Learning Target Candidate Association to Keep Track of What Not to Track Supplementary

Christoph Mayer Martin Danelljan Danda Pani Paudel Luc Van Gool
Computer Vision Lab, D-ITET, ETH Zürich, Switzerland

Appendices

In this supplementary material, we first provide details about training in Sec. A and about inference in Sec. B. We then report a more detailed analysis of out method in Sec. C and provide more detailed results of the experiments shown in the main paper in Sec. D.

Furthermore, we want to draw the attention of the reader to the compiled video available on the project page at https://github.com/visionml/pytracking. The video provides more visual insights about our tracker and compares visually with the baseline tracker SuperDiMP. It shows tracking of distractors and indicates the same object ids in consecutive frames with agreeing color.

A. Training

First, we describe the training data generation and sample selection to train the network more effectively. Then, we provide additional details about the training procedure such as training in batches, augmentations and synthetic sample generation. Finally, we briefly summarize the employed network architecture.

A.1. Data-Mining

We use the LaSOT [15] training set to train our target candidate association network. In particular, we split the 1120 training sequences randomly into a *train-train* (1000 sequences) and a *train-val* (120 sequences) set. We run the base tracker on all sequences and store the target classifier score map and the search area on disk for each frame. During training, we use the score map and the search area to extract the target candidates and its features to provide the data to train the target candidate association network.

Name	Number of candidates	Is a candidate selected as target? $\max(s_i) \geq \eta$	Does the candidate with max score correspond to the target?	Does any candidate correspond to the target?	Num Frames	Ratio
D	1	✓	✓	✓	1.8M	67.9%
H	> 1	✓	✓	✓	498k	18.4%
G	> 1	x	✓	–	8k	0.3%
J	> 1	✓	x	x	76k	2.8%
K	> 1	✓	x	✓	42k	1.5%
other	–	–	–	–	243k	9.1%

Table 1. Categories and specifications for each frame in the training dataset used for data-mining.

We observed that many sequences or sub-sequences contain mostly one target candidate with a high target classifier score. Thus, in this cases target candidate association is trivial and learning on these cases will be less effective. Conversely, tracking datasets contain sub-sequences that are very challenging (large motion or appearance changes or many distractors) such that trackers often fail. While these sub-sequences lead to a more effective training they are relatively rare such that we decide to actively search the training dataset.

First, we assign each frame to one of six different categories. We classify each frame based on four observations about the number of candidates, their target classifier score, if one of the target candidates is selected as target and if this selection is correct, see Tab. 1. A candidate corresponds to the annotated target object if the spatial distance between the candidate location and center coordinate of the target object is smaller than a threshold.

Assigning each frame to the proposed categories, we observe, that the dominant category is D (70%) that corresponds to frames with a single target candidate matching the annotated target object. However, we favour more challenging settings for training. In order to learn distractor associations using self supervision, we require frames with multiple detected target candidates. Category H (18.4%) corresponds to such frames where in addition the candidate with the highest target classifier score matches the annotated target object. Hence, the base tracker selects the correct candidate as target. Furthermore, category G corresponds to frames where the base tracker was no longer able to track the target because the target classifier score of the corresponding candidate fell bellow a threshold. We favour these frames during training in order to learn continue tracking the target even if the score is low.

Both categories J and K correspond to tracking failures of the base tracker. Whereas in K the correct target is de-

tected but not selected, it is not detected in frames of category J. Thus, we aim to learn from tracking failures in order to train the target candidate association network such that it learns to compensate for tracking failures of the base tracker and corrects them. In particular, frames of category K are important for training because the two candidates with highest target classifier score no longer match such that the network is forced to include other cues for matching. We use frames of category J because frames where the object is visible but not detected contain typically many distractors such that these frames are suitable to learn distractor associations using self-supervised learning.

To summarize, we select only frames with category H, K, J for self-supervised training and sample them with a ratio of $2:1:1$ instead of $10:2:1$ (ratio in the dataset). We ignore frames from category D during self-supervised training because we require frames with multiple target candidates. Furthermore, we select sub-sequences of two consecutive frames for partially supervised training. We choose challenging sub-sequences that either contain many distractors in each frame (HH, 350k) or sub-sequences where the base tracker fails and switches to track a distractor (HK, 1001) or where the base tracker is no longer able to identify the target with high confidence (HG, 1380). Again we change the sampling ratio from approximately $350:1:1$ to $10:1:1$ during training. We change the sampling ration in order to use failure cases more often during training than they occur in the training set.

A.2. Training Data Preparation

During training we use two different levels of augmentation. First, we augment all features of target candidate to enable self-supervised training with automatically produced ground truth correspondences. In addition, we use augmentation to improve generalization and overfitting of the network.

When creating artificial features we randomly scale each target classifier score, randomly jitter the candidate location within the search area and apply common image transformations to the original image before extracting the appearance based features for the artificial candidates. In particular, we randomly jitter the brightness, blur the image and jitter the search area before cropping the image to the search area.

To reduce overfitting and improve the generalization, we randomly scale the target candidate scores for synthetic and real sub-sequences. Furthermore, we remove candidates from the sets $\mathcal{V}'$ and $\mathcal{V}$ randomly in order to simulate newly appearing or disappearing objects. Furthermore, to enable training in batches we require the same number of target candidates in each frame. Thus, we keep the five candidates with the highest target classifier score or add artificial peaks at random locations with a small score such that five candidates per frame are present. When computing the losses, we ignore these artificial candidates.

