Siam R-CNN: Visual Tracking by Re-Detection

Paul Voigtlaender[1] Jonathon Luiten[1,2,†] Philip H.S. Torr[2] Bastian Leibe[1]
[1]RWTH Aachen University [2]University of Oxford
{voigtlaender,luiten,leibe}@vision.rwth-aachen.de phst@robots.ox.ac.uk

Abstract

We present Siam R-CNN, a Siamese re-detection architecture which unleashes the full power of two-stage object detection approaches for visual object tracking. We combine this with a novel tracklet-based dynamic programming algorithm, which takes advantage of re-detections of both the first-frame template and previous-frame predictions, to model the full history of both the object to be tracked and potential distractor objects. This enables our approach to make better tracking decisions, as well as to re-detect tracked objects after long occlusion. Finally, we propose a novel hard example mining strategy to improve Siam R-CNN's robustness to similar looking objects. Siam R-CNN achieves the current best performance on ten tracking benchmarks, with especially strong results for long-term tracking. We make our code and models available at www.vision.rwth-aachen.de/page/siamrcnn.

1. Introduction

We approach Visual Object Tracking using the paradigm of Tracking by Re-Detection. We present a powerful novel re-detector, Siam R-CNN, an adaptation of Faster R-CNN [54] with a Siamese architecture, which re-detects a template object anywhere in an image by determining if a region proposal is the same object as a template region, and regressing the bounding box for this object. Siam R-CNN is robust against changes in object size and aspect ratio as the proposals are aligned to the same size, which is in contrast to the popular cross-correlation-based methods [38].

Tracking by re-detection has a long history, reaching back to the seminal work of Avidan [1] and Grabner *et al.* [21]. Re-detection is challenging due to the existence of distractor objects that are very similar to the template object. In the past, the problem of distractors has mainly been approached by strong spatial priors from previous predictions [4, 38, 37], or by online adaptation [1, 21, 2, 56, 23, 57, 32]. Both of these strategies are prone to drift.

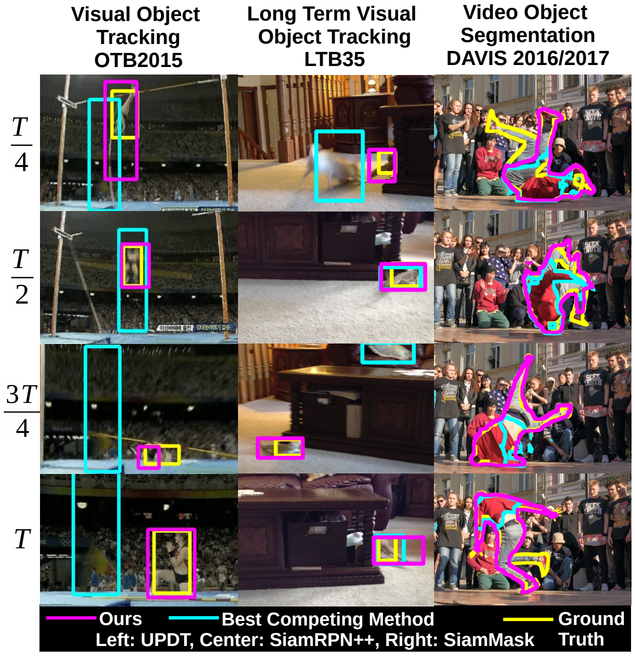
Visual Object Tracking OTB2015
Long Term Visual Object Tracking LTB35
Video Object Segmentation DAVIS 2016/2017
$\frac{T}{4}$
$\frac{T}{2}$
$\frac{3T}{4}$
T
Ours
Best Competing Method
Ground Truth
Left: UPDT, Center: SiamRPN++, Right: SiamMask

Figure 1: Example results of Siam R-CNN on 3 different tracking tasks where it obtains new state-of-the-art results.

We instead approach the problem of distractors by making two novel contributions beyond our Siam R-CNN re-detector design. Firstly we introduce a novel hard example mining procedure which trains our re-detector specifically for difficult distractors. Secondly we propose a novel Tracklet Dynamic Programming Algorithm (TDPA) which simultaneously tracks all potential objects, including distractor objects, by re-detecting all object candidate boxes from the previous frame, and grouping boxes over time into tracklets (short object tracks). It then uses dynamic programming to select the best object in the current timestep based on the complete history of all target object and distractor object tracklets. By explicitly modeling the motion and interaction of all potential objects and pooling similarity information from detections grouped into tracklets, Siam R-CNN is able to effectively perform long-term tracking, while being resistant to tracker drift, and being able to immediately re-detect objects after disappearance. Our TDPA requires only a small set of new re-detections in each timestep, updating its tracking history iteratively online. This allows Siam R-CNN to run at 4.7 frames per second (FPS) and its

[†] Work performed both while at the RWTH Aachen University and on a research visit at the University of Oxford.

speed-optimized variant to run at more than 15 FPS.

We present evaluation results on a large number of datasets. Siam R-CNN outperforms all previous methods on six short-term tracking benchmarks as well as on four long-term tracking benchmarks, where it achieves especially strong results, up to 10 percentage points higher than previous methods. By obtaining segmentation masks using an off-the-shelf box-to-segmentation network, Siam R-CNN also outperforms all previous Video Object Segmentation methods that only use the first-frame bounding box (without the mask) on four recent VOS benchmarks.

2. Related Work

Visual Object Tracking (VOT). VOT is the task of tracking an object through a video given the first-frame bounding box of the object. VOT is commonly evaluated on benchmarks such as OTB [69, 70], VOT [36, 34], and many more [49, 30, 81, 48, 33]. Recently a number of long-term tracking benchmarks have been proposed [45, 62, 18] which extend VOT to a more difficult and realistic setting, where objects must be tracked over many frames, with objects disappearing and reappearing.

Many classical methods use an online learned classifier to re-detect the object of interest over the full image [1, 21, 2, 56, 23, 57, 32]. In contrast, Siam R-CNN learns the expected appearance variations by offline training instead of learning a classifier online.

Like our Siam R-CNN, many recent methods approach VOT using Siamese architectures. Siamese region proposal networks (SiamRPN [38]) use a single-stage RPN [54] detector adapted to re-detect a template by cross-correlating the deep template features with the deep features of the current frame. Here, single-stage means directly classifying anchor boxes [42] which is in contrast to two-stage architectures [54] which first generate proposals, and then align their features and classify them in the second stage.

Recent tracking approaches improve upon SiamRPN, making it distractor aware (DaSiamRPN [82]), adding a cascade (C-RPN [19]), producing masks (SiamMask [66]), using deeper architectures (SiamRPN+ [79] and SiamRPN++ [37]) and maintaining a set of diverse templates (THOR [58]). These (and many more [7, 27, 46]) only search for the object within a small window of the previous prediction. DiMP [5] follows this paradigm while meta-learning a robust target and background appearance model.

