Compensation Tracker: Reprocessing Lost Object for Multi-Object Tracking

Zhibo Zou* Junjie Huang* Ping Luo
School of Automation, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
{s190331072, s190331071}@stu.cqupt.edu.cn
{luoping}@cqupt.edu.cn

Abstract

Tracking by detection paradigm is one of the most popular object tracking methods. However, it is very dependent on the performance of the detector. When the detector has a behavior of missing detection, the tracking result will be directly affected. In this paper, we analyze the phenomenon of the lost tracking object in real-time tracking model on MOT2020 dataset. Based on simple and traditional methods, we propose a compensation tracker to further alleviate the lost tracking problem caused by missing detection. It consists of a motion compensation module and an object selection module. The proposed method not only can re-track missing tracking objects from lost objects, but also does not require additional networks so as to maintain speed-accuracy trade-off of the real-time model. Our method only needs to be embedded into the tracker to work without re-training the network. Experiments show that the compensation tracker can efficaciously improve the performance of the model and reduce identity switches. With limited costs, the compensation tracker successfully enhances the baseline tracking performance by a large margin and reaches 66% of MOTA and 67% of IDF1 on MOT2020 dataset.

1. Introduction

Currently, multi-object tracking is one of the basic tasks of computer vision with wide applications in many fields, for example, intelligent security, automatic driving, pedestrian tracking, intelligent monitoring and so on [13]. Tracking by detection (TBD) framework is the current mainstream tracking paradigm.

In order to improve tracking performance, existing researches usually use multiple sub-networks for object information extraction and data association. Such methods generally use networks or modules with information sharing function such as RNN [32], LSTM [24] and Siamese network [39, 12] to save information of tracked objects and complete spatio-temporal matching. In addition, the use of various object information, for example motion information, spatio-temporal information, appearance information and affinity information [48, 20, 47, 10, 41], will complicate the information extraction network. Also, these methods will increase additional computation cost and do harm to the inference speed. Many researchers notice that the speed of tracking model is as important as the tracking performance. For the real-time model, the structure of the detection network cannot be too complicated. Simultaneously, methods of data association should be simple and effective. So, real-time model usually uses detector with speed-accuracy trade-off, for instance, RetinaNet [29], CenterNet [54, 56, 57, 52], Yolov3 [36, 45, 44], Faster RCNN [1] and so on. On the other hand, a series of simple and fast data association methods are also proposed such as bounding box (BBox) intersection ratio (IoU) matching [5, 6, 3], motion prediction [22, 20] and appearance matching [50, 54, 49, 37]. This kinds of information can be obtained by traditional methods or can be learned together with detection task. Therefore, these easily available object characteristics are often used for data association in the real-time model.

In the existing real-time model, both end-to-end model and non-end-to-end model are essentially TBD paradigm. In other words, the performance of the detector plays a decisive role in the tracking result. It's seen in Fig.1. When an object can be tracked in the past frame but cannot be detected in a certain frame due to the unstable performance of the detector (Yellow Arrow Object), the object will not be able to be matched by data association and is considered as an unmatched tracked object. To this end, the tracking result will not include this object and mark them as lost object. This phenomenon will lead to missing tracking and cause ID switches of the missing tracking object so as to impair the tracking performance.

The above discussion raises some questions: *Is the detector stable enough to avoid missing detection? Are these missing tracking objects not in the tracking area or*

*These authors contribute equally.

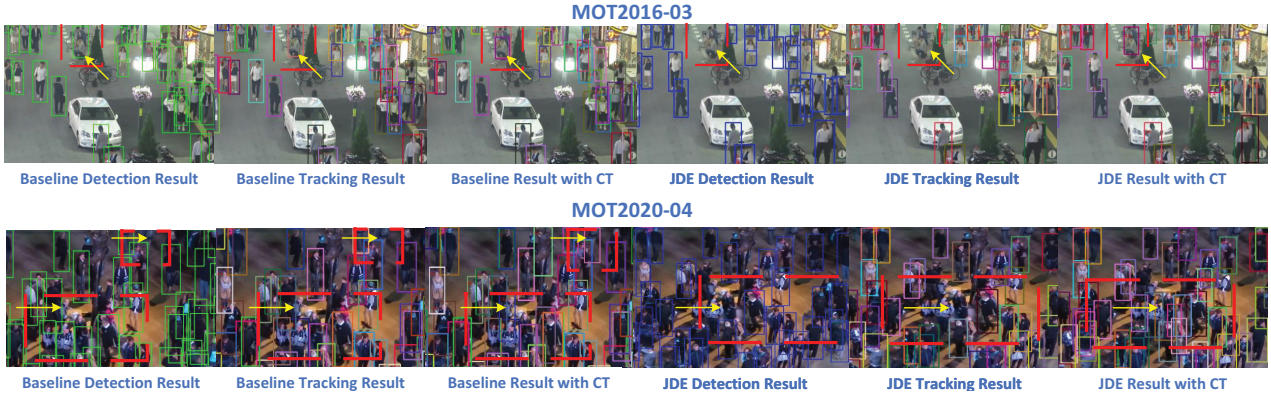

Figure 1. Comparison diagram of model detection result, model tracking result and model with compensation tracker result. The red dashed boxes is the missing detection area, their tracking result area and compensation area of two models. After motion compensation and object selection, the compensation tracker is able to compensate for some previously tracked objects with high reliability (Yellow Arrow Objects).

invisible? The answer is negative! In the existing researches, additional neural networks such as Siamese networks [10, 48, 9] and RNN [18, 47, 38] are often used in the model to re-track lost objects. However, these methods will increase a lot of computation costs and are not friendly to real-time tracking. Using lightweight networks to re-track lost objects will not perform well under complicated scenes [8]. We hope to maintain the speed-accuracy trade-off and re-track the missing tracking objects from the lost objects with computation costs as little as possible. We believe that the tracker not only can exploit the information provided by the detector to perform data association, but also can use the past information to predict positon and compensate for missing tracking objects. Based on this motivation, we propose compensation tracker (CT) based on simple and effective methods to reprocess the lost objects. It contains a motion compensation module and an object selection module. CT screens out the highly credible missing tracking objects from lost objects and re-tracks them without adding abundant computation consumption. As can be seen in Fig.1. The proposed method can effectively alleviate the problem of missing tracking caused by the instability of the detector. Especially in dense crowd regions, the proposed method can greatly improve the tracking performance of the model and reduce ID switches without re-training the network. The contributions of this paper are summarized as follows:

(1) Indicating and analyzing the lost tracking phenomenon of each sub-dataset on MOT2020 dataset.

