target. As a consequence, they are more flexible in handling non-rigid appearance variations. Nonetheless, these methods have their own drawbacks. On the one hand, these methods process local patterns independently and fail to leverage their structure relationship, giving rise to noisy and inaccurate prediction. On the other hand, these methods mostly rely on hand-crafted features. It is still very rare to explore local models using deep learning techniques due to the high computational overhead involved in deep feature extraction for multiple local regions.

To address the above issues, this paper proposes a new structure constrained part-based model for visual tracking using DNNs. As opposed to prior part-based trackers, our method does not explicitly divide the target into parts. Instead, we identify object parts with discriminative patterns using a local pattern detection module, which are more computationally efficient. To enforce the structural relationship among local patterns, the predicted local patterns are further refined by considering contextual information from correlated patterns using a message passing module. For a more principled solution, we formulate the message passing module as the inference process of a CRF, which can be effectively implemented using differentiable operations and embedded into a neural network. As a result, the entire model can be trained in an end to end manner for online tracking, such that the local pattern detection module can learn to automatically identify the key object parts while the message passing module learns to encode the structure relationships among detected patterns. The target tracking is finally achieved through template-candidate matching over the detected local patterns using a Siamese network architecture.

Our method has three advantages over existing DNN based trackers. Firstly, our method performs on the object part level, and is therefore more flexible in handling non-rigid appearance change and partial occlusion. Meanwhile, owing to the local pattern detection module, our method is highly efficient and runs at a real time speed of 45 fps. In addition, our method can effectively leverage the structure context among local patterns, yielding more accurate target detection.

The main contribution of this paper can be summarized as follows: i) We propose a local pattern detection scheme, which can automatically identify discriminative local parts of target objects. ii) We implement the message passing process via differentiable operations, and reformulate it via a neural network module. By doing this, our network can simultaneously learn the local patterns

and the relationships among local patterns in an end-to-end manner. This yields more accurate tracking results. iii) A new matching framework based on the Siamese network is proposed, which successively applies and ensembles the new techniques, and runs at a real time speed. Extensive evaluations performed on three widely adopted benchmarks show that the proposed method performs favorably against state-of-the-art methods in terms of both tracking accuracy and efficiency.

2 Related Work

This section reviews existing tracking methods that are mostly related to ours.

Tracking by discriminative appearance modeling: One simple yet effective manner of using deep networks for visual tracking is to directly apply correlation filters on the multi-dimensional feature maps of deep Convolutional Neural Networks (CNNs), where the pre-trained CNN model is fixed. Recently, Danelljan et al. [10] have introduced a continuous spatial domain formulation named C-COT, allowing effective integration of multi-resolution deep features. C-COT and its improved version ECO [7] have achieved top performance in the VOT challenge [17], but they are not suitable for real-time applications as the tracking speed is rather slow. Another category of deep trackers [34, 35, 24] update a pre-trained CNN online to account for the target-specific appearance at test time. For instance, Wang et al. [34] proposes a feature map selection scheme and predicts a response map for the target with a heavily online updating schedule. However, these methods [34, 35, 24] rely on computationally inefficient search algorithms, such as sliding window or candidate sampling, which significantly reduce their applicability in real time scenarios. Meanwhile, they also highly rely on online updates, which are computationally inefficient and not desirable for real-time tasks.

Tracking by Siamese Network: Siamese network based trackers [31, 3] select target from candidate patches through a matching function learned offline on image pairs. The matching function is usually implemented by two-branch CNNs with tied parameters, which takes the image pairs as input and predicts their similarity. Although SiamFC [3] can run beyond real-time, its tracking accuracy is still inferior to state-of-the-art trackers, due to the lack of online adaptation ability. Despite SINT [31] achieves higher tracking accuracy, it adopts optical flow to facilitate candidate sampling and is much slower (about 2 fps) than SiamFC. Recently, the DSiamM tracker [11] proposes to perform online update of siamese network by integrating correlation filters into the network. In [14], a policy is learned to decide whether to locate objects on early layers to speed up the tracking process. Though we also adopt the Siamese network architecture for tracking, our method significantly differs from existing methods in that ours is able to automatically detect local patterns of target appearance and models their structure relationships. Experiments confirm that our method can better handle challenge cases like drastic appearance change, partial occlusion, and rotation.