A.3. Architecture Details

We use the SuperDiMP tracker [8] as our base tracker. SuperDiMP employs the DiMP [1] target classifier and the probabilistic bounding-box regression of PrDiMP [12], together with improved training settings. It uses a ResNet-50 [22] pretrained network as backbone feature extractor. We freeze all parameters of SuperDiMP while training the target candidate association network. To produce the visual features for each target candidate, we use the third layer ResNet-50 features. In particular, we obtain a $29\times29\times1024$ feature map and feed it into a 2×2 convolutional layer which produces the $30 \times 30 \times 256$ feature map $\mathbf{f}$. Note, that the spatial resolution of the target classifier score and feature map agree such that extracting the appearance based features $\mathbf{f}_i$ for each target candidate v_i at location $\mathbf{c}_i$ is simplified.

Furthermore, we use a four layer Multi-Layer Perceptron (MLP) to project the target classifier score and location for each candidate in the same dimensional space as $\mathbf{f}_i$. We use the following MLP structure: $3 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 256$ with batch normalization. Before feeding the candidate locations into the MLP we normalize it according to the image size.

We follow Sarlin *et al*. [34] when designing the candidate embedding network. In particular, we use self and cross attention layers in an alternating fashion and employ two layers of each type. In addition, we append a 1×1 convolutional layer to the last cross attention layer. Again, we follow Sarlin *et al*. [34] for optimal matching and reuse their implementation of the Sinkhorn algorithm and run it for 10 iterations.

B. Inference

In this section we provide the detailed algorithm that describes the object association module (Sec. 3.7 in the paper). Furthermore, we explain the idea of search area rescaling at occlusion and how it is implemented. We conclude with additional inference details.

B.1. Object Association Module

Here, we provide a detailed algorithm describing the object association module presented in the main paper, see Alg. 1. It contains target candidate to object association and the redetection logic to retrieve the target object after it was lost.

First, we will briefly explain the used notation. Each object can be modeled similar to a *class* in programming. Thus, each object o contains attributes that can be accesses using the "." notation. In particular $(o_j).s$ returns the score

Algorithm 1 Object Association Algorithm.

Require: Set of target candidates $\mathcal{V}$
Require: Set of objects of previous frame $\mathcal{O}'$
Require: Target object $\hat{o}'$

$\mathcal{O} = \{\}, N = |\mathcal{V}|$
for $i = 1, \ldots, N$ **do**
  if $\text{matchOf}(v_i) \neq \text{dustbin}$ && $p(v_i) \geq \omega$ **then**
    $v'_j \leftarrow \text{matchOf}(v_i)$
    $(o'_j).s \leftarrow \text{concat}((o'_j).s, [s_i])$
    $o_i \leftarrow o'_j$
  else
    $o_i \leftarrow \text{new Object}(\text{getNewId}(), [s_i])$
  $\mathcal{O} \leftarrow \mathcal{O} \cup \{o_i\}$
if $\hat{o}' \neq \text{none}$ **and** $\hat{o}'.\text{id} \in \{o.\text{id} \mid o \in \mathcal{O}\}$ **then**
  $\hat{o} = \text{getObjectById}(\mathcal{O}, \hat{o}'.\text{id})$
  for $i = 1, \ldots, N$ **do**
    if $\max(\hat{o}.s) < (o_i).s[-1]$ **then**
      $\hat{o} = o_i$
else
  $i = \text{argmax}_i\{(o_i).s[-1]) \mid o_i \in \mathcal{O}\}$
  if $(o_i).s[-1] \geq \eta$ **then**
    $\hat{o} = o_i$
  else
    $\hat{o} = \text{none}$
return $\hat{o}, \mathcal{O}$

attribute s of object o_j. In total the object class contains two attribute: the list of scores s and the object-id id. Both setting and getting the attribute values is possible.

The algorithm requires the following inputs: the set of target candidates $\mathcal{V}$, the set of detected objects $\mathcal{O}'$ and the object selected as target $\hat{o}$ in the previous frame. First, we check if a target candidate matches with any previously detected object and verify that the assignment probability is higher than a threshold $\omega = 0.75$. If such a match exists, we associate the candidate to the object and append its target classifier score to the scores and add the object to the set of currently visible object $\mathcal{O}$. If a target candidate matches none of the previously detected objects, we create a new object and add it to $\mathcal{O}$. Hence, previously detected objects that miss a matching candidate are not included in $\mathcal{O}$. Once, all target candidates are associated to an already existing or newly created object. We check if the object previously selected as target is still visible in the current scene and forms the new target $\hat{o}$. After the object was lost it is possible that the object selected as target is in fact a distractor. Thus, we select an other object as target if this other object achieves a higher target classifier score in the current frame than any score the currently selected object achieved in the past. Furthermore, if the object previously selected as target object is no longer visible, we try to redetect it by checking if the object with highest target classifier score in the current frame achieves a score higher than a threshold $\eta = 0.25$. If the score is high enough, we select this object as the target.

B.2. Search Area Rescaling at Occlusion

The target object often gets occluded or moves out-of-view in many tracking sequences. Shortly before the target is lost the tracker typically detects only a small part of the target object and estimates a smaller bounding box than in the frames before. The used base tracker SuperDiMP employs a search area that depends on the currently estimated bounding box size. Thus, a partially visible target object causes a small bounding box and search area. The problem of a small search area is that it complicates redetecting the target object, *e.g.*, the target reappears at a slightly different location than it disappeared and if the object then reappears outside of the search area redetection is prohibited. Smaller search areas occur more frequently when using the target candidate association network because it allows to track the object longer until we declare it as lost.

