Global Tracking via Ensemble of Local Trackers

Zikun Zhou[1,*], Jianqiu Chen[1,*], Wenjie Pei[1,†], Kaige Mao[1], Hongpeng Wang[1,2], and Zhenyu He[1,†]
[1]Harbin Institute of Technology, Shenzhen [2]Peng Cheng Laboratory

Abstract

The crux of long-term tracking lies in the difficulty of tracking the target with discontinuous moving caused by out-of-view or occlusion. Existing long-term tracking methods follow two typical strategies. The first strategy employs a local tracker to perform smooth tracking and uses another re-detector to detect the target when the target is lost. While it can exploit the temporal context like historical appearances and locations of the target, a potential limitation of such strategy is that the local tracker tends to misidentify a nearby distractor as the target instead of activating the re-detector when the real target is out of view. The other long-term tracking strategy tracks the target in the entire image globally instead of local tracking based on the previous tracking results. Unfortunately, such global tracking strategy cannot leverage the temporal context effectively. In this work, we combine the advantages of both strategies: tracking the target in a global view while exploiting the temporal context. Specifically, we perform global tracking via ensemble of local trackers spreading the full image. The smooth moving of the target can be handled steadily by one local tracker. When the local tracker accidentally loses the target due to suddenly discontinuous moving, another local tracker close to the target is then activated and can readily take over the tracking to locate the target. While the activated local tracker performs tracking locally by leveraging the temporal context, the ensemble of local trackers renders our model the global view for tracking. Extensive experiments on six datasets demonstrate that our method performs favorably against state-of-the-art algorithms.

1. Introduction

The long-term visual tracking task has attracted more attention from the visual tracking community in recent years. Compared with short-term tracking, the long-term tracking task is much closer to the real-world application, due to the following two differences. First, the average duration of the sequences in the long-term tracking benchmarks (such as LaSOT [13], TLP [29], and OxUvA [33]) is hundreds of seconds, which is much longer than that (tens of seconds) of the short-term tracking benchmarks (OTB2015 [37], TrackingNet [30], and GOT-10k [18], to name a few). Second, the long-term tracking task requires the algorithms to be able to cope with the discontinuous moving of the target caused by the disappearance and reappearance of the target.

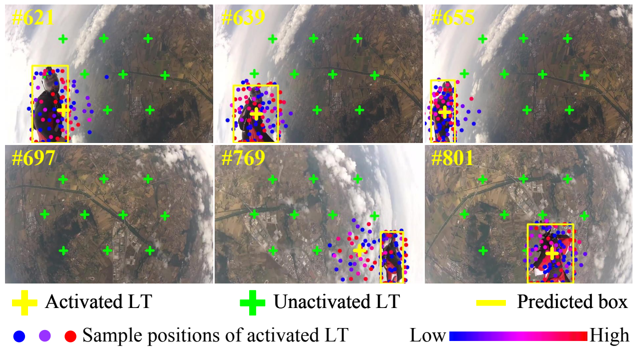

Figure 1. Illustration of the tracking process via ensemble of local trackers for tracking a wingsuit flyer, which disappears from the left-bottom corner of the view and then reappears from the right-bottom corner. The color of sample positions denotes the attention weight. At first, the Local Tracker (LT) at the left-bottom corner is activated and moves following the target to keep tracking it until the target disappears. While the activated local tracker loses the target, it moves back to where it starts moving, as shown in the 697th frame. When the target reappears, the local tracker at the right-bottom corner is activated and keeps tracking the target. The ensemble of local trackers collaborates to achieve global tracking.

Most existing trackers [1,2,8,22,46] search for the target within a local image region, named local trackers, and thus cannot handle the frequently discontinuous moving of the target in the long-term tracking task. To handle this issue, a typical strategy [5, 19, 26, 41] is to equip the local tracker with a global re-detector to detect the target after the local tracker fails. Such a strategy performs switching between local tracking and global detection according to the previous tracking result, named local-global switching strategy. A merit of this strategy is that the temporal context including historical appearances and locations of the target can be exploited for local tracking. However, whether to switch to global detection is totally decided by the local tracking result. It increases the risk that the algorithm misidentifies a

*Equal contribution. †Corresponding authors: Wenjie Pei and Zhenyu He (wenjiecoder@outlook.com and zhenyuhe@hit.edu.cn).

distractor as the target instead of activating the re-detector when the real target is out of the view of the local tracker.

In contrast to the local-global switching strategy, another type of long-term trackers [17, 44] adopt a global tracking strategy which performs global re-detection to locate the target in the entire image on every frame. For example, GlobalTrack [17] performs global tracking by one-shot detection, totally ignoring the temporal context such as historical appearances and locations of the target. As a result, GlobalTrack is vulnerable to the appearance variations of the target and the background distractors. To alleviate the issue, DMTrack [44] introduces a Re-ID embedding into the global re-detection framework to associate the detections across adjacent frames to utilize the previous re-detection results. However, this Re-ID embedding is learned using only pedestrian datasets [11, 28, 38, 45], limiting the generalization ability for tracking an arbitrary target object.

In this paper, we propose to perform global tracking via ensemble of local trackers, which combines the merits of both above strategies: tracking the target in a global view while exploiting the temporal context. Specifically, our algorithm deploys an ensemble of local trackers on different reference positions over the entire image, and every local tracker searches for the target within a local region around the reference position. With reasonable reference positions and search range, the search regions of all local trackers can cover the entire image, and then these local trackers collaborate to achieve global tracking.

The collaboration mechanisms in our design are: 1) When a local tracker successfully locates the target (called *activated*), it will move following the target to attempt to keep tracking it in the subsequent frames. Generally, the target moving smoothly can be tracked by a local tracker consecutively. 2) When the activated local tracker accidentally loses the target due to suddenly discontinuous moving, another local tracker close to the target can readily take over the tracking to locate the target, i.e., being activated, to avoid tracking failure. The local tracker that loses the target will move back to where it starts moving. During the period when a local tracker steadily tracks the target, our algorithm exploits the temporal context to improve the local tracking robustness, which further extends the duration that this local tracker successfully tracks the target.

