

Unified Transformer Tracker for Object Tracking

Fan Ma^{1,3*} Mike Zheng Shou² Linchao Zhu¹

Haoqi Fan³ Yilei Xu³ Yi Yang⁴ Zhicheng Yan^{3†}

¹ReLER Lab, AAIL, University of Technology Sydney ²National University of Singapore

³Meta AI ⁴Zhejiang University

Abstract

As an important area in computer vision, object tracking has formed two separate communities that respectively study Single Object Tracking (SOT) and Multiple Object Tracking (MOT). However, current methods in one tracking scenario are not easily adapted to the other due to the divergent training datasets and tracking objects of both tasks. Although UniTrack [45] demonstrates that a shared appearance model with multiple heads can be used to tackle individual tracking tasks, it fails to exploit the large-scale tracking datasets for training and performs poorly on the single object tracking. In this work, we present the Unified Transformer Tracker (UTT) to address tracking problems in different scenarios with one paradigm. A track transformer is developed in our UTT to track the target in both SOT and MOT where the correlation between the target feature and the tracking frame feature is exploited to localize the target. We demonstrate that both SOT and MOT tasks can be solved within this framework, and the model can be simultaneously end-to-end trained by alternatively optimizing the SOT and MOT objectives on the datasets of individual tasks. Extensive experiments are conducted on several benchmarks with a unified model trained on both SOT and MOT datasets.

1. Introduction

Visual object tracking is one of the fundamental computer vision tasks with numerous applications [10, 25, 40, 55]. Unlike clearly defined object classification and detection problems [18, 19, 46, 50], object tracking is considered in different scenarios and can be categorized into two main paradigms 1) Single Object Tracking (SOT) is to track an annotated target from any object category in the first frame throughout a video [13, 25]; 2) Multiple Object Tracking (MOT) aims at estimating bounding boxes and identities of objects in videos where categories of targets are known and

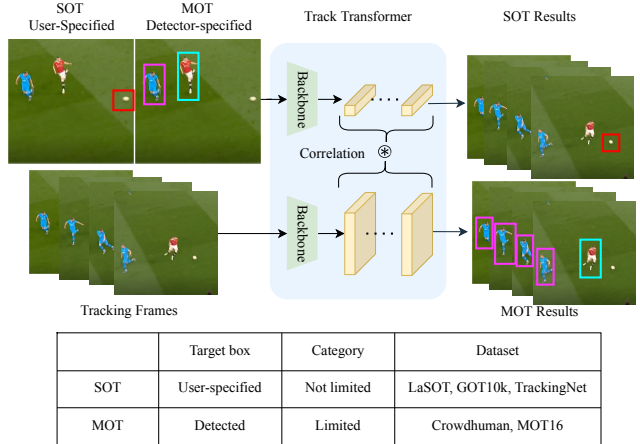


Figure 1. **Unified transformer tracker for both single object tracking (SOT) and multiple object tracking (MOT) task.** The target box in SOT is specified in the first frame while all boxes in reference frames are from the detection model in MOT. We use one tracking model to predict target localization in tracking frames for both tasks.

objects could appear or disappear [53, 55]. Current methods in the tracking community solve individual tasks separately by training models on the individual datasets for either SOT or MOT.

Siamese architecture is widely applied in SOT where various designs focus on improving the discriminative representation of objects [10, 13, 25, 49]. For MOT, tracking by detection is the most popular paradigm and achieves highest tracking performance on several benchmarks [2, 37, 55]. This paradigm is not applicable for SOT as the model would fail to detect objects of unseen categories in SOT. Some MOT methods [24, 60] use the Siamese tracker in SOT [4] to predict the location of the targets in tracking frames and fuse the predicted boxes with the detection boxes to enhance the detection results. However, these methods are not competitive to the tracking-by-detection methods in MOT. Albeit the Siamese trackers have been applied in both tracking tasks, none of these works are able to address two tracking

*This work was done during Fan’s internship at Meta AI.

†Correspondence author.

tasks with a unified paradigm. In practice, a unified tracking system is significant in many fields. For the AR/VR applications, tracking specific or unseen instances like personal cups is related to SOT while perceiving the environment of general classes like people is related to MOT. It is expensive and inefficient to maintain two separate tracking systems. The unified tracking system, which can easily switch tracking mode by demands, becomes more essential in real world deployment.

UniTrack [45] firstly attempts to address SOT and MOT concurrently by sharing the backbone and fusing multiple tracking heads. However, It fails to exploit large scale tracking datasets for training due to the head design and the divergent training datasets in different tasks. As shown in the Fig. 1, the training data in SOT and MOT are from various sources. Datasets in SOT only provide annotations of a single target in one video while dense object annotations are available in MOT datasets although the object categories are fixed. The tracking capacity of UniTrack [45] is thus limited, and the model would fail to track objects in complex scenarios.

In this paper, we introduce a Unified Transformer Tracker (UTT) as shown in Fig. 1 for solving both tracking tasks. For the tracking object in the reference frame, which is specified in SOT or detected in MOT, we provide a small feature map proposal in the tracking frame based on the previous localization. The target feature then correlates with the feature map proposal to update target representation and output the target localization. This enables our UTT to track objects in both SOT and MOT with the same design. The updated target features are further correlated with the new search feature proposal, which is cropped based on the produced target localization. This process is repeated several times to refine the localization of tracking targets. To enable exploiting training samples from both tasks, we alternatively train the network with datasets in each task. Experiments demonstrate that our tracker can be well learned on both tasks. Our main contributions are summarized as follows:

- We propose a novel Unified Transformer Tracker (UTT) for both single and multiple object tracking. To the best of our knowledge, this is the first work that the tracking model is end-to-end trained on both tasks.
- We develop a novel and effective track transformer to localize targets via the correlation between target feature and the tracking frame feature. The target features are well encoded through our transformer design.
- To verify the unified object tracking capability, we evaluate our UTT on both SOT and MOT benchmarks. Our proposed method achieves comparable performance to the state-of-the-art algorithms not only

on LaSOT [16], TrackingNet [32], and GOT-10k [22] in the SOT setting, but also on MOT16 [31] in the MOT setting.

2. Related Work

2.1. Single Object Tracking

The goal of single object tracking is to estimate the location of the object in a video sequence, while only the initial position is provided and the object category is unknown. It has been undergone astonishing progress in recent years, with the development of a variety of approaches, ranging from early correlation filter based methods [7, 14, 21], and recent Siamese network based trackers [4, 5, 62]. The pioneering work by Bolme *et al.* [7] proposes a minimum output sum of square error (MOSSE) filter. Siamese trackers has been used in various SOT methods [4, 10, 25, 44], which learn a general similarity map by cross-correlation between two image regions. Recently, transformer [41] has also been applied in the single object tracking community to improve the tracking performance via learning discriminative target representation [9, 43, 51]. Most SOT methods crop the targets and tracking frames in the reference image for extracting target representation. However, cropping every target and tracking frame becomes inefficient when multiple objects are specified in the tracking scenario. Differently, our tracker extracts target representation on the high-level feature map and constricts the search area by cropping feature maps. Rather than using the common cross attention in the transformer, we adopt the correlation attention between the target feature and search feature to update target representation for tracking.