Hence, we use a procedure to increase the search area if it decreased before the target object was lost. First, we store all search are resolutions during tracking in an list a as long as the object is detected. If the object was lost k frames ago, we compute the new search area by averaging the last k entries of a larger than the search area at occlusion. We average at most over 30 previous search areas to compute the new one. If the target object was not redetected within these 30 frames with keep the search area fixed until redetection.

B.3. Inference Details

In contrast to training, we use all extracted target candidates to compute the candidate associations between consecutive frames. In order to save computations, we extract the candidates and features only for the current frame and cache the results such that they can be reused when computing the associations in the next frames.

B.3.1 KeepTrack Settings

We use the same settings as for SuperDiMP but increase the search area scale from 6 to 8 leading to a larger search are (from 352×352 to 480×480) and to a larger target score map (from 22×22 to 30×30). In addition, we employ the aforementioned search area rescaling at occlusion and skip running the target candidate association network if only one target candidates with high target classifier score is detected in the current and previous frame, in order to save computations.

Memory Memory Confidence	Larger Search Area	Search Area Rescaling	Candidate Association Network	NFS	UAV123	LaSOT
–	–	–	–	64.4	68.2	63.5
✓	–	–	–	64.7	68.0	65.0
✓	✓	–	–	65.3	68.4	65.5
✓	–	✓	–	64.7	68.4	65.8
✓	✓	✓	–	65.2	69.1	65.8
✓	✓	✓	✓	**66.4**	**69.7**	**67.1**

Table 2. Impact of each component in terms of AUC (%) on three datasets. The first row corresponds to our SuperDiMP baseline.

Num GNN Layers	Num Sinkhorn iterations	NFS	UAV123	LaSOT	FPS
–	–	65.2	69.1	65.8	–
0	50	65.9	69.2	66.6	–
2	10	66.4	69.7	67.1	18.3
9	50	66.4	69.8	67.2	12.7

Table 3. Impact of each component of the Target Candidate Association Network in terms of AUC (%) on three datasets.

B.3.2 KeepTrackFast Settings

We use the same settings as for SuperDiMP. In particular, we keep the search area scale and target score map resolution the same to achieve a faster run-time. In addition, we reduce the number of bounding box refinement steps from 10 to 3 which reduces the bounding box regression time significantly. Moreover, we double the target candidate extraction threshold τ to 0.1. This step ensures that we neglegt local maxima with low target classifier scores and thus leads to less frames with multiple detected candidates. Hence, *KeepTrackFast* runs the target candidate association network less often than *KeepTrack*.

C. More Detailed Analysis

In addition to the ablation study presented in the main paper (Sec. 4.1) we provide more settings in order to assess the contribution of each component better. In particular, we split the term *search area adaptation* into *larger search area* and *search area rescaling*. Where larger search area refers to a search area scale of 8 instead of 6 and a search area resolution of 480 instead of 352 in the image domain. Tab. 2 shows all the results on NFS [18], UAV123 [32] and LaSOT [15]. We run each experiment five times and report the average. We conclude that both search area adaptation techniques improve the tracking quality but we achieve the best results on all three datasets when employing both at the same time. Furthermore, we evaluate the target candidate association network with different numbers of Sinkhorn iterations and with different number of GNN layers of the embedding network or dropping it at all, see Tab. 3. We conclude, that using the target candidate association network even without any GNN layers outperforms the baseline on all three datasets. In addition, using either two or nine GNN layers improves the performance even further on all datasets. We achieve the best results when using nine GNN layers and 50 Sinkhorn iterations. However, using a large candidate embedding network and a high number of Sinkhorn iterations reduces the run-time of the tracker to 12.7 FPS. Hence, using only two GNN layers and 10 Sinkhorn iterations results in a negligible decrease of 0.1 on UAV123 and LaSOT but accelerates the run-time by 44%.

D. Experiments

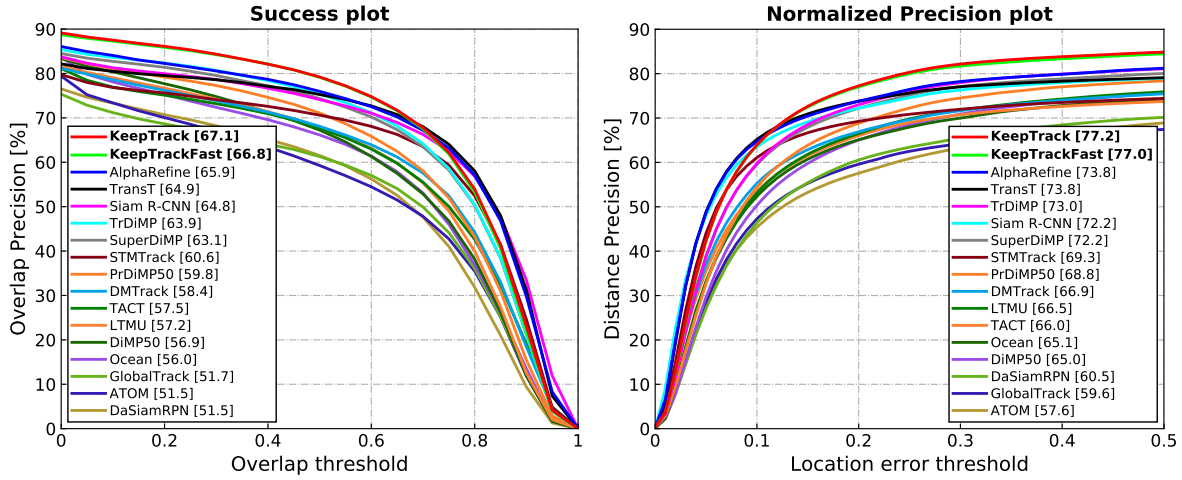
We provide more details to complement the state-of-the-art comparison performed in the paper. And provide results for the VOT2018LT [24] challenge.