Other recent developments in VOT include using domain specific layers with online learning [50], learning an adaptive spatial filter regularizer [14], exploiting category-specific semantic information [61], using continuous [17] or factorized [15] convolutions, and achieving accurate bounding box predictions using an overlap prediction network [16]. Huang *et al.* [31] propose a framework to convert any detector into a tracker. Like Siam R-CNN, they also apply two-stage architectures, but their method relies on meta-learning and it achieves a much lower accuracy.

Long-term tracking is mainly addressed by enlarging the search window of these Siamese trackers when the detection confidence is low [82, 37]. In contrast, we use a two-stage Siamese re-detector which searches over the whole image, producing stronger results across many benchmarks.

Video Object Segmentation (VOS). VOS is an extension of VOT where a set of template segmentation masks are given, and segmentation masks need to be produced in each frame. Many methods perform fine-tuning on the template masks [8, 47, 64, 39, 3, 44], which leads to strong results but is slow. Recently, several methods have used the first-frame masks without fine-tuning [11, 75, 12, 29, 71, 72, 63, 52], running faster but often not performing as well.

Very few methods [66, 76] tackle the harder problem of producing mask tracking results while only using the given template bounding box and not the mask. We adapt our method to perform VOS in this setting by using a second network to produce masks for our box tracking results.

3. Method

Inspired by the success of Siamese trackers [34, 70, 36], we use a Siamese architecture for our re-detector. Many recent trackers [82, 66, 37, 38, 5] adopt a single-stage detector architecture. For the task of single-image object detection, two-stage detector networks such as Faster R-CNN [54] have been shown to outperform single-stage detectors. Inspired by this, we design our tracker as a Siamese two-stage detection network. The second stage can directly compare a proposed Region of Interest (RoI) to a template region by concatenating their RoI aligned features. By aligning proposals and reference to the same size, Siam R-CNN achieves robustness against changes in object size and aspect ratio, which is hard to achieve when using the popular cross-correlation operation [38]. Fig. 2 shows an overview of Siam R-CNN including the Tracklet Dynamic Programming Algorithm (TDPA).

3.1. Siam R-CNN

Siam R-CNN is a Siamese re-detector based on a two-stage detection architecture. Specifically, we take a Faster R-CNN network that has been pre-trained on the COCO [41] dataset for detecting 80 object classes. This network consists of a backbone feature extractor followed by two detection stages; first a category-agnostic RPN, followed by a category-specific detection head. We fix the weights of the backbone and the RPN and replace the category-specific detection head with our re-detection head.

We create input features for the re-detection head for each region proposed by the RPN by performing RoI Align [25] to extract deep features from this proposed region. We

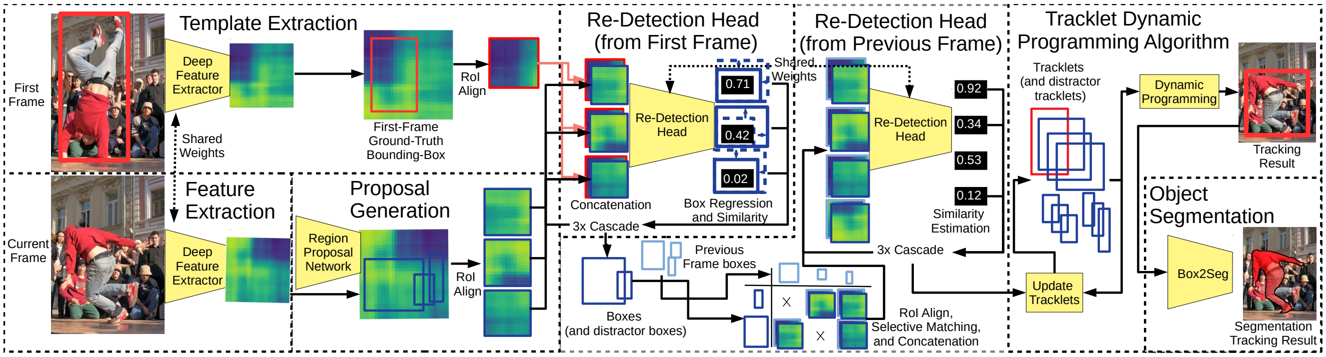

Figure 2: Overview of Siam R-CNN. A Siamese R-CNN provides re-detections of the object given in the first-frame bounding box, which are used by our Tracklet Dynamic Programming Algorithm along with re-detections from the previous frame. The results are bounding box level tracks which can be converted to segmentation masks by the Box2Seg network.

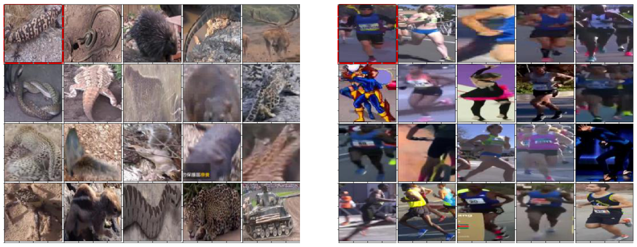

Figure 3: Hard negative mining examples retrieved from other videos for the reference objects shown in red.

also take the RoI Aligned deep features of the initialization bounding box in the first frame, and then concatenate these together and feed the combined features into a 1 × 1 convolution which reduces the number of features channels back down by half. These joined features are then fed into the re-detection head with two output classes; the proposed region is either the reference object or it is not. Our re-detection head uses a three-stage cascade [9] without shared weights. The structure of the re-detection head is the same as the structure of the detection head of Faster R-CNN, except for using only two classes and for the way the input features for the re-detection head are created by concatenation. The backbone and RPN are frozen and only the re-detection head (after concatenation) is trained for tracking, using pairs of frames from video datasets. Here, an object in one frame is used as reference and the network is trained to re-detect the same object in another frame.

3.2. Video Hard Example Mining

During conventional Faster R-CNN training, the negative examples for the second stage are sampled from the regions proposed by the RPN in the target image. However, in many images there are only few relevant negative examples. In order to maximize the discriminative power of the re-detection head, we need to train it on hard negative examples. Mining hard examples for detection has been explored in previous works (*e.g.* [20, 59]). However, rather than finding general hard examples for detection, we find hard examples for re-detection conditioned on the reference object by retrieving objects from other videos.

Embedding Network. A straightforward approach to selecting relevant videos from which to get hard negative examples for the current video, is taking videos in which an object has the same class as the current object [82]. However, object class labels are not always available, and some objects of the same class could be easy to distinguish, while some objects of different classes could also be potentially hard negatives. Hence, we propose to use an embedding network, inspired by person re-identification, which extracts an embedding vector for every ground truth bounding box which represents the appearance of that object. We use the network from PReMVOS [44], which is trained with batch-hard triplet loss [28] to separate classes on COCO before being trained on YouTube-VOS to disambiguate between individual object instances. *E.g.*, two distinct persons should be far away in the embedding space, while two crops of the same person in different frames should be close.