Part-based Trackers: These days, the method to track non-rigid object has attracted great attention. Since common trackers can barely deal with extreme deformations, some trackers aim at this task try to exploit part information and achieve promising performance. In [27], an online gradient boosting decision tree operating on individual patches is intergraded. [38] uses Markov Chain on superpixel graph, but information propagation through a graph could be slow depending on the structure. Both Ting et al. [20] and Yang et al. [19] propose the patch-based trackers based on correlation filter and combine the patches within a particle filter framework. However, these methods separately learn the correlation filter for each part and record the relative positions between each part and the target center. In addition, the existing patch based trackers rigidly divide the target object into fixed number of fragments. Discriminative local structure fails to be maintained by such rough rigid patch dividing, and features of such patches contain little semantic information. Reasonable updating strategies of such methods are hard to design and easy to drift due to drastic deformation changes.

Conditional Random Field for Image Segmentation: Conditional random field (CRF) has been widely used in image segmentation task [4, 40, 16]. They utilize CRF to establish pairwise potentials on all pairs of pixels in the image to exploit interactions of pixels. The method [16] develops a fully connected pairwise CRF with efficient computation to capture fine edge details while also catering for long range dependencies. This model was shown to largely improve the performance of a boosting-based pixel-level classifier. Chen et al. [4] uses a CRF to refine segmentation results obtained from a CNN and Zheng et al. [40] embeds the CRF inference procedure into the network and enables end-to-end training. They all use CRF to capture interactions of pixels and achieve state-of-the-art performance in image segmentation task. Inspired by their approaches, we adopt CRF inference to model the contextual information of local patterns by message passing, and capture the structure of object to enhance the robustness of the tracker. Contrary to adopt fixed Gaussian kernels to formulate pairwise terms as in their work, we use learnable convolution kernels to model pairwise terms, which can better encode object local patterns.

3 Structured Siamese Network

3.1 Overview

In this work, we propose a structured siamese network, which simultaneously performs discriminative pattern detection, local structure learning and integration in an end-to-end manner. Figure 1 overviews the pipeline of our tracking algorithm. The two-stream siamese network [3] is trained offline to locate a 127×127 template image z within a larger 255×255 search image x. A similarity function is learned to densely compare the template image z to each candidate region of the same size in the search image x, so as to predict a score map that highlights the target region. Specifically, a cross-correlation layer is proposed to compute

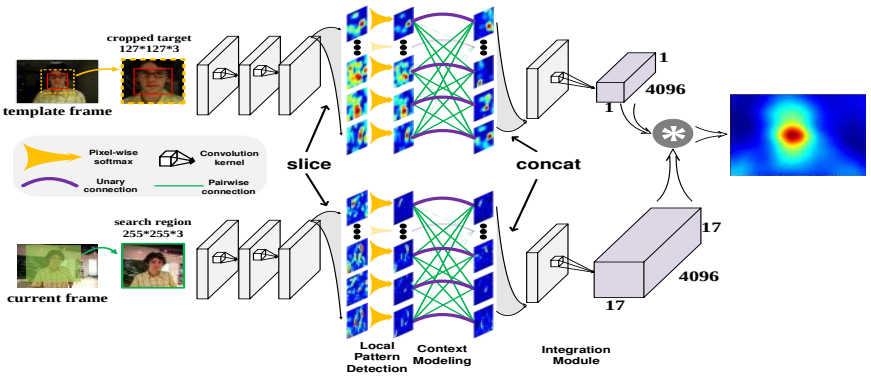


Fig. 1. The pipeline of our StructSiam algorithm.

the similarity for all translated sub-regions in x in one pass:

$$F(z, x) = \varphi(z) * \varphi(x) + v, \tag{1}$$

where φ is the convolutional feature embedding generated by each network stream; $v \in \mathbb{R}$ denotes the bias; and $F(\cdot, \cdot)$ represents the predicted confidence score map of size 17×17.

The two streams of the network share the same architecture and parameters, consisting of three components: local pattern detector, context modeling module and integration module. The details of these components are presented in details in the following sections.

The final cross-correlation operation is based on the obtained maps, which we call structure patterns.

Training the network adopts the logistic loss:

$$L = log(1 + e^{-yv}), \tag{2}$$

where v is the real-valued score of a single template-candidate pair and $y \in \{+1, -1\}$ is its ground-truth label as in [3].

3.2 Informative Local Pattern Detector

Informative local patterns are crucial cues to characterize target appearance. Instead of manually dividing the target region into pre-fixed parts, we design the local pattern detector to automatically identify discriminative patterns through end-to-end training.

The local pattern detector comprises two convolutional layers with kernel size of 11×11 and 5×5 respectively. Each of these convolutional layers is followed

by a batch normalization layer [15], a ReLU layer [23], and a 3×3 max pooling layer. The module takes the image crop as input and detects local patterns of target appearance. The output feature map has 256 channels, each of which corresponds to a specific local pattern. Softmax layer is adopted to normalize the output feature map across channels.