(2) We propose a compensation tracker based on motion compensation and object selection. By performing motion prediction and reasonable features judgment for the lost objects, CT selects out missing tracking objects with high reliability and re-tracks them in the result.

(3) Extensive experiments show that the reasonable use of traditional methods not only can achieve outstanding compensation result, but also only limited computation costs is produced. Particularly in the dense crowd areas, our method has prominent improvements for the tracking performance of the model.

2. Related Works

Towards Real-Time Tracking. According to the association way of detection information, TBD paradigm can be divided into end-to-end tracking model and non-end-to-end tracking model. POI [50] obtains appearance features through GoogLeNet [43] and combines offline and online tracking methods for object similarity association. Similarly, Sort [3] utilizes IoU correlation between the predicted BBox and the detected BBox to accomplish data association. IoU Tracker [5] only employs the IoU between tracked BBox and the detected BBox for data association. Although its tracking speed is very fast, the tracking performance is not accurate enough. Based on Sort, Deep-Sort [45] treats the detection result as the tracking benchmark and uses a Re-Identification network (Re-ID) to acquire appearance information for further data matching. It also proposes a cascade matching method and combines with the Hungarian [25] algorithm for appearance features matching. Subsequently, MOTDT [8] uses a lightweight network (SqueezeNet[21]) to estimate the trajectory scores of each lost object and makes use of the score map to re-track the missing tracking objects from lost objects. JDE [44] simultaneously learns detection information and appearance embedding information in Yolov3 [36]. Some researches [56, 1, 29, 34] also use other detectors or detection optimization methods to improve tracking performance. Based on the JDE, FairMOTV1 [52] utilizes CenterNet [57] to learn detection information and appearance information. It exploits the advantages of heatmaps detection to obtain object information that is more friendly to data association. FairMOVT2 [54] optimizes the training methods and corrects the ground truth (GT) labels on the basis of FairMOTV1. We use FairMOTV1 (FairV1) as the baseline and analyze the phenomenon of lost object on it.

Model	FairV1		JDE	
Objects / Sets	Lost Objects Number	Compensation Objects Number	Lost Objects Number	Compensation Objects Number
MOT20-01	5016	1559	5888	1832
MOT20-02	34434	11361	41949	15147
MOT20-03	86814	32531	72649	30985
MOT20-05	255953	96167	223091	90558

Table 1. The Statistical number of lost objects and compensation objects in FairV1 and JDE on MOT2020 training dataset.

In the sequels, the pipeline end-to-end methods [42, 33] associate tracked objects in spatio-temporal sequences in adjacent frame. The author designs a pipeline IoU learning method and a pipeline IoU matching method for object association. Chained-Tracker [35] considers that the BBox of adjacent frame are the paired-matching relationship. The author designs a paired boxes loss and corresponding matching method for data association. CenterTrack [56] provides a more concise tracking framework. It directly exploits the heatmap offsets direction of adjacent frame for data association. Quasi-Dense [34] makes use of Bi-Softmax and Quasi-Dense similarity learning to improve the robustness of Re-ID features. However, whether it is end-to-end or non-end-to-end tracking paradigm, they excessively rely on the detection quality and emerge from the phenomenon of missing tracking.

Towards Lost Object Tracking. Using a variety of object information to improve matching accuracy is one of the solutions to avoid lost tracking. Therefore, many researchers prone to use RNN, LSTM [18, 59, 38, 24] or other methods with information preservation and decision-making to optimize the data association of multiple object information. Besides, employing Siamese network [12, 10, 11] for each detected object is also another network solution. Some researchers utilize additional attention networks to extract more powerful appearance features and solve the ambiguous featue problems caused by frequent object interaction and insufficient affinity [48, 59]. In recent years, graph neural networks (GNN) have been gradually used for object tracking. Some researches [58, 7, 30, 27] exploit the powerful information transmission capability and graph matching ability of GNN for data matching so as to improve the tracking performance. Multi-hypothesis tracking [4] generates real trajectory and hypothesis trajectory by Kalman filtering [22]. Through clustering and pruning, the algorithm controls the number of hypothetical trajectories and evaluates the credibility of hypothetical trajectories as real trajectories by log-likelihood estimation scores [4, 23]. This method also depends on the performance of the detector and makes insufficient use of the input image features. Based on IoU Tracker, V-IoU [6] processes the lost objects with a visual single object tracker to enhance the continuity of the trajectory. But it depends on the performance of the filter and it will fail to track the missing tracking object under the condition of object deformation and camera motion. Our method only needs to be embedded into the tracker and does not require extra networks. When the object cannot be matched by the basic tracker, CT firstly performs motion compensation to re-predict its position. Then the motion compensation information and hand-crafted appearance information of the object are used to determine whether the object is credible or not. Our method not only inherits the advantages of V-IoU, but also takes the advantages of appearance information to improve the accuracy of object selection.