Specifically, we design a deformable attention-based local tracker to search for the target within a dynamic local region. Thus we can simulate the routine local tracking mechanism in the global view by moving the dynamic local search region following the target. Based on the local tracker, we propose a temporal context transferring scheme to exploit the historical appearances and locations of the target for local tracking. Figure 1 illustrates the tracking process via ensemble of local trackers. To conclude, we make the following contributions:

- We propose a global tracking algorithm via ensemble of local trackers, which can track the target in a global view while exploiting the temporal context.
- We design a deformable attention-based local tracker to simulate the local tracking mechanism in the global view and a temporal context transferring scheme based on the local tracker to exploit the temporal context.
- We achieve favorable performance against state-of-the-art methods on six diverse datasets, demonstrating the effectiveness of our algorithm.

2. Related Work

Local-global Switching Strategy Trackers. Many algorithms [19, 25, 26] address the long-term tracking task through the local-global switching strategy. They equip the local tracker with a global re-detector to re-detect the target after local tracking fails, i.e., performing switching between local tracking and global re-detection based on the previous tracking result. TLD [19] is an early method using the strategy. It uses the optical flow for local tracking and an ensemble of weak classifiers for global re-detection. Recently, several methods [5, 41, 43] introduce advanced deep local trackers and global re-detectors into this framework. Besides, some methods [27, 48] that choose to enlarge the search region instead of global re-detection when local tracking fails can be seen as variants of this strategy.

A key issue for this strategy is how to decide whether to switch to global re-detection (or switch to a larger search region). Several methods [25–27, 48] directly make the switching decision based on the response map predicted by the local tracker, while other approaches [5, 41, 43] adopt an additional learnable verifier to manage the local-global switching. However, whether to switch from local tracking to global re-detection is still totally decided by the local tracking prediction. That is, the information out of the local search region is ignored for making the switch decision. It increases the risk that the algorithm misidentifies a distractor as the target instead of activating the global re-detector when the real target is out of the local search region.

Global Tracking Strategy Trackers. Several methods [16, 17, 35, 44] adopt a global tracking strategy by global re-detection on every frame for long-term tracking. Among these methods, GlobalTrack [17] performs tracking by global one-shot detection without considering the temporal context, which is sensitive to the target appearance variations. Siam R-CNN [35] designs a sophisticated global re-detector and associates the detections with a hand-crafted score for dynamic programming, but runs slowly due to the heavy computing burdens. Inspired by the MOT algorithms, DMTrack [44] introduces a Re-ID branch into the global re-detection framework to associate detections across frames. However, the Re-ID branch is trained using only human

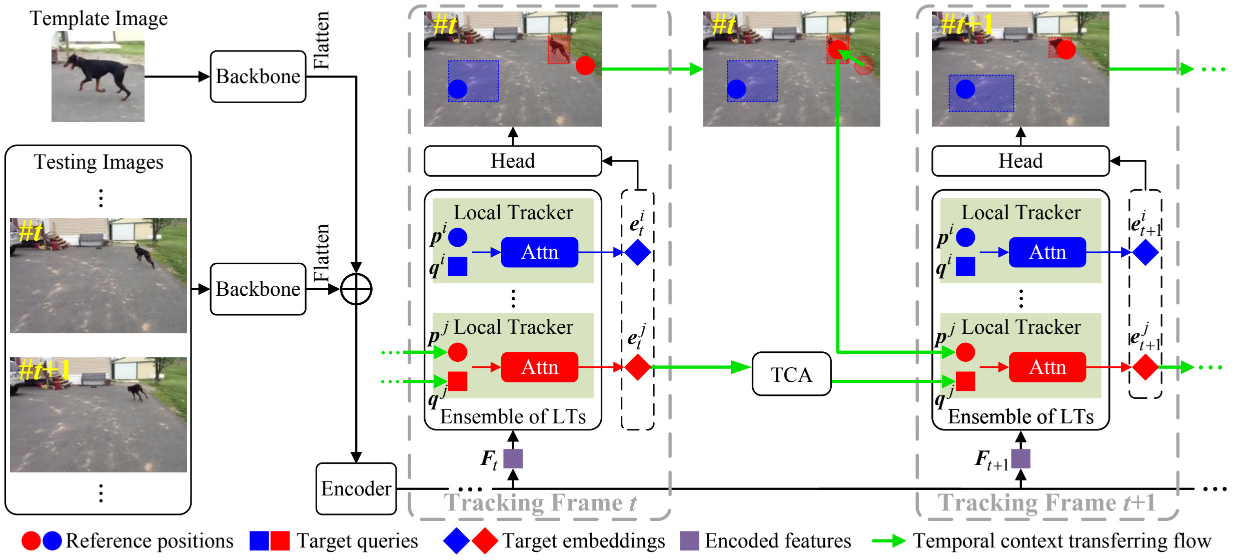

Figure 2. **The proposed global tracking framework via ensemble of local trackers.** It first extracts the backbone features for the template image and the testing images, and then adopts an encoder to enhance the target information in the features of the testing frames, generating the encoded feature for every testing image. An ensemble of local trackers (LTs) implemented with a decoder is constructed on the encoded feature to perform ensemble local tracking. $\oplus$ denotes the concatenation operation.

datasets [11, 28, 38, 45], inevitably damaging the generalization ability. Unlike these global trackers that ignore the temporal context or exploit the temporal context after re-detection by association, our method performs global tracking via ensemble of local trackers, directly exploiting the temporal context for predicting the candidate.

Transformer Tracking. Recently, several transformer-based trackers [4, 36, 39] have been proposed. Our idea is also implemented in an encoder-decoder structure like these works, but different from these methods that are designed as local trackers, our method aims to deploy an ensemble of local trackers in the encoder-decoder structure to achieve global tracking. Besides, some methods [4,34,42] adopt the encoder-decoder framework for MOT. Whilst our method and these MOT methods both use multiple queries, the major difference is: every query in MOT is in charge of detecting and tracking different objects, while in our method all queries work collaboratively for tracking the only target.

3. Method

In this section, we present our global tracking algorithm via ensemble of local trackers, which performs global tracking on every frame while effectively exploiting the temporal context. To track the target in a global view, our method deploys an ensemble of local trackers spreading over the entire image. Every local tracker searches for the target within different local regions. With reasonable distribution, the local search regions can together cover the entire image. Thus, these local trackers can perform global tracking via collaboration. Besides, the temporal context can be exploited when a local tracker continuously tracks the target.