As opposed to features in deeper layers that have low resolution and limited detailed information, the proposed local pattern detector, with only two convolutional layers and two max pooling layers, has a relatively small receptive field. Therefore, it can better focus on local regions of the target and preserve more detailed information (see the visualized results of the local pattern detector module in Figure 1 as examples). This design is also consistent with recent findings [10, 7] in visual tracking that detailed low level features are more discriminative and suitable for target matching. However, the local pattern detector has a major drawback. Since individual patterns are independently detected by different channels of the output feature maps, the structure relationships between these local patterns are mostly ignored. As a consequence, the detection results may be inaccurate and vulnerable to background noise. Based on this observation, we introduce the context modeling module for refinement.

3.3 Context Modeling Module

Generally, our local pattern detector tends to capture local patterns like heads, legs and torsos of persons, wheels of cars or bicycles, and regions with significant edges (we will show examples in Figure 4 in section 4). They are common in visual tracking tasks, and their appearances can be significantly different for various targets, sequences and time. Thus, embedding prior knowledge on these generic local patterns into the network is benefit for target recognition during tracking. We regard the prior knowledge as the relationships among local patterns. When the tracked object is undergoing cluttered background or drastic appearance change, the detection result of each single local pattern is not reliable. Thus the relationships between different local patterns (i.e., context information) should be considered to facilitate the detection process. The context information incorporation is achieved by message passing, which can enforce the responses of regions that are highly structural, and suppress the noisy background responses. To implement the message passing process efficiently, we introduce conditional random field (CRF) approximation into our network. We formulate the target's local pattern detection problem by using a graph and model the joint probability relationships among local patterns generated by previous stage through CRF.

Let X_i be the random variable associated to pixel i, which represents the type of local pattern assigned to the pixel i and can take any value from a pre-defined set $\mathcal{P} = \{p_1, p_2, ..., p_c\}$, and c is the channel size of feature maps. We regard each channel represents a specific local pattern. Let X be the vector formed by the random variables $X_1, X_2, ..., X_N$, where N is the number of pixels in the feature map. Given a graph $G = (V, E)$, where $V = \{X_1, X_2, ..., X_N\}$, and a global observation (image) I, the pair (I, X) can be modeled as a CRF characterized by a Gibbs distribution of the form $P(X = x|I) = \frac{1}{Z(I)} e^{-E(x|I)}$. Here $E(x)$ is

called the energy of the configuration $x \in \mathcal{L}^N$ and $Z(I)$ is the partition function. From now on, we drop the conditioning on I in the notation for convenience.

The energy of a label assignment x is given by:

$$E(x) = \Sigma_i \psi_u(x_i) + \Sigma_i \Sigma_{j=max(0,i-R)}^{min(N-1,i+R)} \psi_p(x_i, x_j), \tag{3}$$

where the unary energy component $\psi_u(x_i)$ measures the inverse likelihood (and therefore, the cost) of the pixel i is subordinate to the local pattern x_i, and pairwise energy component $\psi_p(x_i, x_j)$ measures the cost of assigning local pattern types x_i, x_j to pixels i, j simultaneously, and R is the scope of the surrounding pixels taken into consideration for pairwise energy computation. In our model, unary energies are obtained from local pattern detector's output, which predict labels for pixels without considering the smoothness and the consistency of the local pattern type assignments. The pairwise energies serve as data-dependent smoothing terms and encourage assigning correlated types to pixels with similar features. Considering the translation invariance property of natural images, we implement the pairwise energy using convolutional operations as follows:

$$\psi_p(x_i, x_j) = \Sigma_{j=max(0,i-R)}^{min(N-1,i+R)} w_{i,j} * x_j + b_i, \tag{4}$$

where the kernels $w_{i,j}$ and biases b_i for $i = 1, ..., N$, are shared for all locations in the local patterns' maps and we set $w_{i,j=i} = 0$ to prevent message passing from x_i to itself.

Minimizing the above CRF energy $E(x)$ yields the most probable local structure distribution for the input image. Since the direct minimization is intractable, we adopt the mean-field variational inference to approximate the CRF distribution $P(z)$ with the product of independent marginal distributions, i.e., $Q(z) = \prod_i Q(z_i)$. To this end, we first consider individual steps of the mean-field algorithm summarized in Algorithm 1, and implement them using differentiable operations for end-to-end training. Let $U_i(p)$ denote the negative of the unary energy, i.e., $U_i(p) = -\psi_u(X_i = p)$, where p denotes the local pattern type. In our CRF, the unary energy ψ_u is obtained directly from the output of local pattern detector.