2.2. Multiple Object Tracking

Multi-object trackers focus on tracking an unknown number of objects from a fixed set of categories [15], which is often built on top of the tracking-by-detection paradigm via linking the detections across successive frames [23, 54]. Bea *et al.* [1] introduce the Siamese architecture for learning a dissimilarity metric between a pair of objects. FairMOT [55] learn the object detection task and appearance embedding task from a shared backbone to improve the association accuracy. CenterTrack [58] proposes to detect objects with center points and predict their offsets to the previous frame for tracking. Recently, several methods [40, 53] adopt the transformer-based detectors [8, 61] to propagate boxes between frames. In these methods, the encoder with self attention is first adopted to enhance feature representation, and the decoder is followed with cross attention where the learnable query features are replaced with detected object features in previous frames. Our tracker integrates the encoder and decoder into one object transformer where the

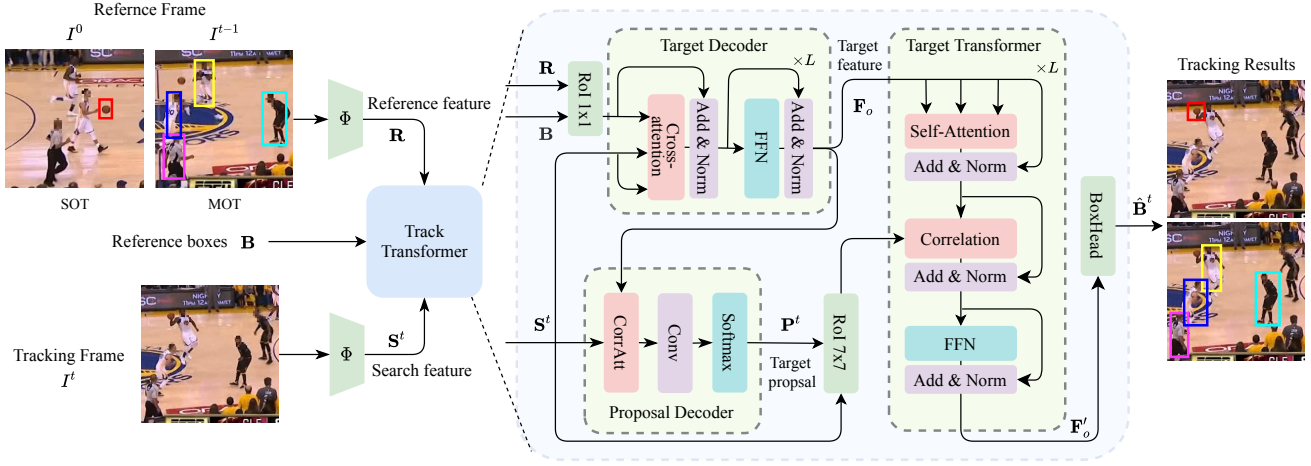


Figure 2. **Framework of our unified transformer tracker (UTT)**. We first use the backbone Φ to extract frame features. There are three inputs to the track transformer, including frame features of the reference and tracking frames, and reference boxes in the reference frame. The goal of track transformer is to predict the target localization in the tracking frame. The target decoder in the track transformer is first used to extract target features, and the proposal decoder is to produce candidate search areas in the tracking frame. Both target features and search features are fed to the target transformer to predict the target localization. Notations are detailed in the Sec. 3.2.

self attention is applied on the multiple target features and correlation attention is then adopted to update target features with the search features.

2.3. Unified Tracking

UniTrack [45] tackles both SOT and MOT problem via a shared backbone. For each task, a unique parameter-free task head is designed to cope with the tracking scenario. ImageNet pretrained features are directly used for all the tracking tasks, and the training recipe on all tracking tasks is not disclosed. It is thus still unknown whether the multiple tracking heads can be trained together on different tracking datasets. Besides, the tracking performance is not competitive to that of current state-of-the-art methods especially on the single object tracking task. In this paper, we propose a unified transformer tracker where different tasks share the same tracking head. It enables our tracker to be trained on both SOT and MOT datasets, and tracking performance on SOT is substantially improved.

3. Method

3.1. Overview

Existing tracking tasks can be categorized into single object tracking (SOT) and multiple-object tracking (MOT). We review their definitions below.

SOT. For a video sequence containing T frames, the tracking object is specified in the first frame I^0 . Let $\mathbf{B}^0 = \{(x_1, y_1, x_2, y_2)\}$ denote the object coordinates in the first frame, the goal of SOT is to localize the target $\{\hat{\mathbf{B}}^t\}_{t=1}^T$ in all T frames. In SOT, the tracking targets could be an object from any category, and the objects in testing data could belong to the unseen categories. The tracking performance is

directly evaluated by comparing the predicted localization $\{\hat{\mathbf{B}}^t\}_{t=1}^T$ with the groundtruth localization $\{\mathbf{B}^t\}_{t=1}^T$.

MOT. Objects from a fixed set of categories are detected and tracked in MOT. Suppose we have N detected objects $\mathbf{B}_d^{t-1} = \{b_i^{t-1}\}_{i=1}^N$ in the previous tracking frame I^{t-1} and M detected boxes $\mathbf{B}_d^t = \{b_i^t\}_{i=1}^M$ in the tracking frame I^t . The number of tracking objects is not fixed among frames as the new objects may appear and old objects may disappear in videos. The core challenge in MOT is to associate the previous N objects with current M objects, where the same object in frames should be assigned with a unique ID.

In our tracker, we predict the target boxes in the tracking frame $\hat{\mathbf{B}}^t \in \mathcal{R}^{N \times 4}$ for previous N objects, and then match them with current detected boxes \mathbf{B}_d^t . The performance of accurately tracked boxes $\{\hat{\mathbf{B}}^t\}_{t=1}^T$ in each frame is key to the overall MOT performance.

Taking a unified perspective, we treat the initial frame in SOT and the previous frame in MOT as the reference frame I , and the initial box in SOT and detected boxes in MOT as the reference boxes \mathbf{B} . The goal in our tracker is then to predict accurate target localization for tracking frame I^t . We propose a Unified Transformer Tracker as shown in Fig. 2 to address both SOT and MOT tasks. For the reference frame I , we first extract frame feature using the backbone as $\mathbf{R} = \Phi(I) \in \mathcal{R}^{H \times W \times C}$ where H and W is the height and width of the feature map. The frame feature can be extracted from different layers, here we only consider one layer for simplicity. The current tracking frame feature \mathbf{S}^t is also extracted in the same way. Then we feed these frame features with target boxes \mathbf{B} to the track transformer for predicting target localization.

3.2. Track Transformer

The track transformer is to produce the target coordinates in the tracking frame given reference boxes \mathbf{B} , the reference feature \mathbf{R} and the tracking frame feature \mathbf{S}^t . The tracking process can be expressed by

$$\hat{\mathbf{B}}^t = \mathcal{T}_\theta(\mathbf{S}^t, \mathbf{R}, \mathbf{B}), \quad (1)$$

where $\hat{\mathbf{B}}^t \in \mathcal{R}^{N \times 4}$ denotes the predicted localization for tracking targets. \mathcal{T}_θ denotes the track transformer parameterized by θ .