3.1. Global Tracking Framework

Figure 2 illustrates the global tracking framework of our method. In the following, we briefly introduce the tracking framework from two aspects: 1) feature extracting and encoding; 2) tracking by ensemble of local trackers.

Feature Extracting and Encoding. Taken as input the template image $\boldsymbol{I}_0$ (cropped from the initial image) and a sequence of testing images $\{\boldsymbol{I}_t\}_{t=1}^T$, our method first extracts their backbone features $\boldsymbol{F}_0^b \in \mathbb{R}^{H_z \times W_z \times C}$ and $\{\boldsymbol{F}_t^b \in \mathbb{R}^{H_s \times W_s \times C}\}_{t=1}^T$ with a backbone network. For efficiency, we then use a linear layer to reduce the channel number of the backbone feature from C to c.

To encode the information of the target to be tracked into the testing images, we use a transformer encoder stacking multi-head self-attention modules to perform fusion between the template image and the testing image, which has been proven to be effective by [4, 39]. Specifically, the features of $\boldsymbol{I}_0$ and $\boldsymbol{I}_t$ are concatenated after a flatten operation and fed into the encoder. Then, the feature pixels corresponding to $\boldsymbol{I}_t$ are retrieved from the output of the encoder and reshaped to a 3-D tensor denoted by $\boldsymbol{F}_t \in \mathbb{R}^{H_s \times W_s \times c}$, in which the target information is enhanced.

Tracking by Ensemble of Local Trackers. To achieve global tracking, we deploy an ensemble of local trackers to search for the target in different local regions in parallel. During tracking, if a local tracker is activated, i.e., locating the target, it will move following the target to keep tracking it in the subsequent frames. When the target moves smoothly, the activated local tracker can keep tracking the target in consecutive frames, forming an activated local tracker flow. Along with such flow, we can readily transfer and exploit the temporal context from multiple historical frames to perform local tracking in our framework. On the other hand, when the target moves discontinuously due to occlusion or disappearance, although the activated local tracker may lose the target, another local tracker close to the target can take over the tracking to locate the target.

In particular, we propose a deformable attention-based local tracker, which maintains a reference position and a target query to perform local tracking on top of the encoded feature $\boldsymbol{F}_t$. Every local tracker outputs a target embedding used for further predicting a candidate by a prediction head. Based on the local tracker, we design a temporal context transferring scheme, which transfers the target information using the reference position and target query as carriers along with the activated local tracker flow.

Denoting the reference position and the target query of the i-th local tracker by $\boldsymbol{p}^i$ and $\boldsymbol{q}^i$, respectively, the tracking process with N local trackers on $\boldsymbol{I}_t$ can be formulated as:

$$\begin{aligned} \{\boldsymbol{e}_t^i\}_{i=1}^N &= \Phi_{\mathrm{LT}}(\{\boldsymbol{q}^i\}_{i=1}^N, \{\boldsymbol{p}^i\}_{i=1}^N\}, \boldsymbol{F}_t), \\ \{y_t^i\}_{i=1}^N &= \Phi_{\mathrm{Head}}(\{\boldsymbol{e}_t^i\}_{i=1}^N). \end{aligned} \tag{1}$$

Herein $\boldsymbol{e}_t^i$ denotes the output target embedding of the i-th local tracker. $y_t^i = \{s_t^i, \boldsymbol{b}_t^i\}$ is the candidate predicted based on $\boldsymbol{e}_t^i$, where s_t^i is the foreground-background classification score and $\boldsymbol{b}_t^i$ is the bounding box. Φ_{LT} and Φ_{Head} denote the parallel local trackers and the head, respectively. Assuming the j-th local tracker is activated on $\boldsymbol{I}_t$, the temporal context transferring scheme can be formulated as:

$$\boldsymbol{p}^j \leftarrow \mathcal{T}_{\mathrm{rp}}(y_t^j), \boldsymbol{q}^j \leftarrow \mathcal{T}_{\mathrm{tq}}(\boldsymbol{e}_t^j), \tag{2}$$

where $\mathcal{T}_{\mathrm{rp}}$ and $\mathcal{T}_{\mathrm{tq}}$ denote the temporal context transferring via the reference position and the target query, respectively. The updated $\boldsymbol{p}^j$ and $\boldsymbol{q}^j$ are then used for tracking on $\boldsymbol{I}_{t+1}$.

3.2. Deformable Attention-based Local Trackers

To perform local tracking within the global view, our local tracker should be able to search for the target in a dynamic local region on the encoded feature $\boldsymbol{F}_t$ of the testing image $\boldsymbol{I}_t$. So that we can move the activated local tracker following the target by changing its search region according to the previous tracking result. To this end, we opt for the Deformable Attention [47] to implement our local tracker due to its ability of adaptive and sparse sampling around a reference location. Thus, we can move the local tracker by setting a new reference position for it.

Every local tracker maintains a target query and a reference position. The target query models the potential target information in appearance and location, while the reference position determines the search region of the local tracker coarsely. In particular, the default target query of a local tracker is an offline learned embedding, and the default reference position is predicted from the target query via a linear layer and sigmoid function. To perform local tracking, a local tracker computes the attention between its target query and the feature pixels sampled from $\boldsymbol{F}_t$ around its reference position, in which the sample positions are produced from its target query by predicting the coordinate offsets to its reference position via a linear layer. Through the attention operation, every local tracker outputs a target embedding that models the appearance and location information of a candidate in the corresponding local search region. Technically, all local trackers are implemented with a decoder, and the parallel local tracking process Φ_{LT} is defined as:

$$\boldsymbol{\Phi}_{\mathrm{LT}}(\{\boldsymbol{q}^i\}_{i=1}^N, \{\boldsymbol{p}^i\}_{i=1}^N\}, \boldsymbol{F}_t) = A_{dc}(A_{ms}(\boldsymbol{q}), \boldsymbol{p}, \boldsymbol{F}_t), \tag{3}$$

where A_{dc} and A_{ms} denote the deformable cross-attention [47] and the multi-head self-attention [34] functions, respectively. $\boldsymbol{q} = \boldsymbol{q}^1 \oplus \cdots \oplus \boldsymbol{q}^N$, $\boldsymbol{p} = \boldsymbol{p}^1 \oplus \cdots \oplus \boldsymbol{p}^N$, and $\oplus$ denotes the concatenation operation.