Index Structure. We next create an efficient indexing structure for approximate nearest neighbor queries (see supplemental material) and use it to find nearest neighbors of the tracked object in the embedding space. Fig. 3 shows examples of the retrieved hard negative examples. As can be seen, most of the negative examples are very relevant and hard.

Training Procedure. Evaluating the backbone on-the-fly on other videos to retrieve hard negative examples for the current video frame would be very costly. Instead, we precompute the RoI-aligned features for every ground truth box of the training data. For each training step, as usual, a random video and object in this video is selected and then a random reference and a random target frame. Afterwards, we use the indexing structure to retrieve for the reference box the 10,000 nearest neighbor bounding boxes from other videos and sample 100 of them as additional negative training examples. More details about video hard example mining can be found in the supplemental material.

Algorithm 1 Update tracklets for one time-step t

```
1:  Inputs $\text{ff\_gt\_feats}, \text{tracklets}, \text{image}_t, \text{dets}_{t-1}$
2:  $\text{backbone\_feats} \leftarrow \text{backbone}(\text{image}_t)$
3:  $\text{RoIs} \leftarrow \text{RPN}(\text{backbone\_feats}) \cup \text{dets}_{t-1}$
4:  $\text{dets}_t \leftarrow \text{redetection\_head}(\text{RoIs}, \text{ff\_gt\_feats})$
5:  ▷ scores are set to $-\infty$ if spatial distance is $> \gamma$
6:  $\text{scores} \leftarrow \text{score\_pairwise\_redetection}(\text{dets}_t, \text{dets}_{t-1}, \gamma)$
7:  for $d_t \in \text{dets}_t$ do
8:      $s_1 \leftarrow \max_{d_{t-1} \in \text{dets}_{t-1}} \text{scores}[d_t, d_{t-1}]$
9:      $\hat{d}_{t-1} \leftarrow \text{argmax}_{d_{t-1} \in \text{dets}_{t-1}} \text{scores}[d_t, d_{t-1}]$
10:     ▷ Max score of all other current detections
11:     $s_2 \leftarrow \max_{\tilde{d}_t \in \text{dets}_t \setminus \{d_t\}} \text{scores}[\tilde{d}_t, \hat{d}_{t-1}]$
12:     ▷ Max score of all other previous detections
13:     $s_3 \leftarrow \max_{d_{t-1} \in \text{dets}_{t-1} \setminus \{\hat{d}_{t-1}\}} \text{scores}[d_t, d_{t-1}]$
14:     if $s_1 > \alpha \wedge s_2 \leq s_1 - \beta \wedge s_3 \leq s_1 - \beta$ then
15:         $\text{tracklet}(\hat{d}_{t-1}).\text{append}(d_t)$  ▷ Extend tracklet
16:     else  ▷ Start new tracklet
17:         $\text{tracklets} \leftarrow \text{tracklets} \cup \{\{d_t\}\}$
18:     end if
19: end for
```

3.3. Tracklet Dynamic Programming Algorithm

Our Tracklet Dynamic Programming Algorithm (TDPA) implicitly tracks both the object of interest and potential similar-looking distractors using spatio-temporal cues. In this way, distractors can be consistently suppressed, which would not be possible using only visual similarity. To this end, TDPA maintains a set of tracklets, *i.e.*, short sequences of detections which almost certainly belong to the same object. It then uses a dynamic programming based scoring algorithm to select the most likely sequence of tracklets for the template object between the first and the current frame.

Each detection is part of exactly one tracklet and it is defined by a bounding box, a re-detection score, and its RoI-aligned features. A tracklet is defined by a set of detections, exactly one for each time step from its start to its end time.

Tracklet Building. We extract the RoI aligned features for the first-frame ground truth bounding box (ff_gt_feats) and initialize a tracklet consisting of just this box. For each new frame, we update the set of tracklets as follows (*c.f.* Algorithm 1): We extract backbone features of the current frame and evaluate the region proposal network (RPN) to get regions of interest (RoIs, lines 2–3). To compensate for potential RPN false negatives, the set of RoIs is extended by the bounding box outputs from the previous frame. We run the re-detection head (including bounding box regression) on these RoIs to produce a set of re-detections of the first-frame template (line 4). Afterwards, we re-run the classification part of the re-detection head (line 6) on the current detections dets_t, but this time with the detections dets_{t-1} from the previous frame as reference instead of the first-frame ground truth box, to calculate similarity scores (scores) between each pair of detections.

To measure the spatial distance of two detections, we represent their bounding boxes by their center coordinates x and y, and their width w and height h, of which x and w are normalized with the image width, and y and h are normalized with the image height, so that all values are between 0 and 1. The spatial distance between two bounding boxes (x_1, y_1, w_1, h_1) and (x_2, y_2, w_2, h_2) is then given by the L_∞ norm, *i.e.*, $\max(|x_1 - x_2|, |y_1 - y_2|, |w_1 - w_2|, |h_1 - h_2|)$. In order to save computation and to avoid false matches, we calculate the pairwise similarity scores only for pairs of detections where this spatial distance is less than γ and set the similarity score to $-\infty$ otherwise.

We extend the tracklets from the previous frame by the current frame detections (lines 7–19) when the similarity score to a new detection is high ($>\alpha$) and there is no ambiguity, *i.e.*, there is no other detection which has an almost as high similarity (less than β margin) with that tracklet, and there is no other tracklet which has an almost as high similarity (less than β margin) with that detection. Whenever there is any ambiguity, we start a new tracklet which initially consists of a single detection. The ambiguities will then be resolved in the tracklet scoring step.

Scoring. A track $A = (a_1, \ldots, a_N)$ is a sequence of N non-overlapping tracklets, *i.e.*, $\text{end}(a_i) < \text{start}(a_{i+1})\ \forall i \in \{1, \ldots, N-1\}$, where start and end denote the start and end times of a tracklet, respectively. The total score of a track consists of a unary score measuring the quality of the individual tracklets, and of a location score which penalizes spatial jumps between tracklets, *i.e.*

$$\text{score}(A) = \sum_{i=1}^{N} \text{unary}(a_i) + \sum_{i=1}^{N-1} w_{\text{loc}} \text{loc\_score}(a_i, a_{i+1}). \tag{1}$$

$$\begin{aligned} \text{unary}(a_i) = \sum_{t=\text{start}(a_i)}^{\text{end}(a_i)} & w_{\text{ff}} \text{ff\_score}(a_{i,t}) \\ & + (1 - w_{\text{ff}}) \text{ff\_tracklet\_score}(a_{i,t}), \end{aligned} \tag{2}$$

where ff_score denotes the re-detection confidence for the detection $a_{i,t}$ of tracklet a_i at time t from the re-detection head using the first-frame ground truth bounding box as reference. There is always one tracklet which contains the first-frame ground truth bounding box, which we denote as the first-frame tracklet a_{ff}. All detections in a tracklet have a very high chance of being a correct continuation of the initial detection of this tracklet, because in cases of ambiguities tracklets are terminated. Hence, the most recent detection of the first-frame tracklet is also the most recent observation that is almost certain to be the correct object. Thus, we use this detection as an additional reference for re-detection producing a score denoted by ff_tracklet_score which is linearly combined with the ff_score.