To localize targets in the tracking frame, we first extract the target feature using the *target decoder* in Fig. 2. For each tracking target, we then adopt the *proposal decoder* to generate candidate search area that may contain targets in the tracking frame. The target feature and proposal are further processed in the *target transformer* to predict the localization in the tracking frame.

3.2.1 Target Decoder

An intuitive way of forming target features is to apply the RoIAlign [18] on the reference feature map, calculated by

$$\mathbf{F} = \text{RoIAlign}(\mathbf{R}, \mathbf{B}) \in \mathcal{R}^{N \times C}. \quad (2)$$

However, this would neglect spatial background information, which is critical in object tracking especially when different tracking targets are similar in appearance. To embed more context information, we adopt Multi-head Cross Attention (MCA) to interact the target feature with the frame feature. The target feature is then update through

$$\mathbf{F}_c = \text{Norm}(\mathbf{F} + \text{MCA}(\mathbf{F}, \mathbf{S}^t, \mathbf{S}^t)), \quad (3)$$

where the target feature \mathbf{F} is the query and the frame feature \mathbf{S}^t represents the key and value in the MCA module. Norm denotes the instance normalization. The frame feature is flattened along with spatial dimensions so that there are HW values and keys.

In addition, we use a standard FFN module, which contains a fully connected feed-forward network that consists of two linear transformations with a ReLU in between. The output target feature is calculated by

$$\mathbf{F}_o = \text{Norm}(\mathbf{F}_c + \text{FFN}(\mathbf{F}_c)). \quad (4)$$

With the target decoder, we embed more context information into the target representation through the cross attention mechanism.

3.2.2 Proposal Decoder

We track every target by correlating the target feature with the tracking frame feature. However, correlating each target feature with the whole frame feature would incur huge

memory and computational cost when the track frame is of high resolution and multiple objects are required to be tracked. We introduce the *target proposal* to crop a unique search area for each object to reduce the search area in the tracking frame. A naive way of setting target proposal would be choosing it to be the tracking result of the last frame $\mathbf{P}^t = \hat{\mathbf{B}}^{t-1}$. However, the tracker would fail to localize the target if the previous tracking box $\hat{\mathbf{B}}^{t-1}$ is full of background and no tracking target is contained. The issue becomes more severe when the specified tracking target disappears from the video and appears at a new location later on. For instance, the tracking target in SOT could be lost if there are no targets in previous predicted localization. It is critical to generate more accurate target proposal in our track transformer.

To this end, we correlate the target feature with tracking frame feature for providing an effective target proposal in SOT. Specifically, we obtain the heat maps for targets via

$$\mathbf{H} = \text{Softmax}(\text{Conv}(\text{CorrAtt}(\mathbf{F}_o, \mathbf{S}^t))) \in \mathcal{R}^{N \times HW \times 2}, \quad (5)$$

where *Conv* contains several convolutional operations with ReLU activations. The *Softmax* operation is executed on the flattened the spatial dimension. The *CorrAtt* operation correlates the target feature with the tracking frame feature via

$$\text{CorrAtt}(\mathbf{F}_o, \mathbf{S}^t) = (\mathbf{F}_o * \mathbf{S}^t) \odot \mathbf{S}^t, \quad (6)$$

where $*$ is the convolution operation and \mathbf{F}_o is the 1×1 filter with one output channel. \odot is the broadcast element-wise product.

The heat map \mathbf{H} indicates the probability distribution of top-left and bottom-right of targets. We follow the process in [48] to generate the target proposal by

$$\mathbf{P}^t = \sum_{i=0}^W \sum_{j=0}^H (i * \mathbf{H}_{i,j}^0, j * \mathbf{H}_{i,j}^0, i * \mathbf{H}_{i,j}^1, j * \mathbf{H}_{i,j}^1), \quad (7)$$

where i and j denote the x and y coordinate, respectively. $\mathbf{H}_{i,j}^0$ and $\mathbf{H}_{i,j}^1$ denote the probability of top-left and right-bottom of the target at the localization (i, j) , respectively.

3.2.3 Target Transformer

We first crop a search feature map for each object in the tracking frame given the proposals \mathbf{P}^t by

$$\mathbf{S}_{RoI}^t = \text{RoIAlign}(\mathbf{S}^t, \mathbf{P}^t) \in \mathcal{R}^{N \times K \times K \times C}, \quad (8)$$

where K is the size of the output feature map. In this way, we have a candidate search feature for each tracking target. We employ the attention mechanism to update target features via

$$\mathbf{F}_a = \text{Norm}(\mathbf{F}_o + \text{MSA}(\mathbf{F}_o)), \quad (9)$$

where MSA denotes the multi-head self-attention. We also use a sine function to generate spatial positional encoding following [61].

To capture the target information in the search area, we connect the target feature with search feature map through

$$\mathbf{F}_d = \text{Norm}(\mathbf{F}_a + \text{Correlation}(\mathbf{F}_a, \mathbf{S}_{RoI}^t)) \quad (10)$$

where Correlation updates the target feature through

$$\text{Correlation}(\mathbf{F}_a, \mathbf{S}_{RoI}^t) = \text{FC}(\text{CorrAtt}(\mathbf{F}_a, \mathbf{S}_{RoI}^t)), \quad (11)$$

where FC denotes the fully connected layer with the input channel K^2C and output channel C . The CorrAtt is the same as the operation in Eq. (6). In addition, a FFN module in Eq. (4) is applied to produce target features \mathbf{F}'_o as shown in Fig. 2. The output feature is then used to predict the target localization. Instead of directly producing coordinates, the box head outputs the delta bias to the proposal boxes for generating target localization.

We repeat the target transformer for L times to localize tracking targets. In the i^{th} ($i > 1$) target transformer, the target proposal is updated by the previous predicted localization to provide new candidate search area. Similarly, the target feature is also updated to correlate with the search feature. All the intermediate outputs are stored in $\{\hat{\mathbf{B}}_i^t\}_{i=1}^L$.

Differences from the original transformer. In the tracking community, all the previous transformer designs [9, 40, 43, 48, 51] use encoder-decoder pipeline to enhance the target representation. The target feature is treated as the query and the whole tracking frame feature is the key and value in the cross-attention operation. It becomes inefficient when the multiple specified objects are required to be tracked and the video is of high resolution. Suppose we have N objects to track and the size of the tracking frame feature map is $H \times W$ and the feature dimension is C . The complexity of the cross attention for all tracking objects will be $\mathcal{O}(HWN C)$. In contrast, we crop the fixed size $K \times K$ ($K < \min(H, W)$) search area from the frame feature. The complexity in our tracker is $\mathcal{O}(K^2 N C)$, which are more efficient. Moreover, the cross attention is replaced with our correlation attention, which is more effective for tracking objects in various scenarios. In general, our tracker is not only able to be end-to-end trained on both SOT and MOT tasks, but also achieves higher performance when testing on SOT and MOT datasets.