It is worth noting that the self-attention operation that models the interaction between all target queries promotes the learned reference positions of all local trackers to spread over the image reasonably during training, which is crucial for learning an effective arrangement of local trackers.

3.3. Temporal Context Transferring

To exploit the temporal context in our global tracking framework, we use the reference position and the target query as the carriers to transfer the temporal context along with the activated local tracker flow. Next, we detail this process assuming that the j-th local tracker keeps activated.

The predicted target location in the previous frame is a direct clue denoting where to search for the target locally in the subsequent frame. We therefore directly use it as the new reference position of the activated local tracker for tracking the subsequent frame. Thus, $\mathcal{T}_{\mathrm{rp}}$ is defined as:

$$\mathcal{T}_{\mathrm{rp}}(y_t^j) = \boldsymbol{c}_t^j, \tag{4}$$

where $\boldsymbol{c}_t^j$ is the center of the predicted target bounding box $\boldsymbol{b}_t^j$ in the previous testing image $\boldsymbol{I}_t$. In this way, the activated local tracker will move following the target until the target is located by another local tracker or disappears. After losing the target, the reference position of this local tracker will be reset to its default value. Namely, this local tracker will move back to where it starts moving.

In addition, the target embedding predicted by an activated local tracker contains the new target information in appearance and location, and thus it can be naturally used to generate a new target query for the activated local tracker to track in the subsequent frame. To this end, we propose a Temporal Context Aggregation (TCA) model to aggregates the temporal context modeled in the recently predicted target embeddings to generate a new target query. We refer to such a new target query as the *online target query* for clarity.

As shown in Figure 3, our TCA model maintains a memory of the online target queries generated in the recent L frames $\boldsymbol{q}_{mem}^j = \boldsymbol{q}_{t-L+1}^j \oplus \cdots \oplus \boldsymbol{q}_t^j$. Such a memory mod-

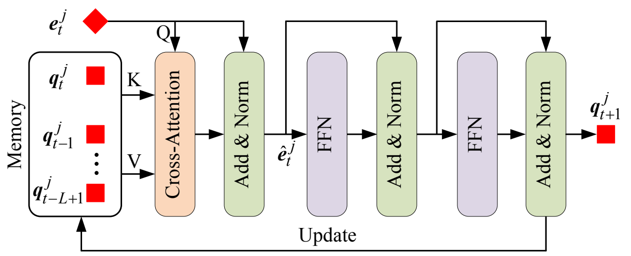

Figure 3. **Architecture of the proposed temporal context aggregation model.** It performs the interaction between the target embedding and the historical online target queries to aggregate the temporal context information modeled by these queries.

els the historical appearance and trajectory information of the target in the recent frames. To aggregate this historical information, the TCA model computes the cross-attention between the target embedding $\boldsymbol{e}_t^j$ (serving as the query) and the memory $\boldsymbol{q}_{mem}^j$ (serving as the key and value) with a multi-head cross-attention layer. Then, the output of cross-attention and the target embedding $\boldsymbol{e}_t^j$ are added and normalized to generate an aggregated target embedding $\hat{\boldsymbol{e}}_t^j$. Finally, two feed-forward networks (FFNs) with a skip connection followed by a normalization layer are used to adjust $\hat{\boldsymbol{e}}_t^j$ to generate the online target query $\boldsymbol{q}_{t+1}^j$ used for tracking on $\boldsymbol{I}_{t+1}$. Formally, $\mathcal{T}_{\text{tq}}$ can be defined as:

$$\mathcal{T}_{\text{tq}}(\boldsymbol{e}_t^j) = \phi_{adj}(\phi_{norm}(A_{mc}(\boldsymbol{e}_t^j, \boldsymbol{q}_{mem}^j) + \boldsymbol{e}_t^j)), \tag{5}$$

where A_{mc}, ϕ_{norm} and ϕ_{adj} denote the cross-attention, normalization, and adjustment operations, respectively. We also use $\boldsymbol{q}_{t+1}^j$ to update the memory and pop out the oldest query. For a newly activated local tracker flow, the memory of the TCA model is empty. In this case, we direct feed the target embedding into the last two FFNs in Figure 3 to generate an online target query, omitting the other layers.

3.4. Supervised Learning of Local Trackers

To learn the temporal context modeling ability, we train the proposed model with sequence samples. In a sequence sample, the target patch cropped from the first frame is used as the template image, and the subsequent frames are used as the testing images. During training, we perform the forward propagation on the testing images following the pipeline shown in Figure 2.

To calculate the loss on the t-th testing image $\boldsymbol{I}_t$, we first use the Hungarian algorithm [21] to match the ground truth $\hat{y}_t = \{\hat{\boldsymbol{b}}_t\}$ with a predicted candidate $y_t^i = \{s_t^i, \boldsymbol{b}_t^i\}$. Herein $\hat{\boldsymbol{b}}_t$ denotes the ground truth bounding box. Based on the Hungarian loss in DETR [3], we further consider the ℓ_1-norm of the distance between the ground truth box center $\hat{\boldsymbol{p}}_t$ and the reference position $\boldsymbol{p}^i$ of the local tracker as a regularization. This regularization encourages the candidate predicted by the local tracker close to the target to be matched with the ground truth, benefiting the learning of our model in two aspects. First, it increases the probability that a local tracker keeps activated in continuous testing images, which is necessary for learning the temporal context modeling. Second, it benefits the local tracker to learn a reasonable deformable attention range. Otherwise, the local trackers tend to over-extend the deformable attention range in training, which would cause a local tracker to perceive too much background information and thus become vulnerable. Our Hungarian loss $\mathcal{L}_H(y_t^i, \hat{y}_t)$ is defined as:

$$\begin{aligned} &\mathcal{L}_H(y_t^i, \hat{y}_t) = \lambda_{cls}\mathcal{L}_{cls}(s_t^i) + \mathcal{L}_{box}(\boldsymbol{b}_t^i, \hat{\boldsymbol{b}}_t) + \lambda_r\mathcal{L}_{\ell_1}(\boldsymbol{p}^i, \hat{\boldsymbol{p}}_t), \\ &\mathcal{L}_{box}(\boldsymbol{b}_t^i, \hat{\boldsymbol{b}}_t) = \lambda_{\ell_1}\mathcal{L}_1(\boldsymbol{b}_t^i, \hat{\boldsymbol{b}}_t) + \lambda_{iou}\mathcal{L}_{iou}(\boldsymbol{b}_t^i, \hat{\boldsymbol{b}}_t). \end{aligned} \tag{6}$$