The location score between two tracklets a_i and a_j is given by the negative L_1 norm of the difference between

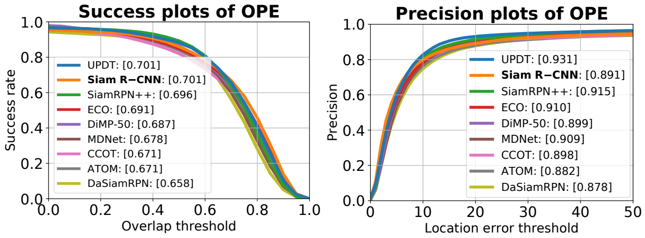

Figure 4: Results on OTB2015 [70].

the bounding box (x, y, w, h) of the last detection of a_i and the bounding box of the first detection of a_j, *i.e.*

$$\text{loc\_score}(a_i, a_j) = -|\text{end\_bbox}(a_i) - \text{start\_bbox}(a_j)|_1.$$

Online Dynamic Programming. We efficiently find the sequence of tracklets with the maximum total score (Eq. 1) by maintaining an array θ which for each tracklet a stores the total score $\theta[a]$ of the optimal sequence of tracklets which starts with the first-frame tracklet and ends with a.

Once a tracklet is not extended, it is terminated. Thus, for each new frame only the scores for tracklets which have been extended or newly created need to be newly computed. For a new time-step, first we set $\theta[a_{\mathit{ff}}] = 0$ for the first-frame tracklet a_{ff}, since all tracks have to start with that tracklet. Afterwards, for every tracklet a which has been updated or newly created, $\theta[a]$ is calculated as

$$\theta[a] = \text{unary}(a) + \max_{\tilde{a}:\text{end}(\tilde{a})<\text{start}(a)} \theta[\tilde{a}] + w_{\text{loc}}\text{loc\_score}(\tilde{a}, a).$$

To retain efficiency for very long sequences, we allow a maximum temporal gap between two tracklets of 1500 frames, which is long enough for most applications.

After updating θ for the current frame, we select the tracklet $\hat{a}$ with the highest dynamic programming score, *i.e.* $\hat{a} = \arg\max_a \theta[a]$. If the selected tracklet does not contain a detection in the current frame, then our algorithm has indicated that the object is not present. For benchmarks that require a prediction in every frame we use the most recent box from the selected tracklet, and assign it a score of 0.

3.4. Box2Seg

To produce segmentation masks for the VOS task, we use an off-the-shelf bounding-box-to-segmentation (Box2Seg) network from PReMVOS [44]. Box2Seg is a fully convolutional DeepLabV3+ [10] network with an Xception-65 [13] backbone. It has been trained on Mapillary [51] and COCO [41] to output the mask for a bounding box crop. Box2Seg is fast, running it after tracking only requires 0.025 seconds per object per frame. We combine overlapping masks such that masks with less pixels are on top.

3.5. Training Details

Siam R-CNN is built upon the Faster R-CNN [54] implementation from [67], with a ResNet-101-FPN backbone [26, 40], group normalization [68] and cascade [9]. It has been pre-trained from scratch [24] on COCO [41]. Except where specified otherwise, we train Siam R-CNN on the training sets of multiple tracking datasets simultaneously: ImageNet VID [55] (4000 videos), YouTube-VOS 2018 [72] (3471 videos), GOT-10k [30] (9335 videos) and LaSOT [18] (1120 videos). In total, we use 18k videos and 119k static images from COCO, which is a significant amount of data, but it is actually less than what previous methods used, *e.g.* SiamRPN++ uses 384k videos and 1867k static images. More details about the amount of training data are in the supplemental material.

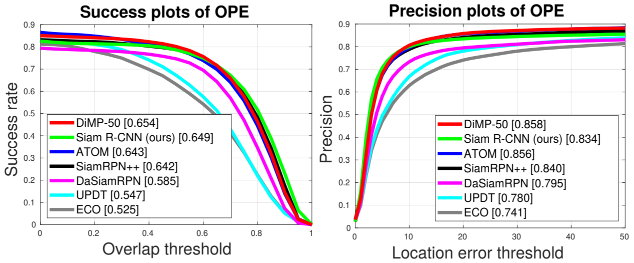

Figure 5: Results on UAV123 [48].

During training, we use motion blur [82], grayscale, gamma, flip, and scale augmentations.

4. Experiments

We evaluate Siam R-CNN for standard visual object tracking, for long-term tracking, and on VOS benchmarks. We tune a single set of hyper-parameters for our Tracklet Dynamic Programming Algorithm (*c.f.* Section 3.3) on the DAVIS 2017 training set, as this is a training set that we did not use to train our re-detector. We present results using these hyper-parameters on all benchmarks, rather than tuning the parameters separately for each one.

4.1. Short-Term Visual Object Tracking Evaluation

We evaluate short-term VOT on six benchmarks, and on five further benchmarks in the supplemental material.

OTB2015. We evaluate on OTB2015 [70] (100 videos, 590 frames average length), calculating the success and precision over varying overlap thresholds. Methods are ranked by the area under the success curve (AUC). Fig. 4 compares our results to eight state-of-the-art (SOTA) trackers [6, 37, 15, 5, 50, 17, 16, 82]. Siam R-CNN achieves 70.1% AUC, which equals the previous best result by UPDT [6].

UAV123. Fig. 5 shows our results on UAV123 [48] (123 videos, 915 frames average length) on the same metrics as OTB2015 compared to six SOTA approaches [5, 16, 37, 82, 6, 15]. We achieve an AUC of 64.9%, which is close to the previous best result of DiMP-50 [5] with 65.4%.

NfS. Tab. 1 shows our results on the NfS dataset [33] (30FPS, 100 videos, 479 frames average length) compared to five SOTA approaches. Siam R-CNN achieves a success score of 63.9%, which is 1.9 percentage points higher than

	Huang *et al.* [31]	UPDT [6]	ATOM [16]	Tripathi *et al.* [61]	DiMP-50 [5]	Siam R-CNN
Success	51.5	53.7	58.4	60.5	62.0	**63.9**

Table 1: Results on NfS [5].

	DaSiamRPN [82]	UPDT [6]	ATOM [16]	SiamRPN++ [37]	DiMP-50 [5]	Siam R-CNN
Precision	59.1	55.7	64.8	69.4	68.7	**80.0**
Norm. Prec.	73.3	70.2	77.1	80.0	80.1	**85.4**
Success	63.8	61.1	70.3	73.3	74.0	**81.2**

Table 2: Results on TrackingNet [49].