3.3. Training

To train our tracker, we calculate the loss between the predicted boxes and ground truth boxes. For MOT, we do not use the proposal decoder to generate initial candidate search areas for training. Instead, we add a Gaussian noise to the ground truth boxes \mathbf{B}^t for producing the initial proposals during training \mathbf{P}^t . This is more similar to the testing

phrase where detected boxes in previous frames are used as the proposals in the current tracking frame. Moreover, the model is unable to be trained if we produce the proposal with the proposal decoder module for all objects. This is due to the high resolution of the feature map and a large quantity of objects in MOT. Let $\mathbf{B}^t \in \mathcal{R}^{N \times 4}$ be the ground truth boxes in t^{th} frame, the loss function in MOT can be written as

$$\mathcal{L}_{MOT}^{box} = \sum_{i=1}^L \lambda_G \mathcal{L}_{GIoU}(\hat{\mathbf{B}}_i^t, \mathbf{B}^t) + \lambda_1 \mathcal{L}_1(\hat{\mathbf{B}}_i^t, \mathbf{B}^t), \quad (12)$$

where \mathcal{L}_{GIoU} and \mathcal{L}_1 denote the generalized IoU loss [36] and 1-norm loss, respectively. λ_G and λ_1 are the hyper-parameters to balance the box loss.

For single object tracking, the target proposal is initialized by the proposal decoder in Fig. 2. To produce a robust proposal for the single object, we also consider the box loss between the initial proposal and the ground truth \mathbf{B}^t . The loss function for the SOT is then expressed by

$$\mathcal{L}_{SOT}^{box} = \sum_{i=0}^L \lambda_G \mathcal{L}_{GIoU}(\hat{\mathbf{B}}_i^t, \mathbf{B}^t) + \lambda_1 \mathcal{L}_1(\hat{\mathbf{B}}_i^t, \mathbf{B}^t), \quad (13)$$

where the initial prediction $\hat{\mathbf{B}}_i^0 = \mathbf{P}^t$ is calculated in the target decoder via Eq. (7).

For single and multiple object tracking, we train the model with Eq. (13) and Eq. (12), respectively. Note that the training datasets are different in the two tasks. SOT datasets only provide a unique target annotation in one video where the dense boxes with classes are given in the MOT datasets. Moreover, object detection is necessary in the MOT as new objects should be also tracked. To this end, we simply add a detection head in our UTT for the detection task. In this paper, we implement the deformable DETR [61] with a set-based Hungarian loss [8] to optimize the detection head. More details of the training paradigm with the detection head are given in the appendix.

To enable a single model for both SOT and MOT tasks, we alternatively train the network with SOT and MOT datasets at each iteration. Specifically, we implement two data loaders for each task. At each iteration, two frames in a video from one data loader are sampled and batched as inputs to our tracker for optimization. For the SOT iteration, we use the target decoder to improve target representation and use the proposal decoder to produce initial target proposals in tracking frames. For the MOT iteration, all annotated objects are used for optimizing the detection head and objects in both frames are used for optimizing the track transformer with Eq. (12). We provide the detailed optimization procedure in the appendix.

Type	Method	LaSOT [16]		TrackingNet [32]		GOT10K [22]		
		Success	Precision	Success	Precision	AO	SR _{0.5}	SR _{0.75}
SOT-Only	SiamFC [4]	33.6	33.9	57.1	66.3	34.8	35.3	9.8
	MDNet [33]	39.7	37.3	60.6	56.5	29.9	30.3	9.9
	ECO [12]	32.4	30.1	55.4	49.2	31.6	30.9	11.1
	VITAL [39]	39.0	36.0	-	-	35.0	36.0	9.0
	GradNet [26]	36.5	35.1	-	-	-	-	-
	SiamDW [57]	38.4	35.6	-	-	41.6	47.5	14.4
	SiamRPN++ [25]	49.6	49.1	73.3	69.4	51.7	61.6	32.5
	ATOM [11]	51.5	50.5	70.3	64.8	55.6	63.4	40.2
	DiMP [5]	56.9	56.7	74.0	68.7	61.1	71.7	49.2
	SiamFC++ [47]	54.3	54.7	75.4	70.5	59.5	69.5	47.9
	D3S [29]	-	-	72.8	66.4	59.7	67.6	46.2
	MAMLTrack [42]	52.3	53.1	75.7	72.5	-	-	-
	SiamAttn [49]	56.0	-	75.2	-	-	-	-
	SiamCAR [17]	50.7	51.0	-	-	56.9	67.0	41.5
	SiamBAN [10]	51.4	52.1	-	-	-	-	-
	KYS [6]	55.4	55.8	74.0	68.8	63.6	75.1	51.5
	Ocean [56]	52.6	52.6	70.3	68.8	59.2	69.5	46.5
	TrDiMP [43]	63.9	61.4	78.4	73.1	68.8	80.5	59.7
	TransT [9]	64.9	69.0	81.4	80.3	72.3	82.4	68.2
Unified	UniTrack [45]	35.1	32.6	59.1	51.2	-	-	-
	UTT (ours)	64.6	67.2	79.7	77.0	67.2	76.3	60.5

Table 1. **Comparisons on TrackingNet, LaSOT, and GOT-10k.** All results are reported from original manuscripts. Our tracker outperforms the UniTrack by a large margin on the SOT datasets.

4. Experiment

4.1. Implementation Details

Training. We train our Unified transformer tracker on the training splits of COCO [27], TrackingNet [32], LaSOT [16], GOT10k [22], MOT16 [31], and CrowdHuman [38] datasets. ResNet [20] is used as the backbone with ImageNet-pretrained weights. The whole model is trained on 8 NVIDIA DGX A100 GPUs for 800K iterations with AdamW optimizer [28]. We use the cosine scheduler for decaying the learning rate from $1e-4$ to $1e-5$ with the weight decay $5e-3$. We have two data loaders for SOT and MOT training, and the two tasks are alternatively trained in the unified training mode. For the SOT setting, we set the batch size as 64, and images are cropped to 352x352 with common SOT data augmentations [13]. The SOT data loader contains both image and video datasets. The COCO [24] is applied with different transformations for generating image pairs during training. For the rest of video datasets, we randomly sample images with frame interval less than 200. For the MOT setting, we set the batch size as 16, and images are randomly resized following the data augmentations in MOT methods [40]. We also add a deformable DETR [61] head in our model for detecting objects.

Online Tracking. We evaluate our model on several datasets in both SOT and MOT tasks. For the single object tracking, we test our model on the testing splits of LaSOT [16], TrackingNet [32], and GOT10K [22] datasets. For the multiple object tracking, we validate our model on MOT16 [31] in two modes, using private or public object detection.

4.2. Evaluations in SOT

We compare our UTT with 20 state-of-the-art SOT methods on three datasets as shown in Tab. 1. Moreover, our track can run at 25 FPS on the SOT datasets.

LaSOT. LaSOT [16] contains 1,400 sequences with 1,120 for training and 280 for testing. We compare different trackers on LaSOT test set using Success and Precision metric. The results are reported in Tab. 1. Our UTT outperforms most of SOT trackers (*i.e.*, SiamRPN++ [25] and KYS [6]) on all datasets, and is competitive to the best tracker TransT [9]. Moreover, our proposed tracker outperforms UniTrack [45] by almost 30 points on the Success metric.