D.1. LaSOT and LaSOTExtSub

In addition to the success plot, we provide the normalized precision plot on the LaSOT [15] test set (280 videos) and LaSOTExtSub [15] test set (150 videos). The normalized precision score NPr_D is computed as the percentage of frames where the normalized distance (relative to the target size) between the predicted and ground-truth target center location is less than a threshold D. NPr_D is plotted over a range of thresholds $D \in [0, 0.5]$. The trackers are ranked using the AUC, which is shown in the legend. Figs. 1b and 2b show the normalized precision plots. We compare with state-of-the-art trackers and report their success (AUC) in Tab. 4 and where available we show the raw results in Fig. 1. In particular, we use the raw results provided by the authors except for DaSiamRPN [49], GlobaTrack [23], SiamRPN++ [27] and SiamMask [38] such results were not provided such that we use the raw results produced by Fan *et al*. [15]. Thus, the exact results for these methods might be different in the plot and the table, because we show in the table the reported result the corresponding paper. Similarly, we obtain all results on LaSOTExtSub directly from Fan *et al*. [14] except the result of SuperDiMP that we produced.

D.2. UAV123, OTB-100 and NFS

We provide the success plot over the 123 videos of the UAV123 dataset [32] in Fig. 3a, the 100 videos of the OTB-100 dataset [39] in Fig. 3b and the 100 videos of the NFS dataset [18] in Fig. 3c. We compare with state-of-the-art trackers SuperDiMP [8], PrDiMP50 [12], UPDT [3], SiamRPN++ [27], ECO [10], DiMP [1], CCOT [11], MDNet [33], ATOM [9], and DaSiamRPN [49]. Our method provides a significant gain over the baseline SuperDiMP on UAV123 and NFS and performs among the top methods on OTB-100. Tab. 5 shows additional results on UAV123, OTB-100 and NFS in terms of success (AUC).



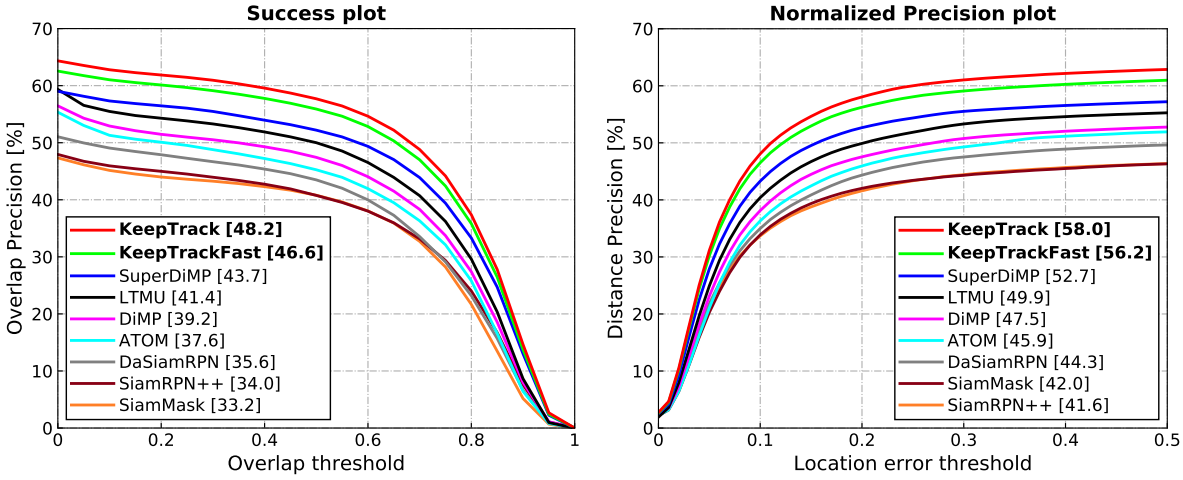
(a) Success (b) Normalized Precision

Figure 1. Success and normalized precision plots on LaSOT [15]. Our approach outperforms all other methods by a large margin in AUC, reported in the legend.