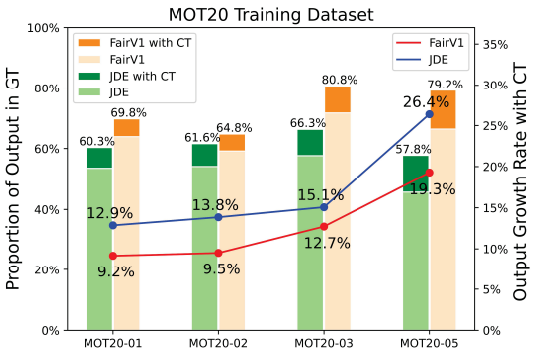

Figure 2. Statistical result of MOT2020 training dataset. The histogram shows the percentage between the tracking result of each sub-dataset and GT in two models. The line chart shows the tracking result growth rate of two models after embedding CT.

3. Analysis of Lost Object

Experiments Setting. All trained network parameters in experiments are from the corresponding authors. The model parameters used for statistic and analysis have not been trained by MOT2020 training dataset. We use DLA34 [51] with DCNv2 [60] in FairV1 and JDE-1088x608 in JDE for statistical experiments. All hyper-parameters will be consistent and the same as authors setting.

Analysis on MOT2020 Result. MOT2020 dataset is a dense crowd sequences with an average of 246 pedestrians in each frame [15]. These sequences include indoor/outdoor and day/night scenes. The sequences with dense crowds and insufficient illumination are a great challenge for detection stability and tracking performance. As can be seen in Fig.2. The tracking quantity of two models (FairV1 & JDE) accounts for about 60% in GT. These results indicate that the generalization ability of two models are not good enough in dense regions. There is a high probability in lost tracking due to the poor performance of detector. As can be seen

in Table 1, the two models have lost objects in each subdataset. In lost objects set, there are some missing tracking objects caused by missing detection. We believe that the tracker can make full use of the existing tracked information to predict position and compensate the output for missing tracking objects. Based on this idea, we use the motion compensation module and the object selection module in CT instead of CNN or GNN. The motion compensation module is used to re-predict the position of the lost object while the object selection module is responsible for feature selection and feature matching. By position judgment and hand-crafted feature matching, CT selects out the highly reliable objects (missing tracking objects) from lost objects and compensates output sequences for them (compesation objects). As can be seen in Fig.2. To this end, FairV1 can be added the tracking sequences by an average of 12.7% and JDE can be increased in the tracking sequences by an average of 17%. This result proves that our compensation objects is effective and credible. To sum up, CT succeeds in improving the tracking performance of the model in dense crowd areas.

4. Methods

4.1. Design of Compensation Tracker

In this section, we will introduce the compensation tracker in detail. It contains three main tracking modules: basic tracker, motion compensation (MC) and object selection (OS). The tracking pipeline can be seen in Fig.3 (K and J represent the object number in the corresponding set).

Basic Tracker. Given an image of frame t, the detected object set D_t can be obtained by network recognition. At frame $t-1$, we can get successfully tracked object set T_{t-1} and lost tracking object set L_{t-1} from the previous frame. Then, the basic tracker performs cascade matching including IoU matching and appearance matching on D_t, L_{t-1} and T_{t-1} [52, 44]. Cascade matching outputs three kinds of objects containing matched tracked objects, newly detected objects (unmatched detected objects) and lost tracking objects (unmatched tracked objects). For the matched tracked objects and the newly detected objects, we update their information and their tracked clipping box images CB_t. Subsequently, the matched tracked objects and the newly detected objects are transmitted in the tracked object set T_t (tracking result). And the lost tracking object set L_t is inputed in CT.

Motion Compensation. Motion compensation module re-predicts the positions for lost tracking object set L_t and outputs the re-predicted results set PB_t. Next, CT implements object selection based on re-predicted results PB_t.

Object Selection. For filtering out the invisible BBox and correcting BBox's size, CT carries out confidence interference filtering, boundary interference filtering and BBox correction for predicted PB_t. Afterwards, IoU interference filtering is used for suppressing the overlap BBox between the predicted PB_t and the tracked object set T_t. Next, we crop the BBox image PCB_t^K according to the predicted BBox. Because the object is tracked in the previous frame, the finally tracked clipping BBox image CB_{t-1}^K is retained. We use PCB_t^K and CB_{t-1}^K for hand-crafted appearance extraction and matching in appearance interference filtering. If an object can be selected out by object selection, we consider that it is the missing tracking object MT_t^K that caused by failing to be detected and we compensate for them in the tracked object set T_t. Otherwise, this lost object that cannot be compensated will be saved in L_t at most 30 frames.

4.2. Motion Compensation

In the motion compensation module, we use the Kalman filter with uniform motion and linear observation by default. Its input can be defined as:

$$Mean = [x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}] \tag{1}$$

where x and y are the horizontal and vertical coordinates of the BBox respectively. a is the ratio between BBox's width and BBox's height. h is the height of the BBox. $\dot{x}, \dot{y}, \dot{a}, \dot{h}$ are the velocities of the corresponding components. $[x, y, a, h]$ are directly observed as object states.

Then, taking $Mean_{t-1}$ as input and calculating the error covariance matrix between the calculation value at frame t and the real value at frame $t-1$:

$$Mean_t^{'} = F_t Mean_{t-1} + A_t X_t \tag{2}$$

$$Cova_t^{'} = F_t Cova_{t-1} {F_t}^{\mathbb{T}} + Q \tag{3}$$

where $Mean_t^{'}$ is the estimation value of the system state at frame t. $Mean_{t-1}$ is the real value of the system state at frame $t-1$. F_t is the motion transformation matrix from the previous state to the current state. A_t is the control martix. X_t is the control variable. $Cova_t{}^{'}$ is the covariance matrix of the error between the calculation value and the real value. $Cova_{t-1}$ is the covariance matrix of the error between the estimation value and the real value. $\mathbb{T}$ is the transpose operator. Q is the multi-variate normal distribution of covariance matrix. Next, Kalman gain is calculated:

$$K_t = Cova_t{}^{'} {H_t}^{\mathbb{T}} (H_t Cova_t{}^{'} {H_t}^{\mathbb{T}} + R)^{-1} \tag{4}$$

$$Mean_t = Mean_t^{'} + K_t(Z_t - H_t Mean_t^{'}) \tag{5}$$

where K_t is the Kalman gain. Z_t is the system measurement value. H_t is the measurement transfer matrix and R is covariance matrix of observation noise [45, 22].