TrackingNet. TrackingNet [32] consists of 30k video sequences, and 511 videos are contained in the test set. As shown in Tab. 1, compared with the UniTrack [45], our UTT obtains 25.8% and 33.5% higher Success and Precision. Moreover, our method is competitive with the best task-specific SOT algorithm TransT [9], which crops both target and tracking frame and is unable to cope with multiple object tracking.

GOT10K. GOT10K [22] collects over 10k video segments and contains 180 testing sequences. We conduct the experiments and submit the tracking results to the official online server for evaluation. The detailed results of AO, SR_{0.5} and SR_{0.75} are reported in Tab. 1. Our tracker outperforms most of Siamese trackers, including SiamRPN++ [25], SiamFC++ [47], and SiamAttn [17]. Our unified tracker achieves second higher SR_{0.75} among all methods.

Type	Method	MOTA \uparrow	IDF1 \uparrow	MOTP \uparrow	MT \uparrow	ML \downarrow	FP \downarrow	FN \downarrow	IDs \downarrow
MOT-Only	TAP [59]	64.8	73.5	78.7	292	164	12980	50635	571
	CNNMTT [30]	65.2	62.2	78.4	246	162	6578	55896	946
	POI [52]	66.1	65.1	79.5	258	158	5061	55914	3093
	TubeTK_POI [34]	66.9	62.2	78.5	296	122	11544	47502	1236
	CTrackerV1 [35]	67.6	57.2	78.4	250	175	8934	48305	1897
	FairMOT [55]	74.9	72.8	-	447	159	-	-	1074
	TransTrack [40]	74.5	63.9	80.6	468	113	28323	112137	3663
Unified	UniTrack [45]	74.7	71.8	-	-	-	-	-	683
	UTT-DFDETR	69.2	53.9	80.1	417	39	22917	29050	4145
	UTT-FairMOT	74.2	63.4	81.4	322	140	7863	37147	2002

Table 2. **Comparisons on MOT Datasets.** \uparrow means the higher the better and \downarrow means the lower the better. UTT-DFDETR denotes that the deformable DETR [61] detection head is contained in our tracker while the UTT-FairMOT indicates that we use the detected objects from FairMOT [55]. Without using any person re-identification features, our tracker achieves highest MOTP among all methods.

4.3. Evaluations in MOT

We also assess our method on the MOT16 [31] for evaluation on the MOT task. The MOT16 [31] benchmark consists of a train set and a test set, each with 7 sequences and pedestrians annotated with full-body bounding boxes. Different aspects of MOT are evaluated by a number of individual metrics [3], and the results on MOT16 are reported in Tab. 2. We report UTT with different detection results on this dataset. First, we use the deformable DETR [61] as our detection head and train the detection head together with our track transformer (denoted by UTT-DFDETR). Our tracker runs at 8 FPS on this dataset. UTT-DFDETR outperforms recent CTracker [35] on the MOTA metric, which also uses ResNet50 as the backbone and manages to match same objects by using tracking localization. Following the UniTrack [45], our tracker also uses the detection results from FairMOT [55]. Instead of using person re-identification features to match detection results in FairMOT [55] or using Kalman filter to estimate target movement in UniTrack [45], our method directly predicts target localization given target features. Our tracker achieves best MOTP score (81.4) among all methods, and attains slightly lower MOTA score compared to FairMOT [55].

4.4. Ablative Studies

To verify the effectiveness of our designed transformer structure, we use the ResNet-18 as the backbone and train the model for 100K iterations. We validate our method on the LaSOT test set in the SOT setting and on the half training set in MOT following [58].

4.4.1 Target- and Proposal Decoder

To validate the target and proposal decoder on the tracking performance, we conduct the experiments on LaSOT and report the Success, Precision and OP75 metrics in Tab. 3. For the baseline model, the target feature is simply extracted with RoIAlign and the target proposal is the tracking result of the last frame. The tracking performance is largely im-

Target decoder	Proposal decoder	Success	Precision	OP75
\times	\times	48.1	44.7	38.3
\checkmark	\times	49.8	48.5	43.9
\times	\checkmark	58.6	59.4	53.9
\checkmark	\checkmark	59.2	59.8	54.3

Table 3. **Ablation study on the target and proposal decoder.** We report the Success, Precision, and OP75 on the LaSOT [16] for comparisons. The tracking performance is largely improved by using target and proposal decoders.

	LaSOT [16]		MOT16 [31]	
	Success	Precision	MOTA	IDF1
MCA	57.1	56.9	64.4	57.7
CorrAtt	57.6	57.9	64.9	61.4
CorrAtt + MSA	58.9	59.4	65.4	59.9

Table 4. **Comparing different correlation designs in the target transformer.**

proved when we add the proposal decoder. This is due to the fact that the tracking localization in previous frames could provide false proposals, containing no objects, and the localization would not be rectified in the tracking sequence. We further add the target decoder as proposed in Fig. 2 to enhance the target feature representation by interacting the target feature with the tracking frame feature, the tracking performance is also improved by 0.6 on the Success metric.

4.4.2 Target Transformer

Correlation. To show the superiority of our correlation layer in the target transformer in Eq. (11), we design a tracker using the multi-head cross attention (MCA) [9, 43]. Specifically, we replace the correlation module and self-attention module in Fig. 2 with the MCA module, where the target features are treated as query and cropped search features are used as key and value. We also remove the multi-head self-attention (MSA) to testify our designed module. The comparison results are shown in Tab. 4. Compared to the UTT-MCA tracker, our method with the correlation improves the Success by 1.8% on the SOT dataset, and MOTA by 1.0% on the MOT dataset.

Training	LaSOT [16]		MOT16 [31]	
	Success	Precision	MOTA	IDF1
SOT-Only	59.2	59.8	-	-
MOT-Only	12.5	5.1	64.3	62.1
Unified	58.9	59.4	65.4	59.9

Table 5. **Comparisons of different training modes.** Our model trained with both SOT and MOT datasets (Unified training) can be used for both tracking scenarios.

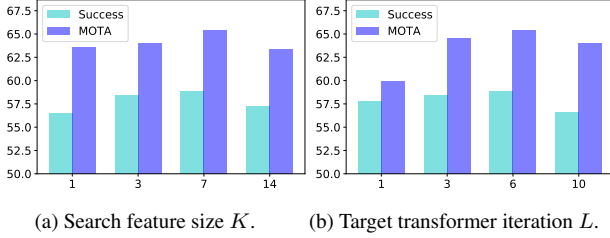


Figure 3. **Comparing different hyperparameter choices.** Success is the performance metric on LaSOT [16] and MOTA is the metric on MOT16 [31].

Search size. For each tracking target, we crop the search feature given the target proposal. The cropped feature interacts with the target feature for updating the localization. We thus validate the model with search features of different sizes in Fig. 3a. Highest tracking performance on both tasks is achieved when the pool size is set to 7.

Iteration. We testify the tracking performance with different number L of target transformer in Fig. 3b. It shows that the model with more iterations indeed achieves higher tracking performance. However, with more iterations, the training and inference speed will be slower as more parameters are contained and computational costs are higher.