	LADCF [73]	ATOM [16]	SiamRPN++ [37]	THOR [5]	DiMP-50 [58]	Ours	Ours (short-t.)
EAO	0.389	0.401	0.414	0.416	**0.440**	0.140	0.408
Accuracy	0.503	0.590	0.600	0.582	0.597	**0.624**	0.609
Robustn.	0.159	0.204	0.234	0.234	**0.153**	1.039	0.220
AO	0.421	-	**0.498**	-	-	0.476	0.462

Table 3: Results on VOT2018 [34].

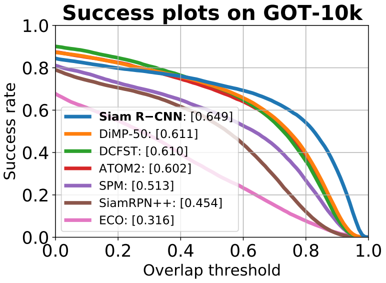

Figure 6: Results on GOT-10k [30].

the previous best result by DiMP-50 [5].

TrackingNet. Tab. 2 shows our results on the TrackingNet test set [49] (511 videos, 442 frames average length), compared to five SOTA approaches. Siam R-CNN achieves a success score of 81.2%, *i.e.*, 7.2 percentage points higher than the previous best result of DiMP-50 [5]. In terms of precision the gap is more than 10 percentage points.

GOT-10k. Fig. 6 shows our results on the GOT-10k [30] test set (180 videos, 127 frames average length) compared to six SOTA approaches [5, 80, 16, 65, 37, 15]. On this benchmark, methods are only allowed to use the GOT-10k training set as video data for training. Therefore we train a new model starting from COCO pre-training, and train only on GOT-10k. We achieve a success rate of 64.9% which is 3.8 percentage points higher than the previous best result from DiMP-50 [5]. This shows that Siam R-CNN's advantage over all previous methods is not just due to different training data, but from the tracking approach itself.

VOT2018. Tab. 3 shows our results on VOT2018 [34] (60 videos, 356 frames average length), where a reset-based evaluation is used. Once the object is lost, the tracker is restarted with the ground truth box five frames later and receives a penalty. The main evaluation criterion is the Expected Average Overlap (EAO) [35]. This extreme short-term tracking scenario is not what Siam R-CNN with the TDPA was designed for. It often triggers resets, which without reset-based evaluation Siam R-CNN could automatically recover from, resulting in an EAO of 0.140. For this setup, we created a simple short-term version of Siam R-CNN which averages the predictions of re-detecting the first-frame reference and re-detecting the previous prediction and combines them with a strong spatial prior. With 0.408 EAO this variant is competitive with many SOTA approaches. Notably, both versions of Siam R-CNN achieve the highest accuracy scores. The last row shows the average overlap (AO), when using the normal (non-reset) evaluation. When estimating rotated bounding boxes from segmentation masks produced by Box2Seg, Siam R-CNN's EAO increases to 0.423 and the accuracy greatly improves to 0.684. More details on rotated boxes and on the short-term tracking algorithm are in the supplemental material.

4.2. Long-Term Visual Object Tracking Evaluation

We evaluate Siam R-CNN's ability to perform long-term tracking on three benchmarks, LTB35 [45], LaSOT [18] and OxUvA [62]. In the supplemental material we also evaluate on UAV20L [48]. In long-term tracking, sequences are much longer, and objects may disappear and reappear again (LTB35 has on average 12.4 disappearances per video, each one on average 40.6 frames long). Siam R-CNN significantly outperforms all previous methods on all of these benchmarks, indicating the strength of our tracking by re-detection approach. By searching globally over the whole image rather than within a local window of a previous prediction, our method is more resistant to drift, and can easily re-detect a target after disappearance.

LTB35. Fig. 7 shows the results of our method on the LTB35 benchmark (also known as VOT18-LT) [45] (35 videos, 4200 frames average length) compared to eight SOTA approaches. Trackers are required to output a confidence of the target being present for the prediction in each frame. Precision (Pr) and Recall (Re) are evaluated for a range of confidence thresholds, and the F-score is calculated as $F = \frac{2PrRe}{Pr+Re}$. Trackers are ranked by the maximum F-score over all thresholds. We compare to the 6 best-performing methods in the 2018 VOT-LT challenge [34] and to SiamRPN++ [37] and SPLT [74]. Siam R-CNN outperforms all previous methods with an F-score of 66.8%, *i.e.*, 3.9 percentage points higher than the previous best result.

LaSOT. Fig. 8 shows results on the LaSOT test set [18] (280 videos, 2448 frames average length) compared to nine SOTA methods [5, 16, 37, 50, 60, 4, 78, 22, 15]. Siam R-CNN achieves an unprecedented result with a success rate of 64.8% and 72.2% normalized precision. This is 8 percentage points higher in success and 7.4 points higher in normalized precision than the previous best method.

OxUvA. Tab. 4 shows results on the OxUvA test set [62]

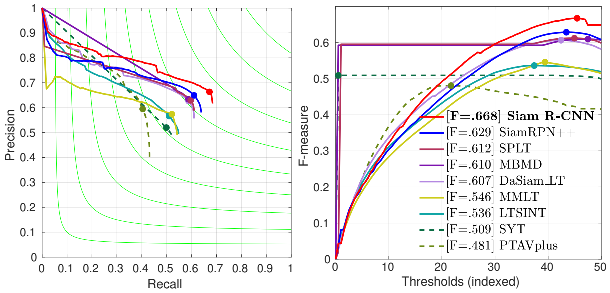

Figure 7: Results on LTB35 [45] (VOT18-LongTerm).

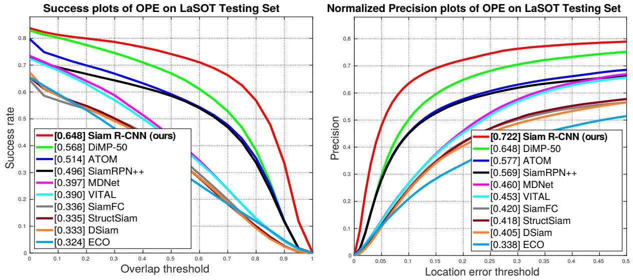

Figure 8: Results on LaSOT [18].

	DaSiam_LT [34]	TLD [32]	SiamFC+R [62]	MBMD [77]	SPLT [74]	Siam R-CNN
MaxGM	41.5	43.1	45.4	54.4	62.2	**72.3**
TPR	68.9	20.8	42.7	60.9	49.8	**70.1**
TNR	0	**89.5**	48.1	48.5	77.6	74.5

Table 4: Results on OxUvA [62].