In the first step of Algorithm 1, we initialize $Q_i(p)$ with $Q_i(p) \leftarrow \frac{1}{Z_i} e^{U_i(p)}$, where $Z_i = \Sigma_p e^{U_i(p)}$. Note that this is equivalent to applying a softmax function over the unary potentials U across all the labels at each pixel. The message passing (step 4 in Algorithm 1) is then performed by applying two 3×3 convolutional kernels on Q as described in (4). The receptive field for each output pixel is 5×5, which is $\frac{5}{6}$ of the target object (considering the output size of the target template) and is enough to model target structure. Since there is no activation layer (e.g., ReLU) between the two convolution layers, they can be used to implement the linear mapping in (4) with less parameters than one 5×5 convolution layer. The kernels are learned to encode the context structural information among local patterns. The output from the message passing stage is subtracted element-wisely from the unary input U. Finally, the normalization step of the iteration can be implemented by another softmax operation with no parameters.

Algorithm 1 Mean-field approximation.

1: Initialize Q^0, $Q_i^0(p) = \frac{1}{Z_i}(U_i(p))$ for all i
2: Initialize the iteration times L.
3: **for** $t = 1 : L$ **do**
4: $\quad \tilde{Q}_i(p) = \Sigma_{j=max(0,i-R)}^{j=min(N-1,i+R)} w_{i,j} \times Q_j^{t-1} + b_i$
5: $\quad \hat{Q}_i \leftarrow U_i(p) - \tilde{Q}_i(p)$
6: $\quad Q_i^t \leftarrow \frac{1}{Z_i} e^{\hat{Q}_i(p)}$
7: **end for**

Given the above implementation, one iteration of the mean-field algorithm can be formulated as a stack of common neural network layers. Multiple mean-field iterations can be achieved by recurrently passing the estimated probability Q through the network multiple times. In practice, we find that three iterative steps are enough to obtain a satisfying performance.

Every local pattern receives message from other patterns, which can be seen as contextual information. The contextual information indicates the structure information of the input image inherently. Consequently, the local pattern maps are effectively refined after message passing stage. The final score map is less noisy when the tracking target is undergoing cluttered background and drastic deformation challenges.

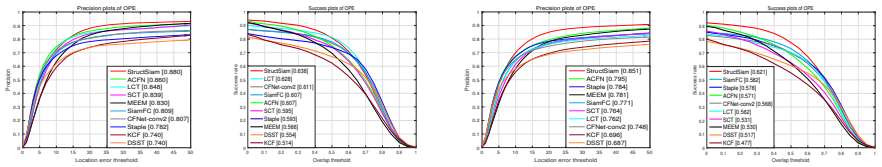
In general, by message passing among local patterns, the CRF probability model describes the universal structure information of generic objects despite the category of the target. Since all operations of the context modeling module are differentiable, the whole network can be trained in an end-to-end manner.

3.4 Integration Module

The output maps of the context modeling module are capable of capturing precise local patterns' information. Since various local patterns correspond to different positions in the target region, directly correlating the template with the search region is vulnerable to deformation. In contrast to SiameseFC [3] using features with spatial layout for correlation, our integration module aggregates the local patterns (of both target and candidates) into a 1×1 feature map, with each channel acting as an attribute to indicate the existence of certain pattern regardless of its location. In our method, feature maps corresponding to the template and search region are fed into the 6×6 convolution layer, which leads to a $1 \times 1 \times 4096$ tensor representing the template and a $17 \times 17 \times 4096$ tensor $\mathbf{T}$ representing the search region. These two tensors are correlated to obtain the final response. Clearly, each spatial position (x, y) in the search region has a corresponding $1 \times 1 \times 4096$ tensor $\mathbf{T}(x, y, :)$ which is different from others. Each pixel in the final maps indicates the local pattern information for a region, and the final correlation finds the pixel (as the center of the target) that incorporates the same local patterns as the target's. The feature variation caused by drastic deformation changes can be reduced to some extent in this way, which will be proved in section 4.

Table 1. The speed of trackers compared.

	Ours	ACFN	LCT	SCT	MEEM	CFNet-conv2	SiameseFC	Staple	KCF	DSST
speed/fps	45	15	27	40	10	75	58	80	172	24



(a) Comparisons on OTB2013 (b) Comparisons on OTB2015

Fig. 2. Comparison on OTB2013 and OTB2015 using distance precision rate (DPR) and overlap success rate (AUC).