4.4.3 Training Recipe

To investigate the tracking performance when trained in different settings, we conduct experiments with three different training modes in Tab. 5. The model is trained on COCO [27], LaSOT [16], GOT10K [22], and TrackingNet [32] in SOT, and is trained on CrowdHuman [38] and MOT16 [31] in MOT. All these datasets are used in the unified training mode. As shown in Tab. 5, the model trained on SOT is not able to track objects in MOT as no detection head is trained, while the model trained with MOT datasets performs poorly on the SOT datasets as only pedestrian annotations are provided. Compared to the MOT-Only, our unified model trained on both SOT and MOT datasets achieves higher MOTA and lower IDF1. The decreased IDF1 means that more IDs are assigned to a single object when dismissed or occluded (higher IDSW). Higher MOTA indicates that the false negative and false positive tracking objects decreased, demonstrating that our unified model provides more accurate tracking results.

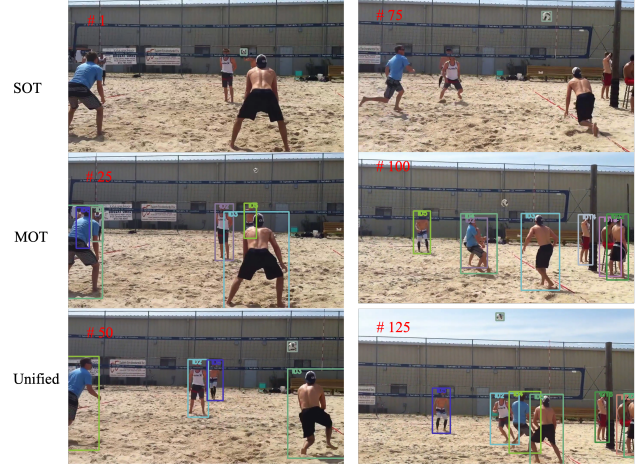


Figure 4. **Visualization of our results on the challenging sequences in different tracking modes.** Best viewed with zooming in. In the unified tracking mode, the specified volleyball is tracked together with multiple persons.

4.4.4 Visualization

We visualize the tracking results with different tracking modes in Fig. 4. The unified tracking mode indicates that our model does SOT and MOT concurrently, where the box of the volleyball is specified and boxes of persons are detected by the deformable DETR head [61]. We run the sequence three times with different tracking modes and the results in different timestamps are displayed. In the MOT and Unified tracking mode, our models assign the unique identification for each tracked person with the same color. From the figure, we observe that the person occluded by others can be also tracked well. It shows that both SOT and MOT tasks can be done concurrently during testing by our single model. The video demo of this sequence with different tracking modes is attached in the supplementary material.

5. Conclusion

Can a single model address different tracking tasks? We propose a Unified Transformer Tracker (UTT) to solve this problem, where the model can be end-to-end trained with datasets in both SOT and MOT tasks. A novel track transformer is introduced in this work to handle different tracking scenarios. From the experiments, we can draw the conclusion that one model is enough to address both SOT and MOT tasks. We believe this will encourage the community to develop more unified tracking algorithms that are applied into more wild tracking scenarios.

Acknowledgements Mike Shou is solely supported by the National Research Foundation, Singapore under its NRFF Award NRF-NRFF13-2021-0008. Linchao Zhu is partially supported by Australian Research Council Discovery Projects DP200100938.

References

- [1] Seung-Hwan Bae and Kuk-Jin Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *PAMI*, 40(3):595–610, 2017. 2
- [2] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019. 1
- [3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 7
- [4] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, pages 850–865, 2016. 1, 2, 6
- [5] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, pages 6182–6191, 2019. 2, 6
- [6] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene information for object tracking. In *ECCV*, pages 205–221, 2020. 6
- [7] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, pages 2544–2550, 2010. 2
- [8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing. 2, 5, 11
- [9] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135, June 2021. 2, 5, 6, 7
- [10] 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*, pages 6668–6677, 2020. 1, 2, 6
- [11] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669, 2019. 6
- [12] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, pages 6638–6646, 2017. 6
- [13] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, pages 7183–7192, 2020. 1, 6
- [14] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 2
- [15] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. Tao: A large-scale benchmark for tracking any object. In *ECCV*, pages 436–454, 2020. 2
- [16] 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. 2, 6, 7, 8
- [17] Dongyan Guo, Jianfeng Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *CVPR*, pages 6268–6276, 2020. 6
- [18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 4
- [19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1
- [20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6
- [21] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *PAMI*, 37(3):583–596, 2014. 2
- [22] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *PAMI*, 2019. 2, 6, 8
- [23] Hao Jiang, Sidney Fels, and James J Little. A linear programming approach for multiple object tracking. In *CVPR*, pages 1–8, 2007. 2
- [24] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *CVPR Workshops*, pages 33–40, 2016. 1, 6
- [25] 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. 1, 2, 6
- [26] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *ICCV*, pages 6161–6170, 2019. 6
- [27] 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. 6, 8
- [28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6
- [29] Alan Lukezic, Jiri Matas, and Matej Kristan. D3s - a discriminative single shot segmentation tracker. In *CVPR*, June 2020. 6
- [30] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools and Applications*, 78:7077–7096, 2018. 7
- [31] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *ArXiv*, abs/1603.00831, 2016. 2, 6, 7, 8
- [32] 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. 2, 6, 8
- [33] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, pages 4293–4302, 2016. 6
- [34] Bo Pang, Yizhuo Li, YiFan Zhang, Muchen Li, and Cewu Lu. Tubetk: Adopting tubes to track multi-object in a one-step training model. In *CVPR*, pages 6307–6317, 2020. 7
- [35] Jinlong Peng, Changhong Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *ECCV*, 2020. 7
- [36] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 5
- [37] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *ICCV*, pages 300–311, 2017. 1
- [38] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *ArXiv*, abs/1805.00123, 2018. 6, 8
- [39] Yibing Song, Chao Ma, Xiaohe Wu, Lijun Gong, Linchao Bao, Wangmeng Zuo, Chunhua Shen, Rynson W. H. Lau, and Ming-Hsuan Yang. Vital: Visual tracking via adversarial learning. In *CVPR*, pages 8990–8999, 2018. 6
- [40] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer, 2021. 1, 2, 5, 6, 7
- [41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 2
- [42] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *CVPR*, pages 6287–6296, 2020. 6
- [43] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, pages 1571–1580, June 2021. 2, 5, 6, 7
- [44] Qiang Wang, Zhu Teng, Junliang Xing, Jin Gao, Weiming Hu, and Stephen Maybank. Learning attentions: residual attentional siamese network for high performance online visual tracking. In *CVPR*, pages 4854–4863, 2018. 2
- [45] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? In *Advances in Neural Information Processing Systems*, volume 34, 2021. 1, 2, 3, 6, 7
- [46] Zuxuan Wu, Hengduo Li, Caiming Xiong, Yu-Gang Jiang, and Larry Steven Davis. A dynamic frame selection framework for fast video recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 1
- [47] Yinda Xu, Zeyu Wang, Zuoxin Li, Yuan Ye, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *AAAI*, page 12549–12556, 2020. 6
- [48] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, pages 10448–10457, October 2021. 4, 5
- [49] Kai Xiang Yang, Zhenyu He, Zikun Zhou, and Nana Fan. Siamatt: Siamese attention network for visual tracking. *Knowl. Based Syst.*, 203:106079, 2020. 1, 6
- [50] Le Yang, Junwei Han, and Dingwen Zhang. Colar: Effective and efficient online action detection by consulting exemplars. In *CVPR*, 2022. 1
- [51] Bin Yu, Ming Tang, Linyu Zheng, Guibo Zhu, Jinqiao Wang, Hao Feng, Xuetao Feng, and Hanqing Lu. High-performance discriminative tracking with transformers. In *ICCV*, pages 9856–9865, October 2021. 2, 5
- [52] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *ECCV Workshops*, 2016. 7
- [53] Fangao Zeng, Bin Dong, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv preprint arXiv:2105.03247*, 2021. 1, 2
- [54] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, pages 1–8, 2008. 2
- [55] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision*, 129(11):3069–3087, 2021. 1, 2, 7, 12
- [56] Zhipeng Zhang and Houwen Peng. Ocean: Object-aware anchor-free tracking. In *ECCV*, 2020. 6
- [57] Zhipeng Zhang, Houwen Peng, and Qiang Wang. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, pages 4586–4595, 2019. 6
- [58] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, 2020. 2, 7
- [59] Zongwei Zhou, Junliang Xing, Mengdan Zhang, and Weiming Hu. Online multi-target tracking with tensor-based high-order graph matching. In *ICPR*, pages 1809–1814, 2018. 7
- [60] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *ECCV*, pages 366–382, 2018. 1
- [61] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *ArXiv*, abs/2010.04159, 2021. 2, 5, 6, 7, 8, 11
- [62] 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. 2