Here, $\mathcal{L}_{cls}$, $\mathcal{L}_{\ell_1}$, and $\mathcal{L}_{iou}$ refer to the focal loss [23], ℓ_1 loss, and generalized IoU loss [32], respectively. λ_{cls}, λ_r, λ_{ℓ_1}, and λ_{iou} are the balance weights. Note that we compute bipartite matching in every testing image instead of directly propagating the assignment in the first testing image to the following ones, as it also leads to the over-extension of the deformable attention range.

Denoting the index of the matched candidate in the t-th testing image as π_t, the training loss $\mathcal{L}$ for a sequence sample with T testing images can be formulated as:

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{N}\lambda_{cls}\mathcal{L}_{cls}(s_t^i) + \mathbb{1}_{\{i=\pi_t\}}\mathcal{L}_{box}(\boldsymbol{b}_t, \hat{\boldsymbol{b}}_t^i), \tag{7}$$

where $\mathbb{1}_{\{i=\pi_t\}}$ equals 1 if $i = \pi_t$ and 0 otherwise.

4. Experiments

4.1. Implementation Details

We adopt ResNet-50 [15] pre-trained on ImageNet [10] as our backbone, and the output of *conv*-4 is used as the backbone feature. We use the head model in DETR [3] to predict the candidate box and the corresponding classification score. Besides, we compute the cosine similarity between the target template and the candidate in the feature space, and then obtain the candidate confidence by multiplying the classification score and the cosine similarity. Similar to [44], We adopt a Hungarian algorithm [21] to select the final prediction from the candidates, which takes the confidence and the position of the candidate into account. Whether the target is present is determined by comparing the confidence score with a threshold θ. The template image is cropped from the initial frame centered on the ground truth target and then resized to 128×128, whose area is 2^2 times that of the target. The testing images are resized to 640×480. The local tracker number N and the memory length L are set to ten and five by default, respectively.

During training, the length of the sequence sample (including one template image and one or more testing images) gradually increases from 2 to 6. We use the training splits of COCO [24], LaSOT [13], TrackingNet [30], and GOT-10k [18] to train our model, and COCO is only used when the length of the sequence sample is 2. λ_{cls}, λ_r, λ_{ℓ_1}, and

Table 1. **Precision (Pre.), normalized precision (nPre.), and AUC for four variants of our method on LaSOT.**

Variants	Our model	OSDet	EGT	SGT
Pre.	0.732	0.707	0.690	0.670
nPre.	0.759	0.732	0.717	0.705
AUC	0.677	0.653	0.623	0.619

λ_{iou} are set to 1.0, 5.0, 5.0, 2.0, respectively. Source codes will be available at https://github.com/ZikunZhou/GTELT.

4.2. Ablation Study

We first conduct experiments to investigate the effectiveness of each proposed technique in our model. To this end, we perform ablation studies on four variants of our method:
1) **Our model**, our intact model that performs global tracking via ensemble of local trackers.
2) **OSDet**, which removes the temporal context transferring scheme from our model. Thus, our model degenerates into a global **O**ne-**S**hot **Det**ector in the transformer framework, using multiple offline learned queries to perform global re-detection without exploiting the temporal context.
3) **EGT**, replacing the deformable attention module in our model with the multi-head attention module, which computes attention globally and densely. Thus, the ensemble of local trackers in our model becomes an **E**nsemble of **G**lobal **T**rackers. Due to these global trackers do not involve reference positions, we perform temporal context transferring only using the target queries as the carriers.
4) **SGT**, reducing the number of target queries in EGT to one, i.e., adopting a **S**ingle multi-head attention-based **G**lobal **T**racker to perform tracking.

Table 1 presents the experimental results of these variants on the testing set of LaSOT [13].

Effect of the Temporal Context Transferring Scheme. The performance gap between our model and OSDet clearly demonstrates the effectiveness of the proposed temporal context transferring scheme for exploiting the historical appearances and locations of the target in global tracking.

Effect of the Deformable Attention-based Local Tracker. Compared with our method, the performance of EGT and SGT decreases by 5.4% and 5.8% in AUC, respectively. We attribute the performance drops to two reasons: 1) EGT and SGT cannot use the previous target location as a direct clue for locating the target in the subsequent frame; 2) Every query in EGT and SGT interacts with all feature pixels in $\boldsymbol{F}_t$, and thus the background information from the entire image inevitably overwhelms the target information. In addition, EGT using multiple queries performs marginally better than SGT using a single query, which is different from the result reported by STARK [39]. In STARK, using multiple queries leads to a performance drop compared with using a single query. We guess the reason for this difference is that more than one query is needed for locating the target when the search region becomes the entire image.

Table 2. **Ablation studies on the local tracker number (N) and the memory length (L) on LaSOT.**

	L=5			N=10				
	N=5	N=10	N=20	L=1	L=3	L=5	L=7	L=9
Pre.	0.712	0.732	0.705	0.725	0.726	0.732	0.731	0.730
nPre.	0.743	0.759	0.734	0.755	0.755	0.759	0.759	0.757
AUC	0.670	0.677	0.660	0.672	0.675	0.677	0.677	0.675

Effect of the local tracker number N and the memory length L. We also conduct experiments to investigate the effect of N and L. Table 2 reports the results of our model by varying N or L on LaSOT. We can observe that both sparse ($N = 5$) and dense ($N = 20$) local trackers damage the performance. We guess the reason is that sparse local trackers can hardly cover the full image while dense local trackers can hardly receive sufficient training since in each frame only the activated one receives the feedback from the supervision. Besides, tracking performance improves and saturates at about $L = 7$ along with increasing L.

