Compensation Tracker: Reprocessing for Lost Object

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

*At present, the main research direction of multi-object tracking framework is tracking by detection. Although the detection-based tracking framework can achieve good results, it is very dependent on the performance of the detector. The tracking results will be affected to a certain extent when the detector has the behaviors of omission and error detection. Therefore, in order to solve the problem of missing detection, we designs a compensation tracker based on motion compensation and objects selection. Besides the tracker can be embedded into other non-end-to-end tracking frameworks. Experiments show that after using the compensation tracker designed in this paper, evaluation indicators have improved in varying degrees on MOT Challenge datasets. With limit cost, the compensation tracker haves reached **66% MOTA and 67% IDF1** in the 2020 datasets of dense scenarios. This shows that the proposed method can effectively improve the tracking performance of the model.*

1. Introduction

Currently, multi-object tracking has used in many scenarios. For example, intelligent security, automatic driving, pedestrian tracking, intelligent monitoring and so on. In the network framework, detection-based tracking is the mainstream multi-object tracking model.

Detection-based tracking can be traced back to Deep-Sort [37], which used the detection results of YOLOv3 [30] as a tracking benchmark, and introduced a Re-ID model specifically for extracting appearance information as a further matching optimization.Moreover, it uses cascading matching and Hungarian algorithm [17]to match and process multiple unmatched object. This provides a tracking framework for future development. Subsequently,
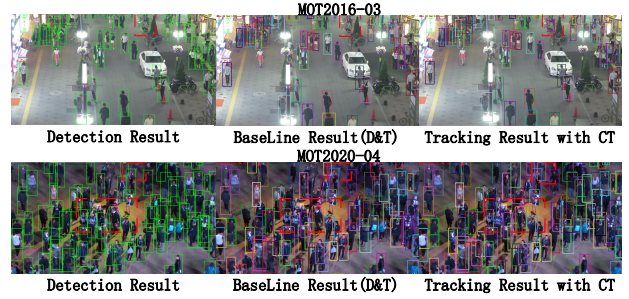

Figure 1. Comparison of test results, baseline tracking results, and compensation result. The red dashed boxes are the detector omission areas, baseline tracking results, and compensation results of the compensation tracker (left to right). After motion compensation and objects selection, our compensation tracker can compensate some previously tracked objects with high reliability.

MOTDT [20] carries out tracking optimization processing on this basis. It uses the network to generate a score maps and scored the trajectory of each tracked object. With the development of the detection field, the tracking field has been greatly influenced. JDE [36] tracker, while following the previous tracking method, combines the Re-ID model with the detection network to form a one-stage tracker. Simultaneously, it uses the new loss function for appearance learning and utilizes the automatic balance loss function to solve the multi-task learning problem of multi-object tracking. Due to the update and development of detectors, the increase of detection accuracy has great influence on tracking. In [43, 46, 1, 22], they all apply new detectors or detection optimization methods to improve tracking performance. Of course, there are also other authors who innovate at other levels. IoU-tracker [4] only uses the boundary boxes intersection over union (IoU) of adjacent frames to track object. Although it achieves good results in speed, it is not accurate enough.

Detection-based tracking models can be divided into end-to-end and non-end-to-end. These end-to-end meth-

---

*The first two authors contributed equally to this work
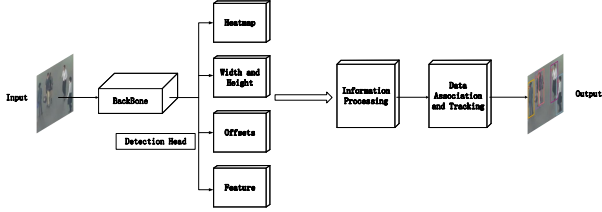
†Corresponding author

Figure 2. Non-End-to-End One-Stage detection and tracking network architecture.

ods such as [28, 29, 23, 19] use pipeline end-to-end tracking architecture, chain tracking architecture, and graph convolutional network architecture. This type of models usually completes the detection and tracking tasks in the network without introducing other object matching methods and trackers at the back end of the network. Those non-end-to-end tracker such as[36, 43] usually takes the detection result of the network as the input of the back-end tracker and then uses a series of prediction and matching methods to allocate, initialize tracking and move the object. Our method is designed and optimized on the backend tracker of non-end-to-end frame work.