Finally, the error covariance matrix between the estimation value and the real value is updated:

$$Cova_t = (1 - K_t \cdot H_t) Cova_t^{'} \tag{6}$$

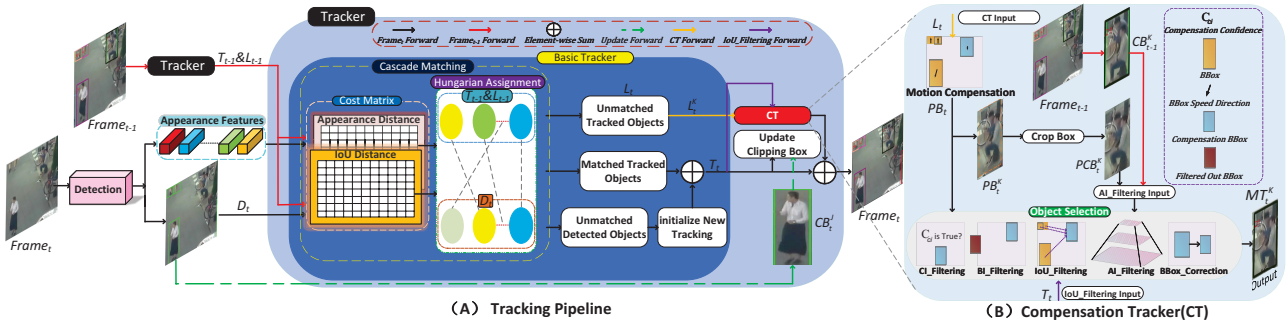

Figure 3. Online tracking pipeline of compensation tracker. The above figure shows the overall data flow and the relationship between the basic tracker and the compensation tracker. Actually, CT is the extension of the basic tracker. When an object is tracked at frame $t-1$ and is not detected at frame t, the object cannot be matched by cascade matching and is considered as lost object L_t^K. Then, L_t^K is transmitted in the compensation tracker for motion compensation and object selection. Meanwhile, tracked object J will be updated their information. After processing the selection result and the association result, the tracker outputs the final tracking result for object tracking.

4.3. Object Selection

Experiments show that only using the motion compensation module to unconditionally compensate for the lost objects will result in wrong compensation. In this section, we will further introduce how to screen the BBox predicted by motion compensation to find out the missing tracking object and compensate for them.

Confidence Interference Filtering(CI_Filtering). Error bounding box (EBBox) is generated by the fact that the tracked object has been lost or the object has not been detected in so many frames, but the motion compensation is still carried out. Referring to the maximum reserving frame value of the lost object in the baseline, we consider it as the maximum of compensation frame value and employ the compensation confidence threshold to avoid the generation of EBBox. The compensation confidence is defined as follow:

$$\mathcal{C}_{ci} = \mathbb{I}\{S_{ts} - L_{ts} > 0\} \qquad s.t. S_{ts} > C_F \tag{7}$$

where $\mathcal{C}_{ci}$ is the compensation confidence. S_{ts} is the number of times that the object is successfully tracked. L_{ts} is the number of times that the object is lost in tracking and C_F is the compensation frame value. When a lost object does not meet the above formula, the object will be filtered out. It's seen in Fig.4.

Boundary Interference Filtering(BI_Filtering). When the tracked scene moves relatively fast, only using the confidence interference filtering will not achieve an optimal compensation result. When a lost object moves away from the tracking area, we need to judge the center position of its predicted BBox by the following formula:

$$\mathcal{C}_{bi} = \mathbb{I}\{x - x_w * \alpha > 0 \quad \wedge \quad w - x - x_w * \alpha > 0\} \tag{8}$$

where $\mathcal{C}_{bi}$ is the confidence of BI_Filtering. x is the center point abscissa of the predicted BBox. x_w is the width of the predicted BBox. α is a boundary weight. w is the width of the image. When a predicted BBox of lost object does not satisfy formula (8), the object will be filtered out. As it's seen in Fig.5.

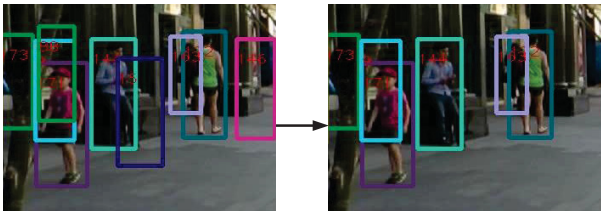

Figure 4. Diagram of confidence interference filtering result. After using the confidence interference filtering, the BBox (ID 15, ID 98 and ID 146) that does not meet the formula (7) will be filtered out and the others are retained because they satisfy the condition.

Figure 5. Diagram of boundary interference filtering result. The width and height of the image are 640 and 480 respectively. The center coordinate of the lost object 129 is (596, 242.3) and its width x_w is 206.9. Because the object 129 does not satisfy the formula (8), it will be filtered out.

IoU Interference Filtering(IoU_Filtering). In order to improve the compensation result, we also eliminate the BBox with object occlusion and object overlap to further prevent wrong compensation. The filtering effect can be seen in Fig.6. We compare the predicted compensation

BBox set PB_t with the tracked object set T_t containing their area ratio, IoU and BBox embedding degree.