4.3. Comparison with State-of-the-art Trackers

We compare our algorithm with state-of-the-art trackers on six datasets including TLP [29], OxUvA [33], VOT2020-LT [20], LaSOT [13], LaSOTExtSub [12], and TrackingNet [30]. The trackers involved in the comparison include three global trackers (DMTrack [44], Siam R-CNN [35], and GlobalTrack [17]), eight local-global switching strategy trackers (KeepTrack [27], KeepTrack-Fast [27], LT_DSE [20], CLGS [20], LTMU [5], SPLT [41], MBMD [43], and TLD [19]), and nine local trackers (STARK [39], TransT [4], TrDiMP [36], PrDiMP [9], AlphaRefine [40], SuperDiMP [6], MDNet [31], STM-Track [14], and SiamFC [1]). We discuss the experimental results per dataset and the running speed below.

TLP. TLP [29] includes 50 long sequences with an average sequence length of about 13,500 frames. Table 3 reports the AUC and precision scores on TLP. Compared with the local-global switching strategy tracker LTMU, our method improves the performance by 1.2% in AUC and 0.9% in precision. Besides, our method outperforms the other two global trackers, DMTrack and GlobalTrack, by a large margin (2.9%/6.9% in AUC and 2.0%/4.4% in precision, respectively), demonstrating the effectiveness of our method.

OxUvA. OxUvA [33] contains 166 long testing sequences. Besides the target bounding box, the OxUvA benchmark also requires trackers to predict whether the target is present. It uses the true positive rate (TPR), the true negative rate (TNR), and the maximum geometric mean (MaxGM) of TPR and TNR as performance metrics. We set the threshold θ to 0.6 for the evaluation on OxUvA. Table 4 presents the experimental results. Compared with the global tracker Siam R-CNN, our method achieves substantial performance gains of 4.5% in MaxGM. KeepTrack is a recently proposed local-global switching strategy tracker,

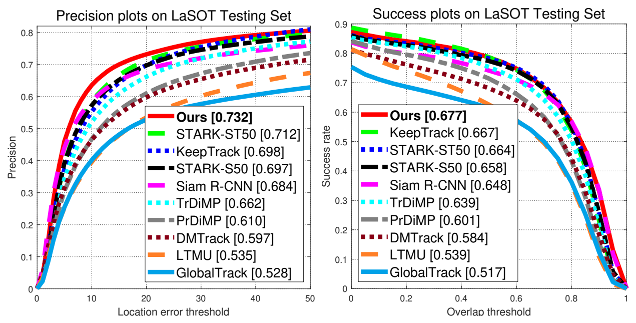

Figure 4. **Precision and success plots of different algorithms on the LaSOT testing set.**

Table 3. **AUC and precision of different trackers on TLP.** The best and second-best scores are marked by **bold** and <u>underline</u>, respectively. From left to right, trackers are divided into local trackers, local-global switching strategy trackers, and global trackers.

	SiamFC [1]	MDNet [31]	TLD [19]	SPLT [41]	LTMU [5]	Global Track [17]	DMTrack [44]	Ours
AUC	0.235	0.370	0.154	0.416	<u>0.558</u>	0.501	0.541	**0.570**
Pre.	0.284	0.384	0.167	0.403	<u>0.602</u>	0.567	0.591	**0.611**

Table 4. **TPR, TNR, and MaxGM of different trackers on the OxUvA testing set.** From left to right, the trackers are divided into local-global switching strategy trackers and global trackers.

	MBMD [43]	SPLT [41]	LTMU [5]	Keep Track [27]	Global Track [17]	DMTrack [44]	Siam R-CNN [35]	Ours
TPR	0.609	0.498	0.749	**0.806**	0.574	0.686	0.701	<u>0.764</u>
TNR	0.485	<u>0.776</u>	0.754	**0.812**	0.633	0.694	0.745	0.772
MaxGM	0.544	0.622	0.751	**0.809**	0.603	0.688	0.723	<u>0.768</u>

Table 5. **Precision, recall, and F-score of different trackers on the VOT2020-LT dataset.**

	Super DiMP [6]	CLGS [20]	KeepTrack Fast [27]	LT_DSE [20]	Keep Track [27]	DMTrack [44]	Ours
Precision	0.676	0.739	0.706	<u>0.715</u>	**0.723**	0.690	0.695
Recall	0.663	0.619	0.680	0.677	**0.697**	0.662	<u>0.690</u>
F-score	0.669	0.674	0.693	<u>0.695</u>	**0.709**	0.687	0.693

which uses an association network to deal with the distractors and performs better than our approach.

VOT2020-LT. VOT2020-LT [20] is a popular long-term tracking benchmark containing 50 challenging sequences. The VOT2020-LT benchmark requires trackers to predict a target bounding box and a corresponding confidence score. Based on the two predictions, precision, recall, and F-score are used as performance metrics. Table 5 reports the experimental results on VOT2020-LT. While KeepTrack achieves the best performance, our method performs on par with KeepTrackFast and sophisticated LT_DSE (VOT2020-LT winner), demonstrating the potential of our method.

LaSOT. LaSOT [13] contains 280 sequences in the testing set with an average sequence length of about 2,500 frames. It uses success, precision, and normalized precision as metrics. Figure 4 shows the precision and success plots on LaSOT. Our method achieves the best AUC and precision scores. Compared with the other global trackers, Siam R-CNN and DMTrack, our method achieves performance gains of 2.9%/9.3% in AUC and 4.8%/13.5% in precision, respectively. Besides, our method performs favorably against KeepTrack by performance gains of 1.0% in AUC and 3.4% in precision.

Table 6. **AUC, precision, and normalized precision on LaSOTExtSub and the speed of different trackers.** All speeds are reported on the RTX 2080Ti GPU other than those denoted by $*$.

	ATOM [7]	DiMP [2]	Super DiMP [6]	SPLT [41]	LTMU [5]	Keep Track [27]	Global Track [17]	Ours
AUC	0.376	0.392	0.433	0.272	0.414	**0.482**	0.356	<u>0.450</u>
Pre.	0.430	0.451	0.505	0.297	0.473	**0.564**	0.411	<u>0.524</u>
nPre.	0.459	0.476	0.524	0.339	0.499	**0.581**	0.436	<u>0.542</u>
FPS	**65**	<u>54</u>	39	26^*	13	18	6^*	26

Table 7. **AUC, precision, and normalized precision of different trackers on TrackingNet.** From left to right, trackers are divided into local trackers and global trackers.