In addition, there are many methods of object tracking. [26, 12, 32] use networks or modules with memory functions such as RNN and LSTM to obtain long-term and short-term information of tracking object to prevent object loss and identity switching. [40, 15, 39], etc., integrate and utilize object features such as motion model, spatiotemporal model, and appearance model for extraction, prediction and correlation, thereby improving object tracking performance and alleviating the interference of similar object to tracking. These tracking methods are long-term tracking of the object without considering the issue of whether the lost object should also be tracked. In the detection-based tracking model, these methods will slow down the inference speed. In a non-end-to-end model, these methods are not necessarily applicable.

For the detection-based tracking model, the detector plays a decisive role in the comprehensive performance of multi-object tracking. However, if the detector does not detect the object on a certain frame, but the object is actually present, it will directly lose the object. We believe that the tracker can not only use the information provided by the detector to match, but also use the past information to predict and compensate the missing object. Therefore, we design a compensation tracker to solve this problem. It comprises a motion compensation module and object selection module. The effect of our compensation tracker is shown in Figure 1.

2. Relate Works

Detection Method based on Anchor-Free. Anchor-based detection method[31, 5, 34, 14] samples fixed shape bounding boxes around low-resolution images and classified each bounding box as' foreground 'or' background '. At the same time, NMS and other methods are used to filter the bounding box, which will increase computation. In this paper, Anchor-Free detection method is adopted, which does not need the above complex operation. It uses heatmaps to extract local peaks [6, 27] and predicts the object center point, so as to predicts the object boundary based on the center point. This key points prediction method can greatly reduce the computation and object ID switch [47].

One-Stage Detection Model. Previous tracking models usually treat object detection and Re-ID as two separate tasks. In [49, 41, 24, 48], all interested objects in the graph are firstly determined by the detector of convolutional neural network, and then the image is cut out according to the boundary box and fed into the identity embedding network to extract the re-id features. Then, the bounding box is linked into multiple tracks [43, 30]. These operations can increase the network computation and complexity and are not conducive to real-time tracking. In this paper, object detection and embedding functions are completed simultaneously in a single network and the model is non-end-to-end tracking method, as shown in Figure 2.

Compensation Tracker. At the present, the tracker is generally a sequential task, which calculates the cost matrix according to Re-ID and bounding box information and then uses Kalman filter [16] and Hungarian algorithm [17] to complete the prediction and assignment tasks. Then, the tracked objects that has not been matched in the current frame is considered to be lost objects and no processe is performed for them. Our method is to further predict and process these lost objects after the existing non-end-to-end tracking and matching process. In brief, we use the bounding box (BBox) of the object, the final tracked object appearance map and the predicted bounding box to determine whether the target still exists in the tracking field and is visible. If the target meets the conditions, the compensated result is finally output together with the previously matched result, as shown in Figure.4.

3. Methods

3.1. Baseline Network Model

We use FairV1 (Non-End-to-End Tracking-by-Detection)[43] as the baseline framework to experiment. In this paper, we only use deep aggregation network (DLA)[42] as the backbone network. Moreover, the deformable convolution [49] (DCNv2) is applied to DLA network to expand the receptive field of the network and improve the detection accuracy. Compared with the

original DLA network, the DLA network with deformable convolution has more jumper layers, which can increase the receptive field and enhance the modeling ability [49], as shown in Fig. 3.

3.2. Design of Compensation Tracker

In this section, we will introduce the compensation tracker in detail. It contains two main tracking modules: motion compensation (MC) and objects selection (OS). The tracking flow can be seen in Figure 4. Given the input picture of frame T,after neural network recognition, the detected objects set D_t can be obtained. For frame $T-1$, we can get the objects set T_{t-1} with successful tracking and the objects set L_{t-1} with missing tracking from the previous frame. Then, the tracker performs cascade matching on D_t,L_{t-1} and T_{t-1} [3]. For the objects matched by the current frame and the newly detected objects, we update the objects information and the clipping box CB to which they can be tracked and output the result to the tracked set T_t.For the L_t in the current frame, we first perform motion compensation to predict the position of the bounding box in the current frame and crop the bounding box image CB_t^K Since the object is tracked in the previous frame, the clipping bounding box image CB_{t-1}^K of the previous frame is retained. Therefore, we input the two cropped images together into the object selection module. If the object can be selected, we consider that it belongs to the target that the detector missed. Next, the object is output to the tracking result together with T_t. Otherwise, the object will be saved in L_t at most 30 frames.