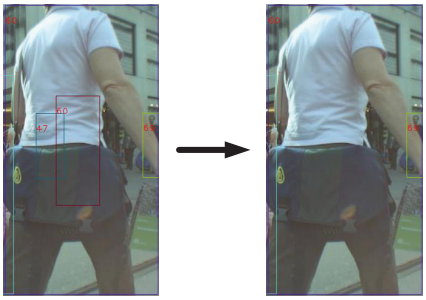

Figure 6. Diagram of IoU interference filtering result. It is obvious from the figure that the wrong BBox has been removed.

Appearance Interference Filtering(AI_Filtering). IoU Filtering cannot solve the problem of occlusion by other objects and severe drift of BBox. So, we use the position information predicted by motion compensation to extract the clipping box image PCB_t^K at the current frame. Afterwards, the hand-crafted feature similarity between PCB_t^K and the finally tracked clopping box image CB_{t-1}^K will be evaluated. Firstly, the two clipping BBox images will be filtered by Gaussian kernel and down-sampled twice respectively. Then, the Gaussian difference calculation between them and themselves will be performed [28]:

$$D(p_x, p_y, \sigma) = L(p_x, p_y, S_f\sigma) - L(p_x, p_y, \sigma) \quad (9)$$

where $D(\cdot)$ is a Gaussian differential operator. $D(p_x, p_y, \sigma)$ is the Gaussian differential image. p_x and p_y are pixel position of the image. $L(\cdot)$ is the Gassian smoothed image which is produced from the convoluition of a variable-scale Gassian with an input image. σ is the scale space operator of the Gauss kernel function. S_f is the feature pyramid scaling factor. After getting $D(p_x, p_y, \sigma)$ from formula (9), the algorithm finds the extremum in it and inputs the extremum point sets into formula (10) to obtain the modulus and direction of each extremum region [28].

$$\begin{cases} m(p_x, p_y) = ||(L(p_{x+1}, p_y) - L(p_{x-1}, p_y)) + \\ \qquad\qquad (L(p_x, p_{y+1}) - L(p_x, p_{y-1}))||_2 \\ \theta(p_x, p_y) = tan^{-1}(\frac{L(p_x, p_{y+1}) - L(p_x, p_{y-1})}{L(p_{x+1}, p_y) - L(p_{x-1}, p_y)}) \end{cases} \quad (10)$$

where $m(p_x, p_y)$ is the modulus of the extreme points and $\theta(p_x, p_y)$ is the direction of the extreme points.

Set m and set θ are input into the Histogram of Oriented Gradients [14] to generate the 128 dimension vectors set. Finally, K-Nearest-Neighbor (KNN) matching is performed for measuring the Euclidean distance of these vectors [19]. If there are enough matching points, the lost object is treated as the missing tracking object and will be re-tracked. The matching formula is as follows:

$$\mathcal{C}_{ai} = \mathbb{I}\left[\mathcal{K}(\mathcal{H}(m_{t-1}^K, \theta_{t-1}^K), \mathcal{H}(m_t^K, \theta_t^K)) > \sigma_m\right] \quad (11)$$

Algorithm 1: Compensation Tracker

Input:
L_t is the lost object set at frame t
T_t is the tracked object set at frame t
CB_{t-1}^K is the K-th clipping box image at frame $t-1$
PCB_t^K is the K-th clipping box image at frame t
Output:
MT_t is the missing tracking object set at frame t

for $l_K \in L_t$ **do**
  $Mean_t, Cova_t \leftarrow$ MC($l_K.Mean, l_K.Cova$) by(1)(2)(3)(4)(5)(6)
  $Mean_t \leftarrow$ BBox_Correction ($Mean_t$,$l_K.Mean$)
  $\mathcal{C}_{ci} \leftarrow$ CI_Filtering(l_K) by(7)
  $\mathcal{C}_{bi} \leftarrow$ BI_Filtering ($Mean_t$) by(8)
  for $T_t^l \in T_t$ **do**
    $l_K \leftarrow$ IoU_Filtering($Mean_t$,T_t^l)
  end for
  $\mathcal{C}_{ai} \leftarrow$ AI_Filtering (CB_{t-1}^K,PCB_t^K) by(9)(10)(11)
  if $\mathcal{C}_{ci} \wedge \mathcal{C}_{bi} \wedge \mathcal{C}_{ai}$ **then**
    $l_K \leftarrow$ Update_Parameter($Mean_t, Cova_t$)
    $MT_t.append(l_K)$
  end if
end for
return MT_t

where $\mathcal{C}_{ai}$ is the confidence of AI_Filtering. $\mathcal{K}$ is the KNN matching function. $\mathcal{H}$ is the Histogram of Oriented Gradients. m_{t-1}^K and θ_{t-1}^K are the modulus and direction of CB_{t-1}^K respectively. m_t^K and θ_t^K are the modulus and direction of PCB_t^K respectively. σ_m is the threshold for KNN matching.

BBox Correction(BBox_Correction). Since the predicted BBox will have an inaccurate size, the object cannot be marked accurately. The BBox's size in adjacent frame changes very limited. When the change of area between lost BBox and its predicted BBox is greater than 1.1, the compensation BBox will be resized in the original size. Algorithm 1 describes how the proposed method works for the lost objects.

5. Experiments

5.1. Experiment Details

We carry out extensive evaluations on MOT2015[26] dataset, MOT2016[31] dataset, MOT2017[31] dataset and the latest MOT2020[15] dataset. Besides, we achieved relatively good results and metrics precision in all datasets.