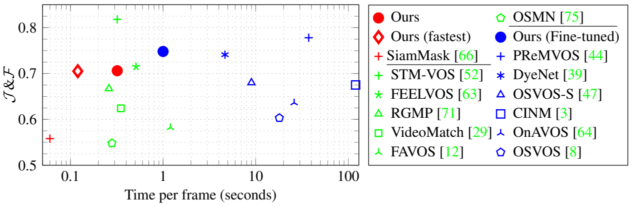

Figure 9: Quality versus timing on DAVIS 2017 (validation set). Only SiamMask [66] and our method (red) can work without the first-frame ground truth mask and require just the bounding box. Methods shown in blue fine-tune on the first-frame mask. Ours (fastest) denotes Siam R-CNN with ResNet-50, half resolution, and 100 RoIs, see Section 4.5.

(166 videos, 3293 frames average length) compared to five SOTA methods. Trackers must make a hard decision each frame whether the object is present. We do this by comparing the detector confidence to a threshold tuned on the dev set. Methods are ranked by the maximum geometric mean (MaxGM) of the true positive rate (TPR) and the true negative rate (TNR). Siam R-CNN achieves a MaxGM more than 10 percentage points higher than all previous methods.

4.3. Video Object Segmentation (VOS) Evaluation

We further evaluate the ability to track multiple objects and to segment them on VOS datasets using the $\mathcal{J}$ metric (mask intersection over union (IoU)), the $\mathcal{F}$ metric (mask boundary similarity), and the bounding box IoU $\mathcal{J}_{box}$.

Init	Method	FT	M	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}_{box}$	t(s)
bbox	**Siam R-CNN (ours)**	✗	✗	**70.6**	66.1	**75.0**	**78.3**	0.32
	Siam R-CNN (fastest)	✗	✗	70.5	**66.4**	74.6	76.9	0.12
	SiamMask [66]	✗	✗	55.8	54.3	58.5	64.3	**0.06**†
	SiamMask [66] (Box2Seg)	✗	✗	63.3	59.5	67.3	64.3	0.11
	SiamRPN++ [37] (Box2Seg)	✗	✗	61.6	56.8	66.3	64.0	0.11
	DiMP-50 [5] (Box2Seg)	✗	✗	63.7	60.1	67.3	65.6	0.10
mask	STM-VOS [52]	✗	✓	**81.8**	**79.2**	**84.3**	–	0.32†
	FEELVOS [63]	✗	✓	71.5	69.1	74.0	71.4	0.51
	RGMP [71]	✗	✓	66.7	64.8	68.6	66.5	**0.28**†
mask+ft	PReMVOS [44]	✓	✓	**77.8**	**73.9**	**81.7**	**81.4**	37.6
	Ours (Fine-tun. Box2Seg)	✓	✓	74.8	69.3	80.2	78.3	**1.0**
	DyeNet [39]	✓	✓	74.1	–	–	–	9.32†
	GT boxes (Box2Seg)	✗	✗	82.6	79.3	85.8	100.0	–
	GT boxes (Fine-t. Box2Seg)	✓	✓	86.2	81.8	90.5	100.0	–

Table 5: Results on the DAVIS 2017 validation set. FT: fine-tuning, M: using the first-frame masks, t(s): time per frame in seconds. †: timing extrapolated from DAVIS 2016. An extended table is in the supplemental material. Siam R-CNN (fastest) denotes Siam R-CNN with ResNet-50 backbone, half input resolution, and 100 RoIs, see Section 4.5.

Init	Method	FT	M	$\mathcal{O}$	$\mathcal{J}_{seen}$	$\mathcal{J}_{unseen}$	t(s)
bbox	**Siam R-CNN (ours)**	✗	✗	**68.3**	**69.9**	**61.4**	0.32
	Siam R-CNN (fastest)	✗	✗	66.2	69.2	57.7	0.12
	SiamMask [66]	✗	✗	52.8	60.2	45.1	**0.06**
mask	STM-VOS [52]	✗	✓	**79.4**	**79.7**	**72.8**	0.30†
	RGMP [71]	✗	✓	53.8	59.5	45.2	**0.26**†
mask+ft	**Ours (Fi.-tu. Box2Seg)**	✓	✓	**73.2**	**73.5**	**66.2**	**0.65**
	PReMVOS [44, 43]	✓	✓	66.9	71.4	56.5	6
	OnAVOS [64]	✓	✓	55.2	60.1	46.6	24.5
	OSVOS [8]	✓	✓	58.8	59.8	54.2	17†

Table 6: Results on the YouTube-VOS 2018 [72] validation set. The notation is explained in the caption of Tab. 5.

DAVIS 2017. Tab. 5 and Fig. 9 show results on the DAVIS 2017 validation set (30 videos, 2.03 objects and 67.4 frames average length per video). Methods are ranked by the mean of $\mathcal{J}$ and $\mathcal{F}$. Siam R-CNN significantly outperforms the previous best method that only uses the first-frame bounding boxes, SiamMask [66], by 14.8 percentage points. To evaluate how much of this improvement comes from Box2Seg and how much from our tracking, we applied Box2Seg to the output of SiamMask. This does improve the results while still being 7.3 percentage points worse than our method. We also run SiamRPN++ [37] and DiMP-50 [5] with Box2Seg for comparison. As a reference for the achievable performance for our tracker, we ran Box2Seg on the ground truth boxes which resulted in a score of 82.6%.

Even without using the first-frame mask, Siam R-CNN outperforms many methods that use the mask such as RGMP [71] and VideoMatch [29], and even some methods like OSVOS-S [47] that perform slow first-frame fine-tuning. Our method is also more practical, as it is far more tedious to create a perfect first-frame segmentation mask by hand than a bounding box initialization. If the first-frame mask is available, then we are able to fine-tune Box2Seg on

Dataset Eval measure	Speed FPS	OTB2015 AUC	LaSOT AUC	LTB35 F
Siam R-CNN	4.7	70.1	**64.8**	66.8
No hard ex. mining	4.7	68.4	63.2	66.5
Argmax	4.9	63.8	62.9	65.5
Short-term	4.6	67.2	55.7	57.2
$\frac{1}{2}$res. + 100 RoIs	13.6	69.1	63.2	66.0
ResNet-50	5.1	68.0	62.3	64.4
ResNet-50 + $\frac{1}{2}$res. + 100 RoIs	**15.2**	67.7	61.1	63.7

Table 7: Ablation and timing analysis of Siam R-CNN.

this, improving results by 4.2 percentage points at the cost of speed. We evaluate on the DAVIS 2017 test-dev benchmark and on DAVIS 2016 [53] in the supplemental material.

YouTube-VOS. Tab. 6 shows results on YouTube-VOS 2018 [72] (474 videos, 1.89 objects and 26.6 frames average length per video). Methods are ranked by the mean $\mathcal{O}$ of the $\mathcal{J}$ and $\mathcal{F}$ metrics over classes in the training set (*seen*) and unseen classes. Siam R-CNN again outperforms all methods which do not use the first-frame mask (by 15.5 percentage points), and also outperforms PReMVOS [44, 43] and all other previous methods except for STM-VOS [52].