3.3. Motion Compensation

In the motion compensation of the tracker, we use the Kalman filter[16] with uniform motion and linear observation by default. Its' input can be defined as:

$$X = [x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}] \quad (1)$$

Where x and y are the horizontal and vertical coordinates of the BBox, respectively; a is the ratio of the width and height of the BBox;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.

Take the above information as input information and calculate the error covariance matrix between the calculated value and the real value at $k-1$ frame:

$$Mean_k^{'} = F_k Mean_{k-1} + A_k X_k \quad (2)$$

$$Conva_k^{'} = F_k Conva_{k-1} F_k^T + Q \quad (3)$$

where $Mean_k^{'}$is the estimated value of the system state at frame k; $Mean_{k-1}$ is the real value of the system state at frame $k-1$; $Conva_k^{'}$ is the convariance matrix of the
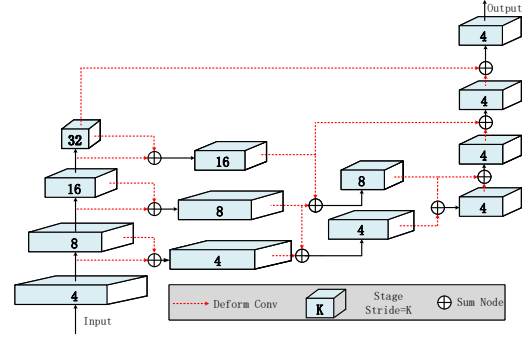

Figure 3. Backbone network structure(DLA-DCNv2-34)

error between the calculated and the real value. $Conva_{k-1}$ is the covariance matrix of the error between the estimated and the real value; F_k is the motion transformation matrix; A_k is the control parameter matrix; X_k is the control quantity. Q is the multi-variate normal distribution of covariance matrix.Then calculate the Kalman gain:

$$K_k = Conva_k^{'} A_k^T (A_k Conva_k^{'} A_k^T + R)^{-1} \quad (4)$$

$$Mean_k = Mean_k^{'} + K_k(Z_k - A_k Mean_k^{'}) \quad (5)$$

where K_k is the Kalman gain and Z_k is the system measurement value at k frame.[37, 16]

Finally,the convariance matrix of the error between the calculated and the real value is updated:

$$Conva_k = (1 - K_k A_k) Conva_k^{'} \quad (6)$$

3.4. Objects Selection

In the experiment, we notice that there are some defects in using only the motion compensation to compensate unconditionally. In this section, we further introduce how to use the motion compensated bounding box to select correct objects. In other words, that is how to find out the missed detection objects from the lost objects.

Confidence Interference Filtering. Error bounding box (EBBox) is caused by the fact that the tracked object has been lost or the object has not been detected in many frames but the boundary box compensation is still carried out. Therefore, we set the frame number of loss compensation within 30 and suppress the generation of EBBox by setting the threshold value of compensation confidence for judgment. We define compensation confidence as:

$$\mathcal{C} = \mathbb{I}\{S_{ts} - L_{ts} > 0\} \qquad s.t. S_{ts} > C_F \quad (7)$$