Experiments Platform. Our experiments are implemented on Pytorch. The computer used in the experiment is equipped with Xeon Platinum 8163 CPU and

C_F	MOTA↑	IDF1↑	ID.SW↓	σ_m	MOTA↑	IDF1↑	ID.SW↓
10	46.5/76.0	66.0/76.8	284/516	1	45.5/76.1	65.2/76.7	278/508
20	46.2/76.0	66.3/76.9	266/514	3	45.7/76.0	64.9/76.9	284/514
30	46.2/76.0	65.6/77.1	263/518	5	46.0/76.0	65.5/76.8	282/509
40	46.0/76.0	65.8/76.8	269/510	7	46.2/76.0	64.8/76.8	274/506

α	MOTA↑	IDF1↑	ID.SW↓	FP↓	FN↓
0.1	46.1/76.0	65.5/76.8	249/508	18940/7467	4061/18534
0.3	46.2/76.0	65.5/77.0	249/511	18891/7449	4065/18526
0.5	46.2/76.0	65.5/76.9	250/505	18871/7434	4071/18542
0.7	46.2/76.0	65.5/76.8	253/512	18855/7436	4083/18565

Table 2. Hyper-parameters experiments on MOT15/MOT16. We uses C_F=30, σ_m=5 and α=0.5 for all datasets and experiments.

RTX2080Ti. Furthermore, we use the trained model parameters of the DLA network for experiments in baseline. And JDE-1088x608 is employed in JDE in ablation experiments. All the trained model parameters are provided by corresponding author. These networks are trained with the following datasets including MOT2017[31], Caltech[16], CityPersons[53], CuhkSysu[46], PRW[55] and ETH[17].

Evaluation Metrics. CLEAR MOT Metrics [2] and IDF1 are used for measuring tracking result. These metrics include multi-object tracking accuracy (MOTA), ID switches (ID.Sw), the ratio of correctly identified detections over the average number of GT and computed detections (IDF1), multi-object tracking precision (MOTP), the most tracked object (MT), the most lost object (ML), the average number of false alarms per frame (FAF), the total number of times a trajectory is fragmented (Frag), the number of false positives (FP) and the number of false negatives (FN) [13]. And FPS is obtained by running on a single RTX2080Ti.

5.2. Hyper-Parameter Experiments

There are three adjustable parameters in CT. They are the compensation frame value C_F, matching threshold σ_m and boundary weight α respectively. Parameters setting do not require heuristic algorithms and do not need to be adjusted for each dataset. C_F is a hyper-parameter that already exists in baseline and JDE. Considering that it also has an impact on the compensation result, we also discuss its robustness. The sensitivity experiments of three hyperparameters on MOT15 and MOT16 are shown in Table 2. C_F is the maximum pre-stored threshold for lost objects. Too large C_F will result in too many pre-stored lost objects and too many memory costs. This setting will lead to ambiguous object matching and reduce tracking accuracy. Too small C_F will raise the problem that missing tracking objects cannot be re-tracked and its ID switches frequently. σ_m affects the appearance matching precision of missing tracking objects. If σ_m is too large, the objects with high reliability cannot be re-tracked. Besides, α is used to filter out the objects that are out of boundary and deal with the drifting problem of the predicted BBox. It only has weak influences on IDF1, FP and FN. Experiments on MOT15 and MOT16 show that the hyper-parameters in CT are robust.

MOT2020 Test Dataset						
Component	MOTA↑	IDF1↑	MT↑	ML↓	ID.Sw↓	FPS ↑
Baseline	58.7	63.7	66.3%	8.5%	6013	15.2
Baseline+MC	65.0	66.6	59.1%	13.0%	2119	14.6
Baseline+MC+OS	66.0	67.0	56.3%	13.3%	2237	13.5

Table 3. Module ablation experiments on MOT2020.

Models	MOTA↑	IDF1↑	MT↑	ML↓	ID.Sw↓	FPS↑
MOT2016 Test Dataset						
JDE[44]	64.4	55.8	35.4%	20.0%	1544	22.3
JDE with CT	65.0	59.1	36.1%	18.8%	1525	19.7
FairV1[52]	68.7	70.4	39.5%	19.0%	953	21.7
FairV1 with CT	69.8	71.1	42.0%	15.8%	912	19.2
MOT2020 Training Dataset						
JDE*[44]	48.2	32.1	318	497	18631	15.0
JDE with CT	54.4	43.1	526	372	11157	12.3
FairV1[52]*	62.3	47.5	790	288	16395	15.2
FairV1 with CT	65.6	57.5	1030	247	7816	13.5

Table 4. 'Private' model ablation experiments. '*' means that the result is evaluated by ourselves.

5.3. Ablation Experiments

As can be seen in Table 3. After using the motion compensation module, some metrics such as MOTA, IDF1 and ID.Sw have been significantly improved due to compensation for the missing tracking objects. However, because of unconditional compensation, some EBBoxes still appear. After conducting the object selection, MOTA and IDF1 can be further improved while only a little quantity of ID switches are raised. Additionally, the speed costs of motion compensation and object selection are only 0.6 FPS and 1.1 FPS respectively.

As can be seen in Table 4. On MOT2016 test dataset, the result of the JDE with CT is better than itself. CT improves JDE by 3.3% on IDF1, 0.6% on MOTA, 0.7% on MT and 1.2% on ML. This result manifests that our tracker can effectively enhance tracking performance and optimize data association of lost objects. Especially on the 2020 training dataset, the improvement is more prominent and various metrics achieve greater gains. Among this metrics, MOTA increased by 6.2%, IDF1 increased by 11% and ID.Sw decreased by 7474. For the baseline model, various metrics have also been improved on MOT2016 dataset. CT ameliorates baseline by 1.1% on MOTA, 2.5% on MT, 3.2% on ML and 41 on ID.Sw. It can also be seen in Table 4 that there is a significant gains in MOT2020 tracking result. After employing CT, tracking instability problem is further alleviated and various metrics such as MOTA, IDF1 have been improved to a certain extent. More importantly, ID.Sw on MOT2020 surpasses baseline by 8579.