	Keep Track	**Keep Track Fast**	Alpha Refine [41]	TransT [4]	Siam R-CNN [35]	Tr DiMP [37]	Super Dimp [8]	STM Track [17]	Pr DiMP [12]	DM Track [47]	TLPG [28]	TACT [6]	LTMU [7]	DiMP [1]	Ocean [46]	Siam AttN [44]
LaSOT	**67.1**	**66.8**	65.3	64.9	64.8	63.9	63.1	60.6	59.8	58.4	58.1	57.5	57.2	56.9	56.0	56.0

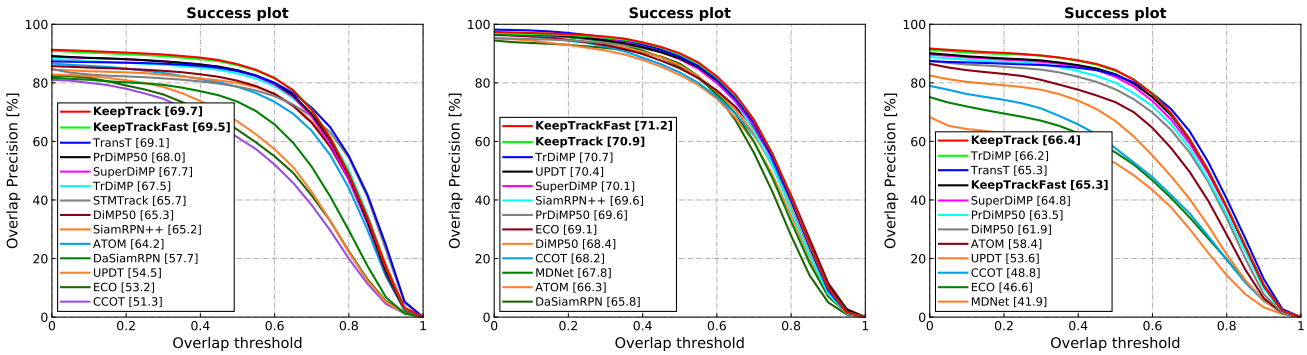
	CRACT [16]	Siam FC++ [40]	Siam GAT [20]	PG NET [30]	FCOS MAML [36]	Global Track [23]	ATOM [9]	DaSiam RPN [49]†	Siam BAN [5]	Siam CAR [21]	CLNet [13]	Siam RPN++ [27]†	Retina MAML [36]	Siam Mask [38]†	ROAM++ [43]	SPLT [42]
LaSOT	54.9	54.4	53.9	53.1	52.3	52.1	51.5	51.5	51.4	50.7	49.9	49.6	48.0	46.7	44.7	42.6

Table 4. Comparison with state-of-the-art on the LaSOT [15] test set in terms of overall AUC score. The average value over 5 runs is reported for our approach. The symbol † marks results that were produced by Fan *et al*. [15] otherwise they are obtained directly from the official paper.



(a) Success (b) Normalized Precision

Figure 2. Success and normalized precision plots on LaSOTExtSub [14]. Our approach outperforms all other methods by a large margin in AUC, reported in the legend.



(a) UAV123 (b) OTB-100 (c) NFS

Figure 3. Success plots on the UAV123 [32], OTB-100 [39] and NFS [18] datasets in terms of overall AUC score, reported in the legend.

	Keep Track	**Keep Track Fast**	Tr DiMP [37]	TransT [4]	Super DiMP [8]	Pr DiMP [12]	Siam R-CNN [35]	STM Track [17]	DiMP [1]	KYS [2]	Siam RPN++ [27]	ATOM [9]	UPDT [3]	Retina MAML [36]	FCOS MAML [36]	Ocean [46]	STN [31]
UAV123	**69.7**	**69.5**	67.5	69.1	67.7	68.0	64.9	64.7	65.3	–	61.3	64.2	54.5	–	–	–	64.9
OTB-100	70.9	71.2	71.1	69.4	70.1	69.6	70.1	**71.9**	68.4	69.5	69.6	66.9	70.2	71.2	70.4	68.4	69.3
NFS	**66.4**	65.3	**66.2**	65.7	64.8	63.5	63.9	–	62.0	63.5	–	58.4	53.7	–	–	–	–

	Auto Track [29]	Siam BAN [5]	Siam CAR [21]	ECO [10]	DCFST [48]	PG-NET [30]	CRACT [16]	GCT [19]	Siam GAT [20]	CLNet [13]	TLPG [28]	Siam AttN [44]	Siam FC++ [40]	MDNet [33]	CCTO [11]	DaSiam RPN [49]	ECOhc [10]
UAV123	67.1	63.1	61.4	53.2	–	–	66.4	50.8	64.6	63.3	–	65.0	–	–	51.3	57.7	50.6
OTB-100	–	69.6	–	69.1	70.9	69.1	**72.6**	64.8	71.0	–	69.8	71.2	68.3	67.8	68.2	65.8	64.3
NFS	–	59.4	–	46.6	64.1	–	62.5	–	–	54.3	–	–	–	41.9	48.8	–	–

Table 5. Comparison with state-of-the-art on the OTB-100 [39], NFS [18] and UAV123 [32] datasets in terms of overall AUC score. The average value over 5 runs is reported for our approach.

D.3. VOT2018LT [25]

Next, we evaluate our tracker on the 2018 edition of the VOT [26] long-term tracking challenge. We compare with the top methods in the challenge [25], as well as more recent methods. The dataset contains 35 videos with 4200 frames per sequence on average. Trackers are required to predict a confidence score that the target is present in addition to the bounding box for each frame. Trackers are ranked by the F-score, evaluated for a range of confidence thresholds. As shown in Tab. 6, our tracker achieves the best results in all three metrics and outperforms the base tracker SuperDiMP by almost 10% in F-score.

E. Speed Analysis

Our method adds an overhead of 19.3ms compared to the baseline tracker. Whereas target candidate extraction is required for every frame, running the target candidate association network is only required if more than one candidate is detected. Candidate extraction is relatively fast and takes only 2.7 ms while performing candidate association requires 16.6 ms. Where computing the candidate embedding takes 10.0 ms, running the Sinkhorn algorithm for 10 iterations takes 3.5 ms and object association 3.1 ms. Thus, we achieve an average run-time of 18.3 FPS for *KeepTrack* and 29.6 FPS for *KeepTrackFast* when using SuperDiMP as base tracker. We report this timings on a singe Nvidia GTX 2080Ti GPU.