4 Experiments

4.1 Implementation details

Training Data. Since the motivation of our network is different from SiameseFC, which aims to learn a matching function to perform metric learning, training only on ILSVRC2014 VID dataset is not suitable. Visual object tracking task is to track generic objects no matter what categories. The ILSVRC2014 VID dataset is more biased to animals contains head, limbs and torso, that will lead to ineffective learning of our structured network. And rotation objects are necessary for structure patterns learning, which aim to response to the center of structural regions. To model generic object inner structure, we use ILSVRC2014 VID dataset [25] and ALOV dataset [26]. We discard the common sequences appear in the testing datasets for a fair comparison.

Parameter Setting. We implement the proposed method in python with the Tensorflow library [1]. The parameters of the whole network are initialized randomly guided by a Gaussian distribution [12]. It is well known that softmax leads to vanishing of gradients which makes the network training inefficient. Thus, we multiply the feature maps with a constant β, which will make the network converge much faster. β is set to equal to the channel size of feature maps, i.e., $\beta = 256$. Training is performed over 50 epochs, each consisting of 60,000 sampled pairs. The gradients for each iteration are estimated using mini-batches of size 8, and the learning rate is annealed geometrically at each epoch from 10^{-2} to 10^{-5} as in [3]. We use the SGD method to conduct optimization and train the network with a single NVIDIA GeForce GTX 1080 and an Intel Core i7-4790 at 3.6GHz.

Tracking Algorithm. With the learned StructSiam, we summarize our tracking algorithm as follows: given the target location at I_1, i.e., a bounding box $b_1 \in \mathbb{R}$, we crop the corresponding region to serve as the target template O_1 that is

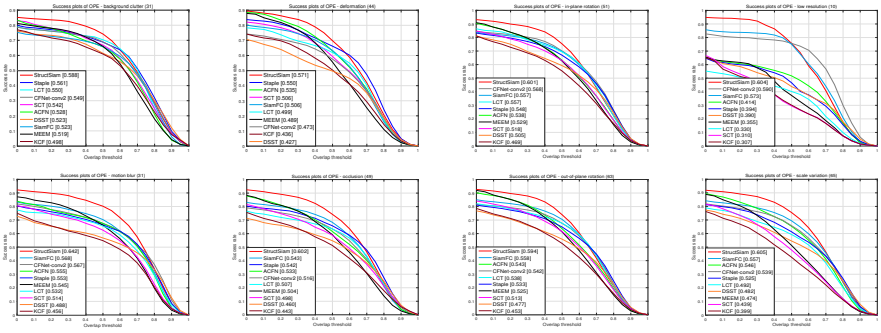


Fig. 3. The success plots over eight tracking challenges, including background clutter, deformation, in-plane rotation, low resolution, motion blur, occlusion, out-of-plane rotation and scale variation.

slightly larger than b_1 and centered at b_1. We then extract the deep features of O_1 to get F_1. When tracking at the tth frame, we crop search regions on three scales, i.e., $1.025^{\{-1,0,1\}} \times S_0$, where S_0 is the original scale, centering at b_{t-1}. Then, we get 3 response maps via (1). We search the maximum value among the three response maps and get its respective location and scale, which leads to b_t.

4.2 Experiments on OTB2013

OTB2013 [36] contains 50 fully annotated sequences that are collected from commonly used tracking sequences. We compare our tracker with other 9 state-of-art real-time trackers, including LCT [22], MEEM [39], SiameseFC [3], Staple [2], KCF [13], DSST [8], ACFN [6], SCT [5], CFNet [33]. Following the protocol in [36], we report the results in one-pass evaluation (OPE) using two metrics: precision and success plots, as shown in Figure 2(a). The precision metric computes the rate of frames whose center location is within some certain distance with the ground truth location. The success metric computes the overlap ratio between the tracked and ground truth bounding boxes. In addition, we report the area under curve (AUC) score of success plots and distance precision score at 20 pixels threshold in the precision plots for each tracking method. Overall, StructSiam performs favorably against other real-time state-of-the-art trackers on this dataset. Our tracker achieves a distance precision rate (DPR) of 87.4% and AUC score of 0.638 with real-time speed of 45 fps. Besides, it outperforms other competing real-time ones in terms of accuracy.