This is because CT alleviates the lost tracking problem caused by the detector instability so that missing tracking objects can be effectively re-tracked. Also, our tracker not only accurately compensates for missing tracking objects, but also lessen unnecessary ID switches. Something is worthy noticeable that time consumption of CT is limited and

the average costs approximate 2-3 frames on two 'Private' real-time models.

For the sake of further demonstrating the data association performance of the CT, Sort, MOTDT and Tracktor ('Public' Tracker) are also used for 'Pubilc' ablation experiments.

As can be seen in Table 5. After using CT, the performance of Sort can be further improved on MOT2020 test dataset. Among these metrics, MOTA increased by 0.6% and MT increased by 0.9%. Especially in ID.Sw and Frag, the reductions are 1499 and 10313 respectively. On MOT2020 training dataset, Sort with CT can obtain 7.1 gains, 18.1 gains to Sort counterparts in MOTA and IDF1. Above all, ID.Sw declines by 9253. It can be seen in Table 5 that the tracking continuity (MT & ML) have also been greatly improved. On the other hand, MOTDT utilizes SqueezeNet [21] to score and re-track lost objects. We replace this network with CT. As it's seen in Table 5. There is a great challenge for the performance of the lightweight network on MOT2020 dataset. However, CT uses historical reliable information and hand-crafted featrue information obtained by traditional methods. The extraction way of traditional methods is stabler than network prediction under the complicated circumstances. Compared with MOTDT, our method has various degrees improvements on MOT2020 dataset. Notably on MOT2020 test dataset, our method achieves 2513 gains in Frag, 211 gains in ID.Sw, 1.4 gains in MOTA, 4.4% gains in MT and 1.1 gains in IDF1. Also, we replace Tracktor's camera motion wtih CT for experiments on MOT2020. Although CT's ID.Sw is higher than camera motion, CT still outperforms Tracktor on MOTA, MT and ML. There are 2.3 gains and 1.2 gains in MOTA of Tracktor wtih CT on MOT2020 training dataset and test dataset respectively. In terms of the performance in 'Public' ablation experiments, CT is effective in data association and has the capability to re-track missing tracking objects.

5.4. Comparison with state-of-art Online Models

MOT2016 & MOT2017. As can be seen in Table 6, the evaluation result of our method on MOT2016 and MOT2017 are outstanding. There are improvements in MOTA (69.8%), IDF1 (71.1%), MT (42%) and ML (15.8%) on MOT2016 test channel. On MOT2017 test channel, MOTA, IDF1, MT and ML are 68.8%, 70.2%, 40.8% and 17.7% respectively. These metrics are higher than other online traking models and reach the state-of-art performance.

MOT2020. The effect of CT is more remarkable on MOT2020 test result. It can be clearly seen in Table 7. CT can reach great gains in MOTA (66%), IDF1 (67%), Frag (4154) and ID.Sw (2237). What's more, CT surpasses FairV1 by 7.3% on MOTA, 3.3% on IDF1 and 14.9% on FAF. Our method outperforms other 'private' models on Frag and ID.Sw. Compared with FairV1, the total number of ID.Sw drops from 6013 to 2237. Based on the performance on the three datasets, our tracker effectively optimizes tracking performance by preventing the performance degradation caused by missing detection.

Models	MOTA↑	IDF1↑	MT↑	ML↓	FAF↓	ID.Sw↓	Frag ↓
MOT2020 Test Dataset							
SORT[3]	42.7	45.1	16.7%	26.2%	6.1	4470	17798
SORT with CT	43.3	45.2	17.6%	26.3%	6.3	2971	7485
MOTDT[8]*	43.7	40.8	14.4%	27.7%	3.4	3705	9225
MOTDT with CT	45.1	41.9	18.8%	27.1%	4.6	3494	6712
Tracktor[1]	52.6	52.6	365	331	/	1648	/
Tracktor with CT	53.8	51.1	386	310	2.0	2456	5046
MOT2020 Training Dataset							
SORT[3]*	45.8	34.1	288	593	/	12992	/
SORT with CT	52.9	52.2	424	417	/	3739	/
MOTDT[8]*	48.5	40.1	393	499	/	7754	/
MOTDT with CT	50.2	40.7	498	499	/	7421	/
Tracktor[1]	66.4	60.7	892	259	/	2664	/
Tracktor wtih CT	68.7	60.6	926	251	/	3654	/

Table 5. 'Public' model ablation experiments. '*' means that the result is evaluated by ourselves.

Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	ID.Sw↓
MOT2016 Test Dataset						
JDE[44]	64.4	55.8	/	35.4%	20.0%	1544
POI[50]	66.1	65.1	79.5	34.0%	20.8%	805
Tube TK POI[33]	66.9	62.2	78.5	39.0%	16.1%	1236
CTracker[35]	67.6	57.2	78.4	32.9%	23.1%	1897
FairV1[52]	68.7	70.4	80.2	39.5%	19.0%	953
QuasiDense[34]	69.8	67.1	79.0	41.6%	19.8%	1097
Ours	69.8	71.1	80.0	42.0%	15.8%	912
MOT2017 Test Dataset						
SST[41]	52.4	49.5	76.9	21.4%	30.7%	8431
Tube_TK[33]	63.0	58.6	78.3	31.2%	19.9%	4137
CTracker[35]	66.6	57.4	78.2	32.2%	24.2%	5529
FairV1[52]	67.5	69.8	80.3	37.7%	20.8%	2868
CTTracker17[56]	67.8	64.7	78.4	34.6%	24.6%	3039
QuasiDense[34]	68.7	66.3	79.0	40.6%	21.9%	3378
Ours	68.8	70.2	80.0	40.8%	17.7%	2805

Table 6. Comparison experiment on MOT2016 and MOT2017.