	TrDiMP [36]	STM Track [14]	Alpha Refine [40]	STARK-ST50 [39]	TransT [4]	Global Track [17]	Siam R-CNN [35]	Ours
AUC	0.784	0.803	0.805	0.813	<u>0.814</u>	0.704	0.812	**0.825**
Pre.	0.731	0.767	0.783	–	<u>0.800</u>	0.656	<u>0.800</u>	**0.816**
nPre.	0.833	0.851	0.856	<u>0.861</u>	**0.867**	0.754	0.854	**0.867**

LaSOTExtSub. LaSOTExtSub [12] is an extension set of LaSOT containing 15 new classes with 10 sequences each. Many objects in LaSOTExtSub are small objects. Specifically, the percentage of small objects (whose area is smaller than 32^2 after resizing the entire image to 640×480) in LaSOTExtSub is 53.8%, while the number in LaSOT is 17.6%. The experimental results are reported in Table 6. KeepTrack performs better than our method. The reason is that KeepTrack resizes the search region whose area is 8^2 times that of the target to a fixed size, and this operation will increase the target resolution when the target is small. By contrast, our method always resizes the entire image to 640×480, which is not conducive to tracking small objects. Nevertheless, our method still outperforms the other long-term trackers, such as LTMU and GlobalTrack.

TrackingNet. Besides the long-term tracking datasets, we evaluate our method on a short-term tracking dataset, TrackingNet [30]. As shown in Table 7, our method performs favorably against the other transformer-based methods, TransT, STARK-ST50, and TrDiMP, by performance gains of 1.1%, 1.2%, and 4.1% in AUC, respectively.

Running Speed. Table 6 reports the running speed of different trackers. Our method runs at 26 FPS, reaching the real-time speed, and runs faster than the long-term trackers including KeepTrack, LTMU, and GlobalTrack.

4.4. Qualitative Comparison

To obtain a qualitative comparison, we show the tracking results on four challenging sequences in Figure 5, in which the main challenges are: full occlusion, appearance varia-

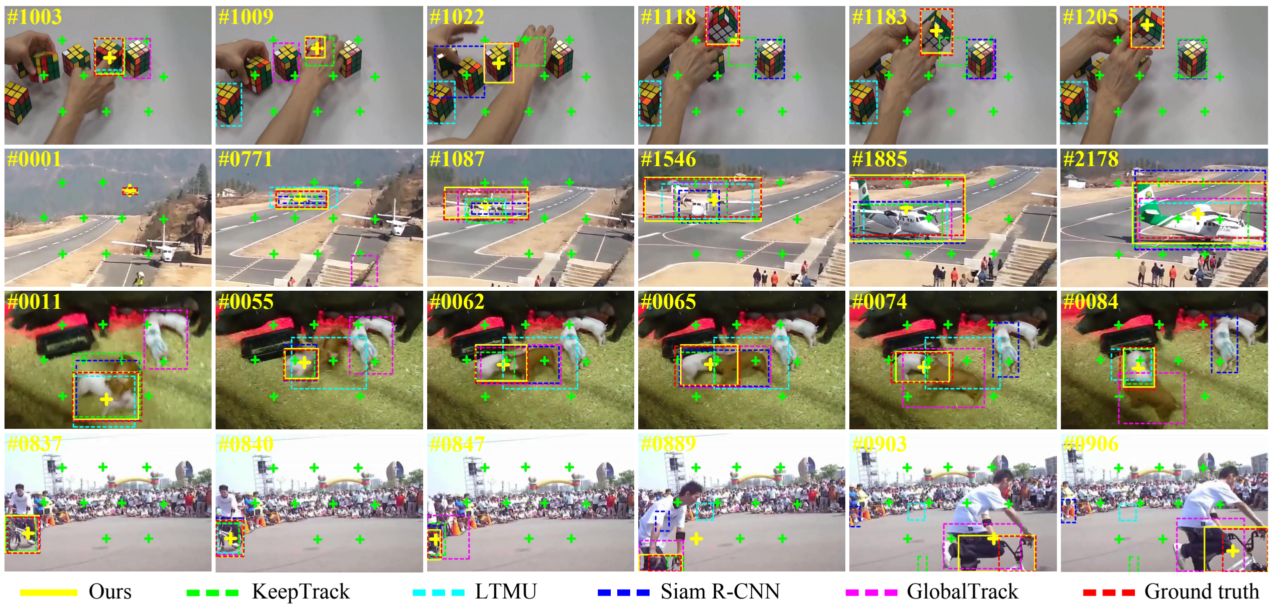

Figure 5. **Qualitative comparison on four challenging sequences.** From top to bottom, the main challenge factors are full occlusion, drastic appearance variation, distractor, and out-of-view, respectively. Our method is more robust than other long-term tracking methods.

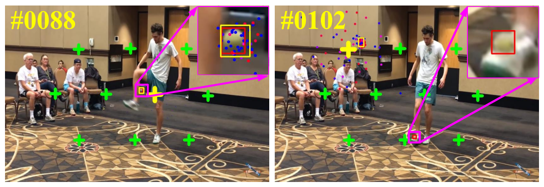

Figure 6. **Failure case of our method on a *footbag* sequence.** The targets in the two frames only occupy 0.35‰ (left) and 0.26‰ (right) of the image area, respectively. When the target becomes substantially small in the 102^{th} frame, whose information is overwhelmed by backgrounds, our method fails to locate it.

tion, distractor, and out-of-view, respectively. The qualitative comparison clearly shows that our method is more robust than other long-term trackers facing these challenges. For example, the first sequence shows the situation where the target is occluded and then reappears. When the target is almost fully occluded in the 1009^{th} frame, LTMU and GlobalTrack lose the target and drift to the distractor, while our method successes tracking the target. Although our method also drifts to a distractor when the target is fully occluded (the 1022^{th} frame), only our method successfully recovers to the real target when it reappears. KeepTrack predicts the target as absent since the 1009^{th} frame (still outputs a previously predicted bounding box). However, it fails to recover to the target when the target reappears and unfortunately drifts to a distractor at the 1205^{th} frame.