where $\mathcal{C}$ is compensation confidence;S_{ts} is the number of times that the object is successfully tracked, L_{ts} is the number of times the that the object is lost in tracking, and
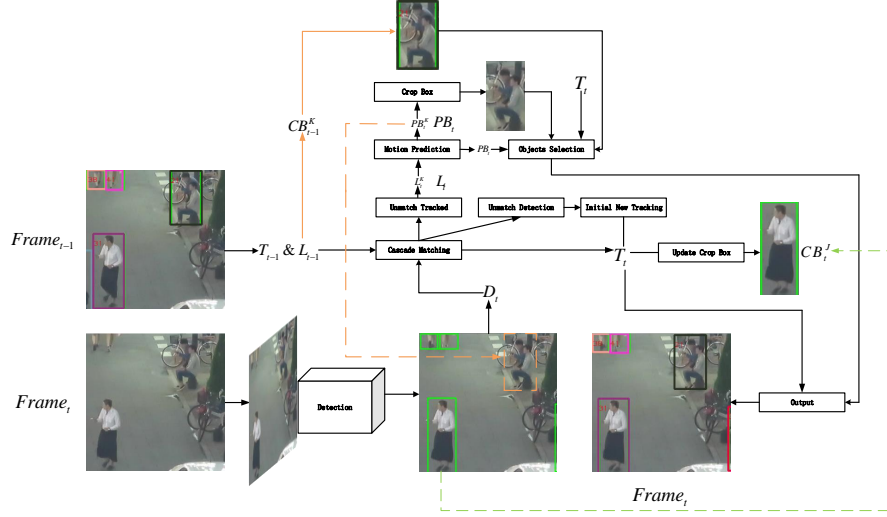
Figure 4. Compensation Tracker (CT) processing. (The orange line is an example of an unmatched tracked object in the current frame. The green line is an example of a successfully matched and tracked object) The figure above is for the case of lost object. When the object is tracked at frame T-1 but is not detected at frame T, the object is sent to the compensation tracker for motion compensation and objects selection. Meanwhile, other object will still match and go on data association. Then, after comprehensively processing the prediction results, selection results and association results, the tracker outputs the final compensation results.

Figure 5. A represents that formula (7) is not added, and B represents that formula (7) is added. The green box represents the successful detection of the detector. It is obvious from the figure that after the compensation confidence is added, the bounding box that does not meet the condition will be filtered out and the bounding box 17 is retained because it satisfies the formula condition.

C_F is the maximum number of compensation frame. When the predicted and updated object does not meet the above formula, the object will be filtered out. (As shown in Figure 5)

Boundary Interference Filtering. Experiments show that when the tracked scene moves relatively fast, only using the error bounding box suppression will not achieve the optimal effect. Therefore, when the object has disappeared in the image but the pedestrian bounding box still exists in the image, we need to judge the position of the center point of the pedestrian bounding box as follows:

$$\mathcal{C} = \mathbb{I}\{x - x_w * \alpha > 0 \quad \& \quad w - x - x_w * \alpha > 0\} \quad (8)$$

Where x is the center point of the bounding box; x_w is the

width of the bounding box; α is the weight of the bounding box width, which is set to 0.22 in the algorithm1; w is the width of the image. When bounding box does not satisfies formula (8), the target pedestrian bounding box will be filtered out. As shown in Figure 6

IoU Interference Filtering. In order to improve the tracking accuracy, we also eliminate the bounding box for pedestrian occlusion and pedestrian overlap to reduce the false detection rate. The effect can be seen in Figure 7 F. Our method uses the motion compensation bounding box and the tracked pedestrian bounding box set T_t for comparison and judgment, including their area ratio, IoU, and bounding box embedding degree.

Appearance Interference Filtering. Since IoU filtering cannot solve the problem of the people occluded by other objects and the serious drift of the bounding box, we use the bounding box image feature to solve it. Specifically, we add a property of bounding box image CB for each tracked pedestrian. When the pedestrian cannot be tracked, we extract the bounding box image CB_{t-1}^K previous frame. After that, it will be downs-sampled twice after a Gaussian kernel filter, and the resulting image is computed for Gauss difference with CB_{t-1}^K [21]:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (9)$$

Among them, $D(x, y, \sigma)$ is a Gauss differential image, $L(x, y, \sigma)$ is the convolution of the original image and the Gauss kernel,σ the scale space operator of the Gauss kernel function. After getting $D(x, y, \sigma)$ from (9), the algorithm

finds the extremum in $D(x, y, \sigma)$ and inputs 128 points near the extremum into (10) to get the modulus and direction of each extremum region.[21]

$$\begin{cases} m(x,y) = || \, [L(x+1,y) - L(x-1,y)] + \\ \qquad\qquad [(L(x,y+1) - L(x,y-1)] \, ||_2 \\ \theta(x,y) = tan^{-1}\left[\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right] \end{cases} \quad (10)$$

Where $m(x, y)$ is the modulus of the extreme point and $\theta(x, y)$ is the direction of the extreme points.