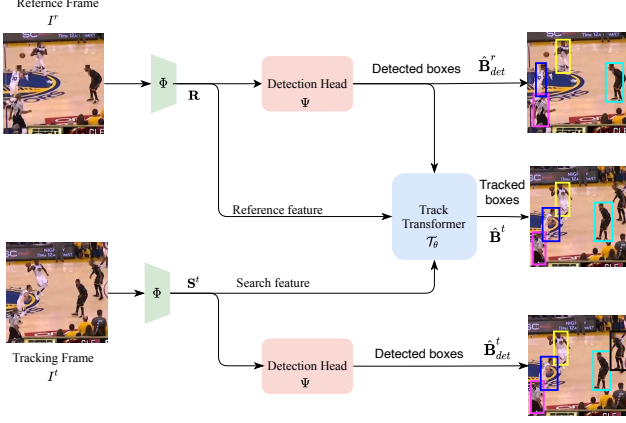


Figure 5. UTT with the detection head.

A. MOT Detection & Tracking

In the multiple object tracking, objects should be detected and tracked. In Fig. 2, we only include the track transformer to predict target localization given previous target coordinates. In this part, we display our Unified Transformer Tracker with the detection head in Fig. 5. In this work, we use the deformable DETR [61] as the detection head Ψ . For the MOT training, the backbone Φ , detection head Ψ , and the track transformer \mathcal{T}_θ are trained together.

During tracking, we first detect all the objects \hat{B}_{det}^r in the reference frame I^r . The detected boxes \hat{B}_{det}^r are then used as target proposals for the tracking frame. We feed the tracking frame feature, target proposals \hat{B}_{det}^r , and the reference feature to the track transformer and produce the tracked boxes \hat{B}^t . Moreover, the detection head also produces the object detection results \hat{B}_{det}^t in the tracking frame. By calculating the IoU between the tracked boxes \hat{B}^t and the detected boxes \hat{B}_{det}^t , we can match the same objects in both previous reference and current tracking frames. The IoU threshold is set to 0.9 for matching the same objects. For the unmatched tracked boxes, we mark them as the lost objects and track them in the latter frames. For the unmatched detected boxes, we assign new identifications to them as they are the new objects appearing in the video. In this way, we track all objects in the MOT and assign the same objects with the same identifications.

B. Unified Training

To train our Unified transformer tracker on both SOT and MOT tasks, we create two data loaders \mathcal{D}_{sot} and \mathcal{D}_{mot} . The detailed procedure to update models with both datasets is described in Algorithm 1.

Specifically, we use \mathcal{D}_{sot} and \mathcal{D}_{mot} to update the model in each iteration. In SOT, we only calculate the tracking box loss to update the backbone Φ and the track transformer \mathcal{T}_θ .

Algorithm 1 Unified Training of UTT on both SOT and MOT

- 1: **Input:** \mathcal{D}_{sot} , \mathcal{D}_{mot} , backbone Φ , detection head Ψ and tracker transformer \mathcal{T}_θ
 - 2: **for** $iter \leftarrow 0$ to max_iter **do**
 - 3: Updating model with SOT datasets via Algorithm 2
 $Alg_{sot}(\mathcal{D}_{sot}, \Phi, \mathcal{T}_\theta)$
 - 4: Updating model with MOT datasets via Algorithm 3
 $Alg_{mot}(\mathcal{D}_{mot}, \Phi, \Psi, \mathcal{T}_\theta)$
 - 5: **end for**
-

Algorithm 2 SOT Iteration Alg_{sot}

- 1: **Input:** \mathcal{D}_{sot} , backbone Φ and tracker transformer \mathcal{T}_θ
 - 2: Sampling $(I^r, I^t, \mathbf{B}^r, \mathbf{B}^t)$ from \mathcal{D}_{sot}
 - 3: Predicting target localization in reference frames I^r :
 $\hat{\mathbf{B}}^r = \mathcal{T}_\theta(\Phi(I^r), \Phi(I^r), \mathbf{B}^r) = \{\hat{\mathbf{B}}_i^r\}_{i=0}^L$
 - 4: Predicting target localization in tracking frames I^t :
 $\hat{\mathbf{B}}^t = \mathcal{T}_\theta(\Phi(I^t), \Phi(I^r), \mathbf{B}^r) = \{\hat{\mathbf{B}}_i^t\}_{i=0}^L$
 - 5: Calculating SOT loss via Eq. (13):

$$\mathcal{L}_{SOT}^{box} = \sum_{j \in \{t, r\}} \sum_{i=0}^L \lambda_G \mathcal{L}_{IoU}(\hat{\mathbf{B}}_i^j, \mathbf{B}^j) + \lambda_1 \mathcal{L}_1(\hat{\mathbf{B}}_i^j, \mathbf{B}^j)$$
 - 6: Updating Φ and \mathcal{T}_θ
-