4.3 Experiments on OTB2015

The OTB2015 [37] dataset is the extension of OTB2013 and is more challenging. The evaluation criteria of two benchmarks are identical. Compared to the same real-time trackers mentioned above, the result is shown in Figure 2(b) and the comparison of speed is shown in Table 1. On the OTB2015 dataset, our

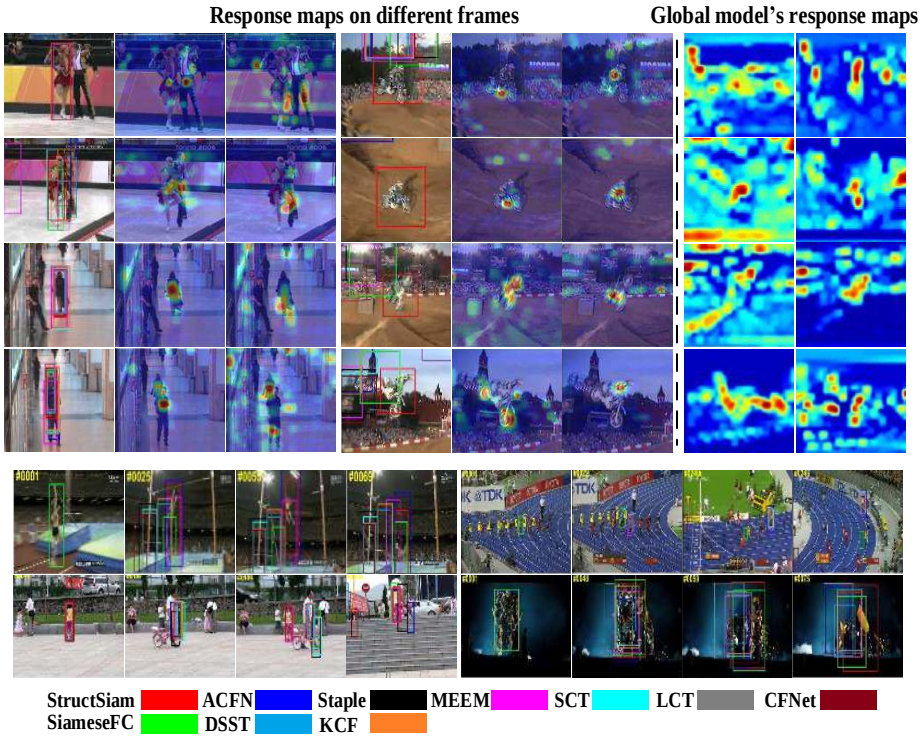


Fig. 4. Qualitative evaluation of the proposed algorithm and other state-of-the-art real-time trackers on seven sequences (Skater2, Walking2, MotorRolling, Jump, Girl2, Bolt2 and Trans.

tracker achieves a DPR of 85.1% and AUC of 62.1%. Considering both speed and accuracy, our method achieves a very competitive result.

Attribute-based evaluation. We further analyze the performance of StructSiam under different attributes in OTB2015 [37] to demonstrate the effectiveness of information passing between local structures on feature learning. Figure 3 shows the OPE plots for eight major attributes, including background clutter, deformation, in-plane rotation, low resolution, motion blur, occlusion, out-of-plane rotation and scale variation. From Figure 3, we have the following observations. First, our method is effective in handling occlusion, owing to that the relationships among local structures are implicitly modeled through message passing. In contrast, the SiameseFC method pre-trains the network only using a single global feature model as the output, which is less effective in handling partial occlusion. Second, our method performs well in the presence of motion blur, since the noisy features extracted from blurred images can be refined by the CRF module. Other trackers based on correlation filters and SiameseFC are sensitive to motion blur, this phenomenon may result from the extracted feature

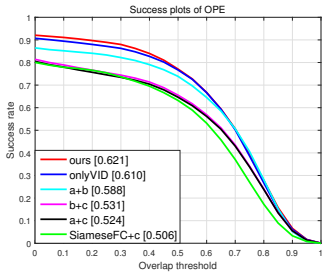


Fig. 5. Ablation study.

is destroyed when motion blur occurs. Thus, this attribute also demonstrates the effectiveness of the CRF approximation. Besides, the proposed method achieves better performance for other challenging factor, which further demonstrates that our method has good generalization ability.

Qualitative evaluation. In Figure 4, we select three representative sequences (including the challenges of deformation, occlusion and rotation) to visualize the effectiveness of our algorithm compared to other real-time trackers. Each column of the responses maps in the top of Figure 4 represents one typical channel's responses before the correlation layer across different frames and different sequences. As we can see, the selected channels are prone to response on regions around heads, legs, and torsos of persons and wheels of motorbikes. Their responses are stable with different inputs. To further compare our algorithm with global model, we show the responses of a global model (implemented with AlexNet) on the right. They are too noisy to distinguish target from background, and fail to consistently highlight the same local parts across frames after severe deformations. More results are shown in the bottom of Figure 4. Overall, the visual evaluation indicates that our StructSiam tracker performs favorably against other real-time state-of-the-art trackers.