Afterwards, the pedestrian bounding box predicted by motion compensation is used to intercept the position of the pedestrian bounding box in the image and then the above operation is performed again. Finally, a K-Mean (KNN) [13] match is processing. If there are enough matching point, the pedestrian is assumed to be able to be tracked in the field. The matching formula is as follows:

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

Where, $\mathcal{K}$ is the K-Mean Function, $CB_{t-1}^K(m_{t-1}, \theta_{t-1})$ is the clipping bounding box that the k-th target finally tracked; CB_t^K is the bounding box clipped by motion compensation for the K-th target at frame t. m_{t-1}, θ_{t-1} and m_t, θ_t are the modulus and direction of feature points respectively, which are obtained by (9)(10); σ_m is the point threshold for K-Mean function matching.

Bounding Box Correction. Since the predicted bounding box will have an inaccurate size, the target cannot be accurately marked. In order to solve this problem, we use the bounding box information obtained by motion compensation and the previously tracked bounding box information for error calculation. Secondly, we believe that the size of the bounding box of two adjacent frames changes very little. When the area change of the above two bounding boxes is greater than 1.1, the compensated bounding box will be resized. Otherwise the compensated bounding box will be used directly. The flow chart of the entire compensation tracking algorithm is shown in Algorithm 1.

4. Experiments

4.1. Experiments Details

In the experiment, we use the datasets on MOT Challenge for testing. The platform provides a variety of datasets. These datasets contain videos of different scenes including moving and fixed visual scenes. At the same time, the datasets include all kinds of scenes such as pedestrian occlusion, pedestrian overlap, sparse pedestrians and dense pedestrians. We have conducted tests and evaluations on MOT2015[18] MOT2016[25], MOT2017[25] and the latest

---

Algorithm 1: Compensation Tracker

Input:
    initialize Track T_t
    Lost Object $l_k = \{l_i | l_i \in L_t\}$
    The K-th Cropping box: CB_{t-1}^K
    The K-th Cropping box for frame t: CB_t^K
Output: object sets T_t of the video

1   for $l_k \in L_t$ do
2     $\mathcal{C} \leftarrow$ Get_Compensation_Confidence(l_k) by(7)
3   if $\mathcal{C}$ do:
4       $Mean_k{'}, Conva_k{'} \leftarrow$ Get_Estimated($l_k.Mean$, $l_k.Conva_k$) by(2)(3)
5       $Mean_k, Conva_k \leftarrow$ Update_Value($Mean_k{'}$, $Conva_k{'}$) by (4)(5)(6)
6       $Mean_k \leftarrow$ Boundary_Interference_Filtering ($Mean_k{'}, Conva_k{'}$)
7       $Mean_k \leftarrow$ Bounding_Box_Correction ($Mean_k, l_k.Mean$)
8   for $T_t^l \in T_t$ do:
9        $l_k \leftarrow$ IoU_filter($Mean_k, T_t^l$)
10   end for
11       $l_k \leftarrow$ Update_Parameter($Mean_k, Conva_k, \mathcal{C}$)
12       $\mathcal{C} \leftarrow$ Calculate_Apperrance_Features (CB_{t-1}^K, CB_t^K) by(9)(10)(11)
13   end if
14   if $\mathcal{C}$ do :
15       $T_t^{l+=1}.append(l_k)$
16   end if
17   end for

Figure 6. Schematic diagram of boundary filter calculation. C is the effect picture before adding formula (8). D is the effect picture after adding formula (8). The width and height of the image are 640 and 480; the center point of the bounding box 12 is (43, 250) and the width is 195.73, and the center point of the bounding box 129 is (596, 242) and the width is 206.9. Since the bounding box 12 satisfy formula (8), the bounding box will be retained. Since the bounding box 129 dose not satisfy the formula (8), it will be filtered out.

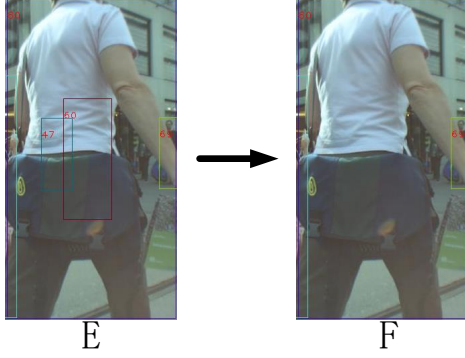MOT2020[8]. And we have achieved relatively good results

E   F

Figure 7. The comparison chart before and after using the IoU interference filter module. E is the effect diagram without using IoU module. F is the effect diagram after using this module. It is obvious from the figure that the wrong bounding boxes has been well suppressed.