MOT2020 Test Dataset								
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FAF↓	ID.Sw↓	Frag↓
FairV1[52]	58.7	63.7	77.2	66.3%	8.5%	24.7	6013	8140
TrTrack[40]	64.5	59.2	80.0	49.1%	13.6%	6.4	3565	11383
Ours	66.0	67.0	77.8	56.3%	13.3%	9.8	2237	4154

Table 7. Comparison experiment on MOT2020.

6. Conclusion

In this paper, we point out the shortcomings of tracking by detection model and analyze the phenomenon of lost object in the real-time model in dense crowd area. Considering the computation burden, we propose a simple and effective compensation tracker and name it as CT. The proposed method has the advantage of plug and play without re-training the network. Our method employs tradition methods to re-track lost objects instead of additional networks. Extensive experiments indicate that the proposed method is able to effectively improve the tracking performance of real-time models with limited time consumption.

References

[1] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE international conference on computer vision*, pages 941–951, 2019.

[2] 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.

[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[4] Samuel S Blackman. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerospace and Electronic Systems Magazine*, 19(1):5–18, 2004.

[5] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2017.

[6] Erik Bochinski, Tobias Senst, and Thomas Sikora. Extending iou based multi-object tracking by visual information. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[7] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020.

[8] Long Chen, Haizhou Ai, Zijie Zhuang, and Chong Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2018.

[9] Peng Chu, Heng Fan, Chiu C Tan, and Haibin Ling. Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 161–170. IEEE, 2019.

[10] Peng Chu and Haibin Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6172–6181, 2019.

[11] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4836–4845, 2017.

[12] Qi Chu, Wanli Ouyang, Bin Liu, Feng Zhu, and Nenghai Yu. Dasot: A unified framework integrating data association and single object tracking for online multi-object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10672–10679, 2020.

[13] Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, and Francisco Herrera. Deep learning in video multi-object tracking: A survey. *Neurocomputing*, 381:61–88, 2020.

[14] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[15] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020.

[16] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: A benchmark. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 304–311. IEEE, 2009.

[17] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A mobile vision system for robust multi-person tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[18] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475. IEEE, 2018.

[19] Keinosuke Fukunaga and Patrenahalli M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE transactions on computers*, 100(7):750–753, 1975.

[20] Piao Huang, Shoudong Han, Jun Zhao, Donghaisheng Liu, Hongwei Wang, En Yu, and Alex ChiChung Kot. Refinements in motion and appearance for online multi-object tracking. *arXiv preprint arXiv:2003.07177*, 2020.

[21] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[22] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[23] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 4696–4704, 2015.

[24] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 200–215, 2018.

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

[26] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015.

[27] Jiahe Li, Xu Gao, and Tingting Jiang. Graph networks for multiple object tracking. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 719–728, 2020.

[28] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[29] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14668–14678, 2020.

[30] Cong Ma, Yuan Li, Fan Yang, Ziwei Zhang, Yueqing Zhuang, Huizhu Jia, and Xiaodong Xie. Deep association: End-to-end graph-based learning for multiple object tracking with conv-graph neural network. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pages 253–261, 2019.

[31] 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.

[32] Anton Milan, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[33] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6308–6318, 2020.

[34] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. *arXiv preprint arXiv:2006.06664*, 2020.

[35] Jinlong Peng, Changan 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 *European Conference on Computer Vision*, pages 145–161. Springer, 2020.

[36] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[37] Ergys Ristani and Carlo Tomasi. Features for multi-target multi-camera tracking and re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6036–6046, 2018.

[38] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 300–311, 2017.

[39] Bing Shuai, Andrew G Berneshawi, Davide Modolo, and Joseph Tighe. Multi-object tracking with siamese track-rcnn. *arXiv preprint arXiv:2004.07786*, 2020.

[40] Peize Sun, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple-object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020.

[41] ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal S Mian, and Mubarak Shah. Deep affinity network for multiple object tracking. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[42] ShiJie Sun, Naveed Akhtar, XiangYu Song, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Simultaneous detection and tracking with motion modelling for multiple object tracking. In *European Conference on Computer Vision*, pages 626–643. Springer, 2020.

[43] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[44] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2(3):4, 2019.

[45] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[46] Tong Xiao, Shuang Li, Bochao Wang, Liang Lin, and Xiaogang Wang. Joint detection and identification feature learning for person search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3415–3424, 2017.

[47] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3988–3998, 2019.

[48] Junbo Yin, Wenguan Wang, Qinghao Meng, Ruigang Yang, and Jianbing Shen. A unified object motion and affinity model for online multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6768–6777, 2020.

[49] Young-Chul Yoon, Du Yong Kim, Young-min Song, Kwangjin Yoon, and Moongu Jeon. Online multiple pedestrians tracking using deep temporal appearance matching association. *Information Sciences*, 561:326–351, 2021.

[50] 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 *European Conference on Computer Vision*, pages 36–42. Springer, 2016.

[51] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018.

[52] Yifu Zhan, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*, 2020.

[53] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. Citypersons: A diverse dataset for pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 2017.

[54] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *arXiv preprint arXiv:2004.01888*, 2020.

[55] Liang Zheng, Hengheng Zhang, Shaoyan Sun, Manmohan Chandraker, Yi Yang, and Qi Tian. Person re-identification

in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1367–1376, 2017.

[56] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020.

[57] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[58] Zongwei Zhou, Junliang Xing, Mengdan Zhang, and Weiming Hu. Online multi-target tracking with tensor-based high-order graph matching. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1809–1814. IEEE, 2018.

[59] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018.

[60] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019.