Ablation study. Our algorithm contains local pattern detectors, messege passing layers and integration layer. We conduct ablation analysis to compare each component's effect on the performance on OTB15 dataset. As shown in the Figure 5, a, b, and c denote the local pattern detectors, message passing module and integration layer respectively, and onlyVID denotes the network that trained with only ILSVRC VID dataset. Specially, we test the performance of the network without message passing layers by replacing them with simple 3×3 convolution layers. The results show that, all components proposed in this framework are essential and complementing each other. The performance of the proposed tracker will degrade drastically if any component is removed. Due to the relatively large performance gain through integration module, we further prove that the simply embedding integration module into SiameseFC (denoted as "SiameseFC+c") leads to an inferior performance. Therefore, our performance improvement does not mainly come from the integration layer. Since SiameseFC

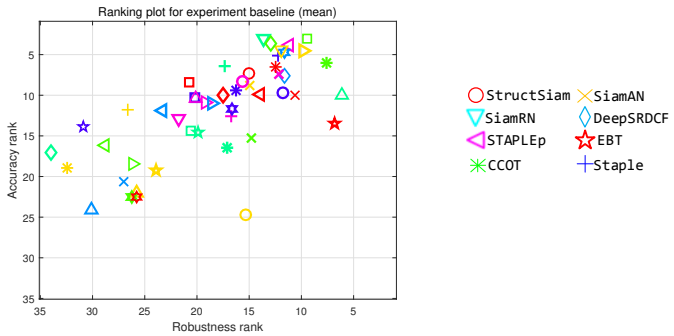


Fig. 6. Comparison on VOT2016 benchmark.

was trained with only ILSVRC VID dataset, we test the performance of our tracker without ALOV300 training dataset for a fair comparison. As we can see in the Figure 5, in this condition, our tracker performs better than SiameseFC with a large margin, which illustrates the effectiveness of our algorithm.

4.4 Experiments on VOT2016 benchmark [17]

VOT-2016 has 60 sequences and re-initializes testing trackers when it misses the target. The expected average overlap (EAO) considering both bounding box overlap ratio (accuracy) and the re-initialization times (robustness) serves as the major evaluation metric on VOT-2016. We compare our StructSiam tracker with state-of-the-art trackers on the VOT 2016 benchmark, including CCOT [10], Staple [2], EBT [41], DeepSRDCF [9] and SiamFC [3]. As indicated in the VOT 2016 report [17], the strict state-of-the-art bound is 0.251 under the EAO metric. That is, the trackers are considered as state-of-the-art ones when their EAO values exceed 0.251.

Table 2 and Figure 6 show the results from our StructSiam tracker and the state-of-the-art trackers. Among these methods, CCOT achieves the best results under the expected average overlap (EAO) metric. However, the top performance trackers are far from real-time requirements. CCOT and DeepSRDCF are less than 1 fps, and EBT only has 3 fps. Meanwhile, the performance of our StructSiam tracker is higher than SiamAN, which has the same depth of network with ours. SiamRN represents SiameseFC using ResNet as architecture and its performance is higher than ours but the speed is far more slower. It is likely to be due to the deeper network. StructSiam achieves the fastest speed and the lower speed of SiamAN may be credited to its hardware condition. According to the analysis of VOT report and the definition of the strict state-of-the-art bound, our StructSiam tracker can be regarded as a state-of-the-art method with real-time performance.

Table 2. Comparison with the state-of-the-art trackers on the VOT 2016 dataset with expected average overlap (EAO) measuring method.

	CCOT	Staple	EBT	DeepSRDCF	StructSiam	SiamAN	SiamRN
EAO	0.331	0.295	0.291	0.276	0.264	0.235	0.277
speed	0.5	11	3	0.38	16	9	5

5 Conclusion

This work presents a powerful local structure-based siamese network for visual object tracking. The proposed network can detect discriminative local patterns automatically and model the contextual information to form the structure of the object in a probability way by message passing. The final matching procedure is based on the final structural patterns of the target object, which facilitates dealing with several challenges such as drastic appearance change, rotation, partial occlusion and motion blur. The experimental results demonstrate that our tracker achieves promising performance with a real-time speed. In the future, we will extend the proposed structured Siamese network to handle other vision tasks.