	Keep Track	**Keep Track Fast**	LTMU [7]	Siam R-CNN [35]	PGNet [30]	Siam RPN++ [27]	Super DiMP [8]	SPLT [42]	MBMD [45]	DaSiamLT [49, 25]
Precision	**73.8**	70.1	**71.0**	–	67.9	64.9	64.3	63.3	63.4	62.7
Recall	**70.4**	67.6	**67.2**	–	61.0	60.9	61.0	60.0	58.8	58.8
F-Score	**72.0**	68.8	**69.0**	66.8	64.2	62.9	62.2	61.6	61.0	60.7

Table 6. Results on the VOT2018LT dataset [25] in terms of F-Score, Precision and Recall.

E.1. Number of Candidates and Time Complexity

In practice, we found the number of target candidates to vary from 0 up to 15 in exceptional cases. For frames with less than 2 candidates, we naturally do not need to apply our association module. Further, we observed no measurable increase in run-time from 2 to 15 candidates, due to the effective parallelization. Therefore, we do not explicitly limit the number of detected candidates.

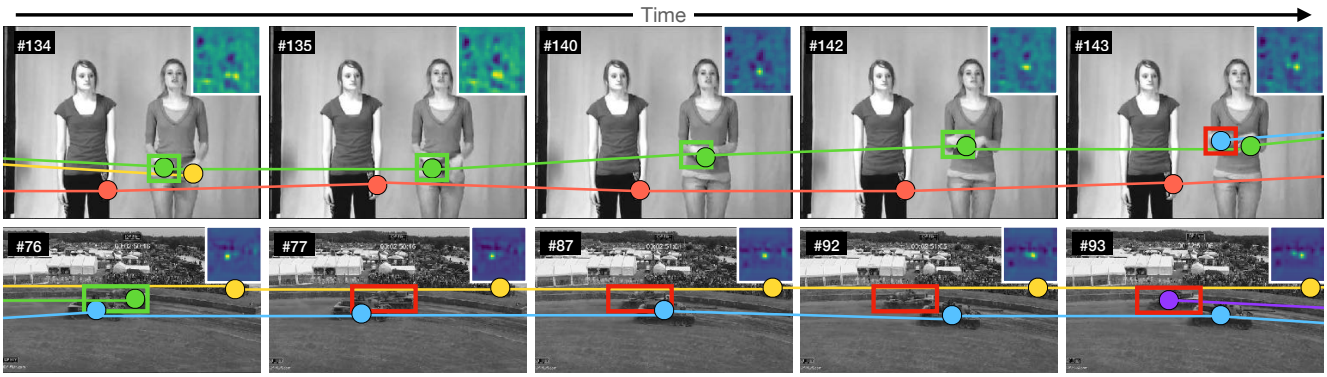


Figure 4. Failure Cases: a very challenging case is when a distractor crosses the target's location, since positional information is then of limited use. The box represents the ground truth bounding box of the target object, where green indicates the the selected target candidates corresponds to the sought target and red indicates that the tracker selected a candidate corresponding to a distractor object.

F. Failure Cases

While *KeepTrack* is particularly powerful when distractor objects appear in the scene, it also fails to track the target object in complex scenes, such as the examples shown in Fig. 4.

The top row shows such a challenging case, where the target object is the right hand of the person on the right (indicated by [■] or [■]). *KeepTrack* manages to individually track all hands (●,●,●) visible in the search area until frame number 134. In the next frame, both hands of the person are close and our tracker only detects one candidate for both hands (●). Thus, the tracker assigns the target id to the remaining candidate. The tracker detects two candidates (●,●) as soon as both hands move apart in frame 143. However, now it is unclear which hand is the sought target. If two objects approach each other it is unclear whether they cross each other or not. In this scenario positional information is of limited use. Hence, deeper understanding of the scene and the target object seems necessary to mitigate such failure cases.

The bottom row in Fig. 4 shows a similar failure case where again a distractor object (●) crosses the target's (●) location. This time, the tracker fails to extract the candidate corresponding to the target from frame number 77 on wards. The tracker detects that the target candidate previously assumed to represent the target has vanished but the remaining distractor object (●) achieves such a high target score that the tracker reconsiders its previous target selection and continues tracking the distractor object (●) instead. Thus, the tracker continues tracking the distractor object even if the a target candidate for the sought target (●) appears (frame number 93).

G. Attributes

Tabs. 7 and 8 show the results of various trackers including *KeepTrack* and *KeepTrackFast* based on different sequence attributes. We observe that both trackers are superior to other trackers on UAV123 for most attributes. In particular, we outperform the runner-up by a large margin in terms of AUC on the sequences corresponding to the following attributes: *Aspect Ratio Change* $(+2.5/2.6\%)$, Full Occlusion $(+1.8/1.9\%)$, Partial Occlusion $(+2.5/2.3\%)$, Background Clutter $(+1.5/1.3\%)$, Illumination Variation $(+1.7/1.5\%)$, Similar Object $(+1.8/1.1\%)$. Especially, the superior performance on sequences with the attributes *Full Occlusion*, *Partial Occlusion*, *Background Clutter* and *Similar Object* clearly demonstrates that *KeepTrack* mitigates the harmful effect of distractors and allows to track the target object longer and more frequently than other trackers. Fig. 3a shows a similar picture: *KeepTrack* is the most robust tracker but others achieve a higher bounding box regression accuracy. In addition, Tab. 7 reveals that *KeepTrack* achieves the highest (red) or second-highest (blue) AUC on the sequences corresponding to each attribute except for the attribute *Out-of-View*.

