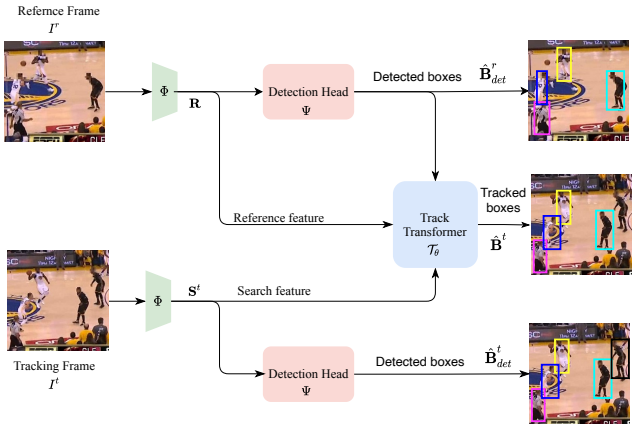

Figure 5. UTT with the detection head.

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 [58] as the detection head Ψ. For the MOT training, the backbone Φ, detection head Ψ, and the track transformer $\mathcal{T}_\theta$ are trained together.

During tracking, we first detect all the objects $\hat{\mathbf{B}}^r_{det}$ in the reference frame I^t. The detected boxes $\hat{\mathbf{B}}^r_{det}$ are then used as target proposals for the tracking frame. We feed the tracking frame feature, target proposals $\hat{\mathbf{B}}^r_{det}$, and the reference feature to the track transformer and produce the tracked boxes $\hat{\mathbf{B}}^t$. Moreover, the detection head also produces the object detection results $\hat{\mathbf{B}}^t_{det}$ in the tracking frame. By calculating the IoU between the tracked boxes $\hat{\mathbf{B}}^t$ and the detected boxes $\hat{\mathbf{B}}^t_{det}$, 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 $\mathcal{T}_\theta$.

Algorithm 1 Unified Training of UTT on both SOT and MOT

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

Algorithm 2 SOT Iteration Alg_{sot}

Input: $\mathcal{D}_{sot}$, backbone Φ and tracker transformer $\mathcal{T}_\theta$
Sampling $(I^r, I^t, \mathbf{B}^r, \mathbf{B}^t)$ from $\mathcal{D}_{sot}$
Predicting target localization in reference frames I^r: $\hat{\mathbf{B}}^r = \mathcal{T}_\theta(\Phi(I)^r, \Phi(I)^r, \mathbf{B}^r) = \{\hat{\mathbf{B}}^r_i\}^L_{i=0}$
Predicting target localization in tracking frames I^t: $\hat{\mathbf{B}}^t = \mathcal{T}_\theta(\Phi(I)^t, \Phi(I)^r, \mathbf{B}^r) = \{\hat{\mathbf{B}}^t_i\}^L_{i=0}$
Calculating SOT loss via Eq. (13):

$$\mathcal{L}^{box}_{SOT} = \sum_{j\in\{t,r\}} \sum_{i=0}^{L} \lambda_G \mathcal{L}_{IoU}(\hat{\mathbf{B}}^j_i, \mathbf{B}^j) + \lambda_1 \mathcal{L}_1(\hat{\mathbf{B}}^j_i, \mathbf{B}^j)$$

Updating Φ and $\mathcal{T}_\theta$

Algorithm 3 MOT Iteration Alg_{mot}

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

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

Calculating the MOT loss: $\mathcal{L}_{MOT} = \mathcal{L}^{box}_{MOT} + \mathcal{L}^{det}_{MOT}$
Updating Φ, Ψ, and $\mathcal{T}_\theta$

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 $\hat{\mathbf{B}}^t_0$ 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 [58] 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↑	MOTP↑	IDF1↑	IDP↑	IDR↑	TP↑	FP↓	FN↓	Rcll↑	Prcn↑	MTR↑	PTR	MLR↓	MT↑	PT↓	ML↓	IDSW↓	FAR↓	FM↓
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 [52].

seq	MOTA↑	MOTP↑	IDF1↑	IDP↑	IDR↑	TP↑	FP↓	FN↓	Rcll↑	Prcn↑	MTR↑	PTR	MLR↓	MT↑	PT↓	ML↓	IDSW↓	FAR↓	FM↓
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 Guassian 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 $\mathcal{T}_\theta$ 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 denots the model trained with deformable detection head, and all the objects are dected using the detection head. The UTT-FairMOT uses the detected boxes from FairMOT [52], 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 Detectron2[1] and Pytracking[2]. 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