6 Acknowledgement

This work was supported by the Natural Science Foundation of China under Grant 61725202, 61751212, 61771088, 61632006 and 91538201.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M.: Tensorflow: Large scale machine learning on heterogeneous distributed systems. In: arXiv preprint arXiv:1603.04467 (2016)
2. Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., Torr, P.H.S.: Staple: Complementary learners for real–time tracking. In: CVPR (2016)
3. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: Fully-convolutional siamese networks for object tracking. In: ECCV Workshop (2016)
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR (2015)
5. Choi, J., Chang, H.J., Jeong, J., Demiris, Y., Jin, Y.C.: Visual tracking using attention-modulated disintegration and integration. In: CVPR (2016)
6. Choi, J., Chang, H.J., Yun, S., Fischer, T., Demiris, Y., Jin, Y.C.: Attentional correlation filter network for adaptive visual tracking. In: CVPR (2017)
7. Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: Eco: Efficient convolution operators for tracking. In: CVPR (2017)
8. Danelljan, M., Hger, G., Khan, F.S., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: BMVC (2014)
9. Danelljan, M., Hger, G., Khan, F.S., Felsberg, M.: Convolutional features for correlation filter based visual tracking. In: ICCV Workshop (2015)
10. Danelljan, M., Robinson, A., Khan, F.S., Felsberg, M.: Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: ECCV (2016)
11. Guo, Q., Feng, W., Zhou, C., Huang, R., Wan, L., Wang, S.: Learning dynamic siamese network for visual object tracking. In: ICCV (2017)
12. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human–level performance on imagenet classification. In: ICCV (2015)
13. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. In: ICVS (2008)
14. Huang, C., Lucey, S., Ramanan, D.: Learning policies for adaptive tracking with deep feature cascades. In: ICCV (2017)
15. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
16. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected crfs with gaussian edge potentials. In: NIPS. pp. 109–117 (2011)
17. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., ehovin, L., Vojr, T., Hger, G., Lukei, A., Fernndez, G.: The visual object tracking VOT2016 challenge results. In: ECCV Workshop (2016)
18. Li, P., Wang, D., Wang, L., Lu, H.: Deep visual tracking: Review and experimental comparison. Pattern Recognition (76), 323–338 (2018)
19. Li, Y., Zhu, J., Hoi, S.C.H.: Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In: CVPR (2015)
20. Liu, T., Wang, G., Yang, Q.: Real-time part-based visual tracking via adaptive correlation filters. In: CVPR (2015)
21. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: ICCV (2015)
22. Ma, C., Yang, X., Zhang, C., Yang, M.H.: Long-term correlation tracking. In: CVPR (2015)

23. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML (2010)
24. Nam, H., Baek, M., Han, B.: Modeling and propagating cnns in a tree structure for visual tracking. In: arXiv preprint arXiv:1608.07242 (2016)
25. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3), 211–252 (2014)
26. Smeulders, A.W.M., Chu, D.M., Cucchiara, R., Calderara, S., Dehghan, A., Shah, M.: Visual tracking: An experimental survey. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1442–1468 (2014)
27. Son, J., Jung, I., Park, K., Han, B.: Tracking-by-segmentation with online gradient boosting decision tree. In: ICCV. pp. 3056–3064 (2016)
28. Song, Y., Ma, C., Gong, L., Zhang, J., Lau, R., Yang, M.H.: Crest: Convolutional residual learning for visual tracking. In: ICCV (2017)
29. Sun, C., Wang, D., Lu, H., Yang, M.H.: Correlation tracking via joint discrimination and reliability learning (2018)
30. Sun, C., Wang, D., Lu, H., Yang, M.H.: Learning spatial-aware regressions for visual tracking (2018)
31. Tao, R., Gavves, E., Smeulders, A.W.M.: Siamese instance search for tracking. In: CVPR (2016)
32. Teng, Z., Xing, J., Wang, Q., Lang, C., Feng, S., Jin, Y.: Robust object tracking based on temporal and spatial deep networks. In: ICCV (2017)
33. Valmadre, J., Bertinetto, L., Henriques, J.F., Vedaldi, A., Torr, P.H.S.: End-to-end representation learning for correlation filter based tracking. In: CVPR (2017)
34. Wang, L., Ouyang, W., Wang, X., Lu, H.: Visual tracking with fully convolutional networks. In: ICCV (2015)
35. Wang, L., Ouyang, W., Wang, X., Lu, H.: Stct: Sequentially training convolutional networks for visual tracking. In: CVPR (2016)
36. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR (2013)
37. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1834–1848 (2015)
38. Yeo, D., Son, J., Han, B., Han, J.H.: Superpixel-based tracking-by-segmentation using markov chains. In: CVPR. pp. 511–520 (2017)
39. Zhang, J., Ma, S., Sclaroff, S.: Meem: Robust tracking via multiple experts using entropy minimization. In: ECCV (2014)
40. Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H.S.: Conditional random fields as recurrent neural networks. In: ICCV (2015)
41. Zhu, G., Porikli, F., Li, H.: Beyond local search: Tracking objects everywhere with instance–specific proposals. In: CVPR (2016)