4.4. Ablation and Timing Analysis

Tab. 7 shows a number of ablations of Siam R-CNN on three datasets together with their speed (using a V100 GPU). Siam R-CNN runs at 4.7 frames per second (FPS) using a ResNet-101 backbone, 1000 RPN proposals per frame, and TDPA. The row "No hard ex. mining" shows the results without hard example mining (*c.f*. Sec. 3.2). Hard example mining improves results on all datasets, by up to 1.7 percentage points. We compare TDPA to using just the highest scoring re-detection in each frame ("Argmax") and the short-term algorithm we used for the reset-based VOT18 evaluation ("Short-term"). TDPA outperforms both of these on all datasets. A per-attribute analysis of the influence of TDPA can be found in the supplemental material. For the long-term datasets, Argmax significantly outperforms both the short-term variant and even all previous methods.

4.5. Making Siam R-CNN Even Faster

Tab. 7 also shows the result of three changes aimed at increasing the speed of Siam R-CNN (smaller backbone, smaller input resolution, and less RoI proposals). More details and analyses are in the supplemental material.

When evaluating with a ResNet-50 backbone, Siam R-CNN performs slightly faster and still achieves SOTA results (62.3 on LaSOT, compared to 56.8 for DiMP-50 with the same backbone). This shows that the results are not only due to a larger backbone. When using half input resolution and only 100 RoIs from the RPN, the speed increases from 4.7 FPS to 13.6 FPS, or even 15.2 FPS in the case of ResNet-50. These setups still show very strong results, especially for long-term tracking. Note that even the fastest variant is not real-time and our work focuses on accuracy achieving much better results, especially for long-term tracking, while still running at a reasonable speed.

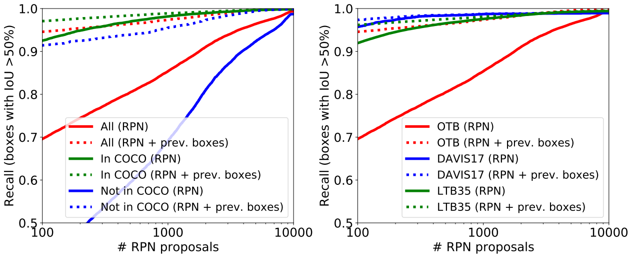

Figure 10: RPN recall with varying number of proposals. Dotted lines have up to 100 re-detections from the previous frame added. Left: comparison on COCO/non-COCO classes of OTB2015. Right: comparison over three datasets.

4.6. Generic Object Tracking Analysis

Siam R-CNN should be able to track any generic object. However, its backbone and RPN have been trained only on 80 object classes in COCO and have then been frozen. In Fig. 10, we investigate the recall of our RPN on the 44% of OTB2015 sequences that contain objects not in COCO, compared to the rest. With the default of 1000 proposals, the RPN achieves only 69.1% recall for unknown objects, compared to 98.2% for known ones. One solution is to increase the number of proposals used. When using $10,000$ proposals the RPN achieves 98.7% recall for unknown objects but causes Siam R-CNN to run much slower (around 1 FPS). Our solution is to instead include the previous-frame re-detections (up to 100) as additional proposals. This increases the recall to 95.5% for unknown objects when using 1000 RPN proposals. This shows why Siam R-CNN is able to outperform all previous methods on OTB2015, even though almost half of the objects are not from COCO classes. We also run a recall analysis on the DAVIS 2017 and LTB35 datasets where most objects belong to COCO classes and we achieve excellent recall (see Fig. 10 right).

5. Conclusion

We introduce Siam R-CNN as a Siamese two-stage full-image re-detection architecture with a Tracklet Dynamic Programming Algorithm. Siam R-CNN outperforms all previous methods on ten tracking benchmarks, with especially strong results for long-term tracking. We hope that our work will inspire future work on using two-stage architectures and full-image re-detection for tracking.

Acknowledgements: For partial funding of this project, PV, JL and BL would like to acknowledge the ERC Consolidator Grant DeeViSe (ERC-2017-COG-773161) and a Google Faculty Research Award. PHST would like to acknowledge CCAV project Streetwise and EPSRC/MURI grant EP/N019474/1. The authors would like to thank Sourabh Swain, Yuxin Wu, Goutam Bhat, and Bo Li for helpful discussions.

References

[1] S. Avidan. Support vector tracking. *PAMI*, 2004. 1, 2

[2] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *PAMI*, 2011. 1, 2

[3] L. Bao, B. Wu, and W. Liu. CNN in MRF: video object segmentation via inference in a cnn-based higher-order spatiotemporal MRF. In *CVPR*, 2018. 2, 7

[4] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr. Fully-convolutional siamese networks for object tracking. In *ECCVW*, 2016. 1, 6

[5] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte. Learning discriminative model prediction for tracking. In *ICCV*, 2019. 2, 5, 6, 7

[6] G. Bhat, J. Johnander, M. Danelljan, F. S. Khan, and M. Felsberg. Unveiling the power of deep tracking. In *ECCV*, 2018. 5, 6

[7] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 2

[8] S. Caelles, K.-K. Maninis, J. Pont-Tuset, L. Leal-Taixé, D. Cremers, and L. Van Gool. One-shot video object segmentation. In *CVPR*, 2017. 2, 7

[9] Z. Cai and N. Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *CVPR*, 2018. 3, 5

[10] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 5

[11] Y. Chen, J. Pont-Tuset, A. Montes, and L. Van Gool. Blazingly fast video object segmentation with pixel-wise metric learning. In *CVPR*, 2018. 2

[12] J. Cheng, Y-H. Tsai, W-C. Hung, S. Wang, and M-H. Yang. Fast and accurate online video object segmentation via tracking parts. In *CVPR*, 2018. 2, 7

[13] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, 2017. 5

[14] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li. Visual tracking via adaptive spatially-regularized correlation filters. In *CVPR*, 2019. 2

[15] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. ECO: Efficient convolution operators for tracking. In *CVPR*, 2017. 2, 5, 6

[16] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. ATOM: accurate tracking by overlap maximization. In *CVPR*, 2019. 2, 5, 6

[17] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 2, 5

[18] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling. LaSOT: A high-quality benchmark for large-scale single object tracking. In *CVPR*, 2019. 2, 5, 6, 7

[19] H. Fan and H. Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, 2019. 2

[20] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 2010. 3

[21] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC*, 2006. 1, 2

[22] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang. Learning dynamic siamese network for visual object tracking. In *ICCV*, 2017. 6

[23] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. S. Torr. Struck: Structured output tracking with kernels. *PAMI*, 2015. 1, 2

[24] K. He, R. Girshick, and P. Dollár. Rethinking imagenet pre-training. In *ICCV*, 2019. 5

[25] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017. 2

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[27] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *PAMI*, 2015. 2

[28] A. Hermans, L. Beyer, and B. Leibe. In Defense of the Triplet Loss for Person Re-Identification. *arXiv preprint arXiv:1703.07737*, 2017. 3