Algorithm 3 MOT Iteration Alg_{mot}

- 1: **Input:** \mathcal{D}_{mot} , backbone Φ , detection head Ψ and tracker transformer \mathcal{T}_θ
 - 2: Sampling $(I^r, I^t, \mathbf{B}^r, \mathbf{B}^t)$ from \mathcal{D}_{sot}
 - 3: Predicting object detection in reference frames I^r :
 $\hat{\mathbf{B}}_{det}^r = \Psi(\Phi(I^r))$
 - 4: Calculating the detection loss:
 $\mathcal{L}_{MOT}^{det} = SetCriterion(\hat{\mathbf{B}}_{det}^r, \mathbf{B}^r)$
 - 5: Predicting target localization in tracking frames I^t :
 $\hat{\mathbf{B}}^t = \mathcal{T}_\theta(\Phi(I^t), \Phi(I^r), \mathbf{B}^r) = \{\hat{\mathbf{B}}_i^t\}_{i=1}^L$
 - 6: Calculating MOT tracking loss via Eq. (12):

$$\mathcal{L}_{MOT}^{box} = \sum_{i=1}^L \lambda_G \mathcal{L}_{IoU}(\hat{\mathbf{B}}_i^t, \mathbf{B}^t) + \lambda_1 \mathcal{L}_1(\hat{\mathbf{B}}_i^t, \mathbf{B}^t)$$
 - 7: Calculating the MOT loss:
 $\mathcal{L}_{MOT} = \mathcal{L}_{MOT}^{box} + \mathcal{L}_{MOT}^{det}$
 - 8: Updating Φ , Ψ , and \mathcal{T}_θ
-

In the SOT iteration, we use the target decoder to extract target features, and use the proposal decoder to produce candidate search areas for the target transformer. The produced proposal $\hat{\mathbf{B}}_0^t$ is thus used to calculate the loss in Eq. (13). For the MOT iteration, the detection head Ψ is also adopted to predict object detection in frames. In this work, we employ the Deformable DETR [61] as the detection head. The detection head predicts both box localization and classification results. For simplicity, we only present the box notation in Algorithm 3. The set criterion loss in [8] is used

seq	MOTA \uparrow	MOTP \uparrow	IDF1 \uparrow	IDP \uparrow	IDR \uparrow	TP \uparrow	FP \downarrow	FN \downarrow	Rcll \uparrow	Prcn \uparrow	MTR \uparrow	PTR	MLR \downarrow	MT \uparrow	PT \downarrow	ML \downarrow	IDSW \downarrow	FAR \downarrow	FM \downarrow
MOT16-01	57.36	81.60	43.17	52.24	36.78	4116	386	2279	64.36	91.43	43.48	39.13	17.39	10	9	4	62	0.86	73
MOT16-03	89.18	82.45	73.16	74.87	71.53	96764	3134	7792	92.55	96.86	87.84	12.16	0.00	130	18	0	385	2.09	434
MOT16-06	61.93	80.48	59.35	71.76	50.60	7736	400	3802	67.05	95.08	33.48	42.08	24.43	74	93	54	190	0.34	191
MOT16-07	63.96	80.25	47.60	53.87	42.64	11780	1139	4542	72.17	91.18	40.74	55.56	3.70	22	30	2	201	2.28	251
MOT16-08	44.32	81.16	36.72	53.21	28.03	8231	585	8506	49.18	93.36	30.16	46.03	23.81	19	29	15	228	0.94	236
MOT16-12	58.30	80.14	62.05	73.67	53.60	5479	556	2816	66.05	90.79	33.72	44.19	22.09	29	38	19	87	0.62	100
MOT16-14	46.32	74.68	45.88	56.23	38.74	11073	1663	7410	59.91	86.94	23.17	48.78	28.05	38	80	46	849	2.22	442
OVERALL	74.22	81.39	63.36	69.42	58.27	145179	7863	37147	79.63	94.86	42.42	39.13	18.45	322	297	140	2002	1.33	1727

Table 6. Detailed results on MOT16 test set with UTT FairMOT. All results are reported from online evaluation sever. The detected boxes are from FairMOT [55].

seq	MOTA \uparrow	MOTP \uparrow	IDF1 \uparrow	IDP \uparrow	IDR \uparrow	TP \uparrow	FP \downarrow	FN \downarrow	Rcll \uparrow	Prcn \uparrow	MTR \uparrow	PTR	MLR \downarrow	MT \uparrow	PT \downarrow	ML \downarrow	IDSW \downarrow	FAR \downarrow	FM \downarrow
MOT16-01	47.27	78.03	41.49	45.27	38.30	4265	1145	2130	66.69	78.84	39.13	47.83	13.04	9	11	3	97	2.54	137
MOT16-03	87.00	80.99	62.98	65.36	60.76	94373	2819	10183	90.26	97.10	84.46	14.19	1.35	125	21	2	588	1.88	997
MOT16-06	55.23	78.63	51.69	52.03	51.36	9124	2266	2414	79.08	80.11	50.68	44.34	4.98	112	98	11	485	1.90	326
MOT16-07	58.49	80.17	42.59	43.07	42.13	12904	3062	3418	79.06	80.82	50.00	50.00	0.00	27	27	0	295	6.12	368
MOT16-08	31.76	79.15	35.45	32.33	39.22	13042	7258	3695	77.92	64.25	57.14	42.86	0.00	36	27	0	468	11.61	482
MOT16-12	44.70	81.89	59.52	56.30	63.13	6550	2752	1745	78.96	70.41	52.33	45.35	2.33	45	39	2	90	3.06	199
MOT16-14	39.39	76.16	34.82	36.75	33.07	13018	3615	5465	70.43	78.27	38.41	48.78	12.80	63	80	21	2122	4.82	565
OVERALL	69.22	80.17	53.94	54.88	53.03	153276	22917	29050	84.07	86.99	54.94	39.92	5.14	417	303	39	4145	3.87	3074

Table 7. Detailed results on MOT16 test set with UTT-DFDETR. All results are reported from online evaluation sever. The detected boxes are produced using our deformable detection head.

to optimize the detection part. Also we calculate the tracking loss with Eq. (12). Instead of using target proposals, the target proposals in the MOT iteration are produced by adding Gaussian noise to the ground truth boxes. To ensure that the target is included in the proposal, we set the minimum IoU as 0.1 between the ground truth proposal and the noise proposal. The backbone Φ , detection head Ψ , and the track transformer \mathcal{T}_θ are then updated with both detection and tracking losses.

C. Detailed Results on MOT16

The tracking results are submitted to the online server for evaluation. We report the detailed scores on every testing video in Tab. 6 and Tab. 7. The UTT-DFDETR denotes the model trained with deformable detection head, and all the objects are detected using the detection head. The UTT-FairMOT uses the detected boxes from FairMOT [55], only track transformer is used in this model to track detected object in previous reference frames.

D. Limitations

We propose a unified transformer tracker to tracking objects in different scenarios. Ideally, the tracker is required to track objects over 30 FPS. Currently, our tracker is not able to do the online tracking on the MOT task. Another limitation is that our tracker in the present paper is only trained with pedestrian in the MOT task. However, objects of various categories should be considered (such the vehicles) although this part is more related to the detection head. We mainly focus on the unified transformer tracker in the current manuscript.

E. Code

We will release our code to the public later. Our implementation is mainly based on Detectron2¹ and Pytracking². Our implementation supports distributed training and testing. Previous methods (especially the Siamese pipelines) can be easily reproduced with our code.

¹<https://github.com/facebookresearch/detectron2>

²<https://github.com/visionml/pytracking>