4.5. Limitations

Herein we discuss the limitations of our algorithm. Our local tracker is able to adjust its perception range by adaptively sampling the feature pixels from $\boldsymbol{F}_t$ to adapt to the objects of different sizes. However, this adaption mechanism does not work well for small objects. The reason is that small objects only occupy one or two feature pixels on $\boldsymbol{F}_t$, whose total stride is 16. Such limited target information tends to be overwhelmed by the backgrounds, making it arduous to locate the small objects. Figure 6 shows the failure case of our method on a *footbag* sequence. Besides, our method still cannot address the extremely challenging distractor issue well. As shown in the 1022^{th} frame in the first sequence in Figure 5, our method locates the distractor when the target disappears, as most long-term trackers do.

5. Conclusion

We have presented the global tracking algorithm via ensemble of local trackers, which can perform tracking in a global view while exploiting the temporal context. The smooth moving of the target can be handled by one single local tracker. If a local tracker loses the target due to discontinuous moving, another local tracker close to the target can take over the tracking to locate the target. Specifically, we design a deformable attention-based local tracker to simulate the local tracking mechanism within the global view. Further, we propose a temporal context transferring scheme to exploit the historical target appearances and locations for local tracking. The proposed method performs favorably against state-of-the-art trackers on six benchmarks.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (U2013210, 62006060, and 62172126), the Shenzhen Stable Support Plan Fund for Universities (GXWD20201230155427003-20200824125730001), and the Shenzhen Research Council (JCYJ20210324120202006 and JCYJ20210324132212030).

References

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCVW*, pages 850–865, 2016.

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, pages 6182–6191, 2019.

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.

[4] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135, 2021.

[5] Kenan Dai, Yunhua Zhang, Dong Wang, Jianhua Li, Huchuan Lu, and Xiaoyun Yang. High-performance long-term tracking with meta-updater. In *CVPR*, pages 6298–6307, 2020.

[6] Martin Danelljan and Goutam Bhat. Pytracking: Visual tracking library based on pytorch. *https://github.com/visionml/pytracking*, 2019.

[7] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669, 2019.

[8] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 6638–6646, 2017.

[9] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, pages 7183–7192, 2020.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009.

[11] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: A benchmark. In *CVPR*, pages 304–311, 2009.

[12] 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. *IJCV*, 129(2):439–461, 2021.

[13] 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 *CVPR*, pages 5374–5383, 2019.

[14] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. Stmtrack: Template-free visual tracking with space-time memory networks. In *CVPR*, pages 13774–13783, 2021.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[16] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Bridging the gap between detection and tracking: A unified approach. In *ICCV*, pages 3999–4009, 2019.

[17] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Globaltrack: A simple and strong baseline for long-term tracking. In *AAAI*, volume 34, pages 11037–11044, 2020.

[18] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE TPAMI*, 43(5):1562–1577, 2021.

[19] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE TPAMI*, 34(7):1409–1422, 2011.

[20] Matej Kristan, Aleš Leonardis, Jiří Matas, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kämäräinen, Martin Danelljan, Luka Čehovin Zajc, Alan Lukežič, Ondrej Drbohlav, et al. The eighth visual object tracking vot2020 challenge results. In *ECCV*, pages 547–601. Springer, 2020.

[21] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[22] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, pages 4282–4291, 2019.

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014.

[25] Alan Lukežič, Luka Čehovin Zajc, Tomáš Vojíř, Jiří Matas, and Matej Kristan. Fucolot–a fully-correlational long-term tracker. In *ACCV*, pages 595–611, 2018.

[26] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *CVPR*, pages 5388–5396, 2015.

[27] Christoph Mayer, Martin Danelljan, Danda Pani Paudel, and Luc Van Gool. Learning target candidate association to keep track of what not to track. In *ICCV*, pages 13444–13454, 2021.

[28] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[29] Abhinav Moudgil and Vineet Gandhi. Long-term visual object tracking benchmark. In *ACCV*, pages 629–645, 2018.

[30] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, pages 300–317, 2018.

[31] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, pages 4293–4302, 2016.

[32] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, pages 658–666, 2019.

[33] Jack Valmadre, Luca Bertinetto, Joao F Henriques, Ran Tao, Andrea Vedaldi, Arnold WM Smeulders, Philip HS Torr, and Efstratios Gavves. Long-term tracking in the wild: A benchmark. In *ECCV*, pages 670–685, 2018.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia

Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[35] Paul Voigtlaender, Jonathon Luiten, Philip HS Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *CVPR*, pages 6578–6588, 2020.

[36] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, pages 1571–1580, 2021.

[37] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE TPAMI*, 37(9):1834–1848, 2015.

[38] Tong Xiao, Shuang Li, Bochao Wang, Liang Lin, and Xiaogang Wang. Joint detection and identification feature learning for person search. In *CVPR*, pages 3415–3424, 2017.

[39] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, pages 10448–10457, 2021.

[40] Bin Yan, Xinyu Zhang, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Alpha-refine: Boosting tracking performance by precise bounding box estimation. In *CVPR*, pages 5289–5298, 2021.

[41] 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 *ICCV*, pages 2385–2393, 2019.

[42] Fangao Zeng, Bin Dong, Tiancai Wang, Cheng Chen, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv preprint arXiv:2105.03247*, 2021.

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

[44] Zikai Zhang, Bineng Zhong, Shengping Zhang, Zhenjun Tang, Xin Liu, and Zhaoxiang Zhang. Distractor-aware fast tracking via dynamic convolutions and mot philosophy. In *CVPR*, pages 1024–1033, 2021.

[45] Liang Zheng, Hengheng Zhang, Shaoyan Sun, Manmohan Chandraker, Yi Yang, and Qi Tian. Person re-identification in the wild. In *CVPR*, pages 1367–1376, 2017.

[46] Zikun Zhou, Wenjie Pei, Xin Li, Hongpeng Wang, Feng Zheng, and Zhenyu He. Saliency-associated object tracking. In *ICCV*, pages 9866–9875, 2021.

[47] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2020.

[48] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV*, pages 101–117, 2018.