[29] Y.-T. Hu, J.-B. Huang, and A. G. Schwing. Videomatch: Matching based video object segmentation. In *ECCV*, 2018. 2, 7

[30] L. Huang, X. Zhao, and K. Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018. 2, 5, 6

[31] L. Huang, X. Zhao, and K. Huang. Bridging the gap between detection and tracking: A unified approach. In *ICCV*, 2019. 2, 6

[32] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *PAMI*, 2012. 1, 2, 7

[33] H. Kiani Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 2017. 2, 5

[34] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pfugfelder, L. C. Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, G. Fernandez, and et al. The sixth visual object tracking VOT2018 challenge results. In *ECCVW*, 2018. 2, 6, 7

[35] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Čehovin, G. Fernández, T. Vojir, G. Häger, G. Nebehay, R. Pflugfelder, A. Gupta, and et al. The visual object tracking VOT2015 challenge results. In *ICCVW*, 2015. 6

[36] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin. A novel performance evaluation methodology for single-target trackers. *PAMI*, 2016. 2

[37] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. SiamRPN++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 1, 2, 5, 6, 7

[38] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018. 1, 2

[39] X. Li and C. Change Loy. Video object segmentation with joint re-identification and attention-aware mask propagation. In *ECCV*, 2018. 2, 7

[40] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 5

[41] T-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 2, 5

[42] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 2

[43] J. Luiten, P. Voigtlaender, and B. Leibe. PReMVOS: Proposal-generation, refinement and merging for the YouTube-VOS challenge on video object segmentation 2018. *ECCVW*, 2018. 7, 8

[44] J. Luiten, P. Voigtlaender, and B. Leibe. PReMVOS: Proposal-generation, refinement and merging for video object segmentation. In *ACCV*, 2018. 2, 3, 5, 7, 8

[45] A. Lukežič, L. Č. Zajc, T. Vojíř, J. Matas, and M. Kristan. Now you see me: evaluating performance in long-term visual tracking. *arXiv preprint arXiv:1804.07056*, 2018. 2, 6, 7

[46] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 2

[47] K.-K. Maninis, S. Caelles, Y. Chen, J. Pont-Tuset, L. Leal Taixé, and L. Van Gool. Video object segmentation without temporal information. *PAMI*, 2018. 2, 7

[48] M. Mueller, N. Smith, and B. Ghanem. A benchmark and simulator for uav tracking. In *ECCV*, 2016. 2, 5, 6

[49] M. Müller, A. Bibi, S. Giancola, S. Al-Subaihi, and B. Ghanem. TrackingNet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018. 2, 6

[50] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 2, 5, 6

[51] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder. The Mapillary Vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017. 5

[52] S. Wug Oh, J.-Y. Lee, N. Xu, and S. Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 2, 7, 8

[53] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016. 8

[54] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2, 5

[55] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 2015. 5

[56] A. Saffari, M. Godec, T. Pock, C. Leistner, and H. Bischof. Online multi-class lpboost. In *CVPR*, 2010. 1, 2

[57] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *ICCVW*, 2009. 1, 2

[58] A. Sauer, E. Aljalbout, and S. Haddadin. Tracking holistic object representations. In *BMVC*, 2019. 2, 6

[59] A. Shrivastava, A. Gupta, and R. B. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016. 3

[60] Y. Song, C. Ma, X. Wu, L. Gong, L. Bao, W. Zuo, C. Shen, R. Lau, and M.-H. Yang. VITAL: Visual tracking via adversarial learning. In *CVPR*, 2018. 6

[61] A. S. Tripathi, M. Danelljan, L. Van Gool, and R. Timofte. Tracking the known and the unknown by leveraging semantic information. In *BMVC*, 2019. 2, 6

[62] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. W. M. Smeulders, P. H. S. Torr, and E. Gavves. Long-term tracking in the wild: A benchmark. In *ECCV*, 2018. 2, 6, 7

[63] P. Voigtlaender, Y. Chai, F. Schroff, H. Adam, B. Leibe, and L.-C. Chen. FEELVOS: Fast end-to-end embedding learning for video object segmentation. In *CVPR*, 2019. 2, 7

[64] P. Voigtlaender and B. Leibe. Online adaptation of convolutional neural networks for video object segmentation. In *BMVC*, 2017. 2, 7

[65] G. Wang, C. Luo, Z. Xiong, and W. Zeng. Spm-tracker: Series-parallel matching for real-time visual object tracking. In *CVPR*, 2019. 6

[66] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 2, 7

[67] Y. Wu et al. Tensorpack. `https://github.com/tensorpack/`, 2016. 5

[68] Y. Wu and K. He. Group normalization. In *ECCV*, 2018. 5

[69] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 2

[70] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *PAMI*, 2015. 2, 5

[71] S. Wug Oh, J.-Y. Lee, K. Sunkavalli, and S. Joo Kim. Fast video object segmentation by reference-guided mask propagation. In *CVPR*, 2018. 2, 7

[72] N. Xu, L. Yang, Y. Fan, J. Yang, D. Yue, Y. Liang, B. Price, S. Cohen, and T. Huang. YouTube-VOS: Sequence-to-sequence video object segmentation. In *ECCV*, 2018. 2, 5, 7, 8

[73] T. Xu, Z.-H. Feng, X.-J. Wu, and J. Kittler. Learning adaptive discriminative correlation filters via temporal consistency preserving spatial feature selection for robust visual object tracking. *Trans. Image Proc.*, 2019. 6

[74] B. Yan, H. Zhao, D. Wang, H. Lu, and X. Yang. 'Skimming-Perusal' Tracking: A framework for real-time and robust long-term tracking. In *ICCV*, 2019. 6, 7

[75] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos. Efficient video object segmentation via network modulation. In *CVPR*, 2018. 2, 7

[76] D. Yeo, J. Son, B. Han, and J. Hee Han. Superpixel-based tracking-by-segmentation using markov chains. In *CVPR*, 2017. 2

[77] Y. Zhang, D. Wang, L. Wang, J. Qi, and H. Lu. Learning regression and verification networks for long-term visual tracking. *arXiv preprint arXiv:1809.04320*, 2018. 7

[78] Y. Zhang, L. Wang, J. Qi, D. Wang, M. Feng, and H. Lu. Structured siamese network for real-time visual tracking. In *ECCV*, 2018. 6

[79] Z. Zhang, H. Peng, and Q. Wang. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, 2019. 2

[80] L. Zheng, M. Tang, J. Wang, and H. Lu. Learning features with differentiable closed-form solver for tracking. *arXiv preprint arXiv:1906.10414*, 2019. 6

[81] P. Zhu, L. Wen, X. Bian, L. Haibin, and Q. Hu. Vision meets drones: A challenge. *arXiv preprint arXiv:1804.07437*, 2018. 2

[82] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, 2018. 2, 3, 5, 6