The attribute-based analysis on LaSOT allows similar observations. In particular, *KeepTrack* and *KeepTrackFast* outperform all other trackers by a large margin in AUC on the sequences corresponding oth the following attributes: *Partial Occlusion* $(+1.8/1.5\%)$, *Background Clutter* $(+2.3/1.2$, *Viewpoint Change* $(+1.6/2.3\%)$, *Full Occlusion* $(+2.7/1.8\%)$, *Fast Motion* $(4.1/3.5\%)$, *Out-of-View* $(+1.9/1.2\%)$, *Low Resolution* $(+3.4/3.4\%)$. Moreover, the superior performance is even clearer when comparing to the base tracker SuperDiMP, *e.g.*, and improvement of $+6.0/5.2\%$ for *Full Occlusion* or $+7.0/6.4\%$ for *Fast Motion*. *KeepTrack* achieves the highest AUC score for every attributed except two where *KeepTrackFast* achieves slightly higher scores. Again, the best performance on sequences with attributes such as *Background Clutter* or *Full Occlusion* clearly demonstrates the effectiveness of our proposed target and distractor association strategy.

	Scale Variation	Aspect Ratio Change	Low Resolution	Fast Motion	Full Occlusion	Partial Occlusion	Out-of-View	Background Clutter	Illumination Variation	Viewpoint Change	Camera Motion	Similar Object	Total
ATOM	63.0	61.9	49.5	62.7	46.2	58.1	61.4	46.2	63.1	65.1	66.4	63.1	64.2
DiMP50	63.8	62.8	50.9	62.7	47.5	59.7	61.8	48.9	63.9	65.2	66.9	62.9	65.3
STMTrack	63.9	64.2	46.4	62.2	48.9	58.0	**68.2**	46.2	61.9	70.2	67.5	58.0	65.7
TrDiMP	66.4	66.1	54.3	66.3	48.6	62.1	66.3	45.1	61.5	70.0	68.3	64.9	67.5
SuperDiMP	66.6	66.4	54.9	65.1	52.0	63.5	63.7	51.4	63.2	67.8	69.8	65.5	67.7
PrDiMP50	66.8	66.3	55.2	65.3	53.6	63.5	63.9	53.9	62.4	69.4	70.4	66.1	68.0
TransT	68.0	66.3	55.6	**67.4**	48.4	63.2	**69.1**	44.1	62.6	**71.8**	70.5	65.3	69.1
KeepTrackFast	**68.4**	**68.8**	**57.3**	67.2	**55.4**	**66.0**	65.9	**55.4**	**65.6**	70.3	**71.2**	**67.9**	**69.5**
KeepTrack	**68.7**	**68.9**	**57.0**	**68.0**	**55.5**	**65.8**	66.8	**55.2**	**65.4**	**70.4**	**71.8**	**67.2**	**69.7**

Table 7. UAV123 attribute-based analysis in terms of AUC score. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute.

	Illumination Variation	Partial Occlusion	Deformation	Motion Blur	Camera Motion	Rotation	Background Clutter	Viewpoint Change	Scale Variation	Full Occlusion	Fast Motion	Out-of-View	Low Resolution	Aspect Ration Change	Total
LTMU	56.5	54.0	57.2	55.8	61.6	55.1	49.9	56.7	57.1	49.9	44.0	52.7	51.4	55.1	57.2
PrDiMP50	63.7	56.9	60.8	57.9	64.2	58.1	54.3	59.2	59.4	51.3	48.4	55.3	53.5	58.6	59.8
STMTrack	65.2	57.1	64.0	55.3	63.3	60.1	54.1	58.2	60.6	47.8	42.4	51.9	50.3	58.8	60.6
SuperDiMP	67.8	59.7	63.4	62.0	68.0	61.4	57.3	63.4	62.9	54.1	50.7	59.0	56.4	61.6	63.1
TrDiMP	67.5	61.1	64.4	62.4	68.1	62.4	58.9	62.8	63.4	56.4	53.0	60.7	58.1	62.3	63.9
Siam R-CNN	64.6	62.2	65.2	63.1	68.2	64.1	54.2	65.3	64.5	55.3	51.5	62.2	57.1	63.4	64.8
TransT	65.2	62.0	**67.0**	63.0	67.2	64.3	57.9	61.7	64.6	55.3	51.0	58.2	56.4	63.2	64.9
AlphaRefine	69.4	62.3	66.3	**65.2**	70.0	63.9	58.8	63.1	65.4	57.4	53.6	61.1	58.6	64.1	65.3
KeepTrackFast	**70.1**	**63.8**	66.2	65.0	**70.7**	**65.1**	**60.1**	**67.6**	**66.6**	**59.2**	**57.1**	**63.4**	**62.0**	**65.6**	**66.8**
KeepTrack	**69.7**	**64.1**	**67.0**	**66.7**	**71.0**	**65.3**	**61.2**	**66.9**	**66.8**	**60.1**	**57.7**	**64.1**	**62.0**	**65.9**	**67.1**

Table 8. LaSOT attribute-based analysis. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute.

References

[1] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2, 4, 5, 6

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene information for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 6

[3] Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Unveiling the power of deep tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 4, 6

[4] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5, 6

[5] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[6] Janghoon Choi, Junseok Kwon, and Kyoung Mu Lee. Visual tracking by tridentalign and context embedding. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020. 5

[7] Kenan Dai, Yunhua Zhang, Dong Wang, Jianhua Li, Huchuan Lu, and Xiaoyun Yang. High-performance long-term tracking with meta-updater. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[8] Martin Danelljan and Goutam Bhat. PyTracking: Visual tracking library based on PyTorch. https://github.com/visionml/pytracking, 2019. Accessed: 1/08/2020. 2, 4, 5, 6

[9] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ATOM: Accurate tracking by overlap maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4, 5, 6

[10] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ECO: efficient convolution operators for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2017. 4, 6

[11] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, October 2016. 4, 6

[12] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 4, 5, 6

[13] Xingping Dong, Jianbing Shen, Ling Shao, and Fatih Porikli. Clnet: A compact latent network for fast adjusting siamese trackers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 5, 6

[14] Heng Fan, Hexin Bai, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Mingzhen Huang, Juehuan Liu, Yong Xu, et al. Lasot: A high-quality large-scale single object tracking benchmark. *International Journal of Computer Vision (IJCV)*, 129(2):439–461, 2021. 4, 5

[15] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 4, 5

[16] Heng Fan and Haibin Ling. Cract: Cascaded regression-align-classification for robust visual tracking. *arXiv preprint arXiv:2011.12483*, 2020. 5, 6

[17] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. Stmtrack: Template-free visual tracking with space-time memory networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5, 6

[18] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 2017. 4, 6

[19] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. Graph convolutional tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 6

[20] Dongyan Guo, Yanyan Shao, Ying Cui, Zhenhua Wang, Liyan Zhang, and Chunhua Shen. Graph attention tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5, 6

[21] Dongyan Guo, Jun Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[23] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Globaltrack: A simple and strong baseline for long-term tracking. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, February 2020. 4, 5

[24] Matej Kristan, Aleš Leonardis, Jiří Matas, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kämäräinen, Martin Danelljan, Luka Čehovin Zajc, Alan Lukežič, Ondrej Drbohlav, Linbo He, Yushan Zhang, Song Yan, Jinyu Yang, Gustavo Fernández, and et al. The eighth visual object tracking vot2020 challenge results. In *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, August 2020. 4

[25] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Cehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, and et al. The sixth visual object tracking vot2018 challenge results. In *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)*, September 2018. 6

[26] Matej Kristan, Jiri Matas, Aleš Leonardis, Tomas Vojir, Roman Pflugfelder, Gustavo Fernandez, Georg Nebehay, Fatih Porikli, and Luka Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38(11):2137–2155, 2016. 6

[27] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4, 5, 6

[28] Siyuan Li, Zhi Zhang, Ziyu Liu, Anna Wang, Linglong Qiu, and Feng Du. Tlpg-tracker: Joint learning of target localization and proposal generation for visual tracking. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, July 2020. 5, 6

[29] Yiming Li, Changhong Fu, Fangqiang Ding, Ziyuan Huang, and Geng Lu. Autotrack: Towards high-performance visual tracking for uav with automatic spatio-temporal regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 6

[30] Bingyan Liao, Chenye Wang, Yayun Wang, Yaonong Wang, and Jun Yin. Pg-net: Pixel to global matching network for visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 5, 6

[31] Yuan Liu, Ruoteng Li, Yu Cheng, Robby T. Tan, and Xiubao Sui. Object tracking using spatio-temporal networks for future prediction location. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 6

[32] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for uav tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, October 2016. 4, 6

[33] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 4, 6

[34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[35] Paul Voigtlaender, Jonathon Luiten, Philip H.S. Torr, and Bastian Leibe. Siam R-CNN: Visual tracking by redetection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[36] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[37] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5, 6

[38] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip H.S. Torr. Fast online object tracking and segmentation: A unifying approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4, 5

[39] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(9):1834–1848, 2015. 4, 6

[40] Yinda Xu, Zeyu Wang, Zuoxin Li, Ye Yuan, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, February 2020. 5, 6

[41] Bin Yan, Xinyu Zhang, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Alpha-refine: Boosting tracking performance by precise bounding box estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5

[42] Bin Yan, Haojie Zhao, Dong Wang, Huchuan Lu, and Xiaoyun Yang. 'skimming-perusal' tracking: A framework for real-time and robust long-term tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 5, 6

[43] Tianyu Yang, Pengfei Xu, Runbo Hu, Hua Chai, and Antoni B. Chan. Roam: Recurrently optimizing tracking model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5

[44] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R. Scott. Deformable siamese attention networks for visual object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 5, 6

[45] Yunhua Zhang, Lijun Wang, Dong Wang, Jinqing Qi, and Huchuan Lu. Learning regression and verification networks for long-term visual tracking. *International Journal of Computer Vision (IJCV)*, 129(9):2536–2547, 2021. 6

[46] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 5, 6

[47] Zikai Zhang, Bineng Zhong, Shengping Zhang, Zhenjun Tang, Xin Liu, and Zhaoxiang Zhang. Distractor-aware fast tracking via dynamic convolutions and mot philosophy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 5

[48] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Learning feature embeddings for discriminant model based tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 6

[49] Zheng Zhu, Qiang Wang, Li Bo, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 4, 5, 6