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

Table 1. Module ablation experiment on MOT2020 Test. The best result for each indicator will be bold and red.

MOT2016 Test					
Models	MOTA↑	IDF1↑	MT↑	ML↓	ID.Sw↓
JDE[36]	64.4	55.8	20.0%	34.0%	1544
JDE with CT	65.0	59.1	36.1%	18.8%	1525
FairV1[43]	68.7	70.4	39.5%	19.0%	953
FairV1 with CT	69.8	71.1	42.0%	15.8%	912
MOT2020 Train					
Models	MOTA↑	IDF1↑	MT↑	ML↓	ID.Sw↓
JDE[36]*	48.2	32.1	318	497	18631
JDE with CT	54.4	43.1	526	372	11157
FairV1[43]*	62.3	47.5	790	288	16395
FairV1 with CT	65.6	57.5	1030	247	7816

Table 2. 'Private' model ablation experiment. '*' means that the model data is evaluated by motchallenge-devkit. The best result for each indicator will be bold and red

and indicator data in these datasets.

Experiment Platform. Our experiment is implemented on Pytorch. The computer used in the experiment is Intel(R) i5-9400H CPU @2.9GHz and GTX1080Ti graphics card. At the same time, we only use the training model parameters of the DLA network for experiments. The DLA network is trained through the following datasets, which include: MOT2017[25], Caltech[9],

CityPersons[44], CuhkSysu[38], PRW[45], ETH[10]. We conduct 30 epochs of training on 4 GPUs based on the DLA-34 pre-trained models. The learning rate and the batch size are set to 1*e-4 and 12 respectively. Then it will be fine-tuned 10 epochs on the MOT17 dataset [43][36]. More importantly, because our tracker retains the original confidence of the lost object and increases the confidence of the compensation, the overall confidence range is now between 0.3 and 0.6 for the best results.

Evaluation Indicators. Our experiment uses data indicators CLEAR Metrics [2] and IDF1, including multi-object tracking accuracy (MOTA), ID switching (ID Switch), the number of correct detections and the ratio of Ground True (IDF1), multi-object tracking accuracy (MOTP), the most tracked object (MT), the most lost object (ML), the average number of false alarms per frame (FAF) and the number of times the tracking process is interrupted (Frag) [7].

4.2. Ablation Experiments

In this part, we will use the Compensation Tracker (CT) on different models. And the network parameters model used in the 'Private' experiment do not use MOT2020 training sets for training. Compared with other datasets, the MOT2020 datasets are denser with pedestrians and more demanding on the performance of the detector.

As can be seen in Table 1, after using the motion compensation, the MOTA, IDF1 and ID switching have been significantly improved due to compensation for the missed object. However, because of unconditional compensation, some EBBoxes will still appear. Therefore, after adding the objects selection on this basis, MOTA and IDF1 can be further improved while only a small amount of ID switching is raised.

As can be seen in Table 2, in the MOT2016 test sets, after using our tracker, the effect of the JDE model is better than the original one. Among them, IDF1 has a 3.3% increase, and MOTA has a 1.4% increase. More importantly, MT increased by 16.1% and ML decreased by 15.2%. This shows that our tracker can effectively improve tracking performance and optimize the entire model. Especially in the 2020 training sets, the improvement effect is more prominent, and various indicators have improved greatly. Among them, MOTA increased by 6.2%, IDF1 increased by 11%, and ID switching decreased by 7474. For the baseline model, various indicators have also been improved in MOT2016. Among them, MOTA increased by 1.1%, MT increased by 2.5%, ML decreased by 3.2%, and ID switching decreased by 41. Furthermore, it can also be seen that in the dense scene MOT2020, after using our tracker, the tracking instability problem is further alleviated, and various indicators such as MOTA , IDF1 have been improved to a certain extent. Also, the ID switch dropped by 8579

This is because the compensation tracker alleviates the

MOT2020 Test							
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FAF↓	ID.Sw↓	Frag↓
Sort[3]	42.7	45.1	78.5	16.7%	26.2%	6.1	4470	17798
Sort+CT	43.3	45.2	78.2	17.6%	26.3%	6.3	2971	7485
MOT2020 Train							
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FAF↓	ID.Sw↓	Frag↓
Sort[3]*	45.8	34.1	87.9	288	593	/	12992	/
Sort+CT	52.9	52.2	87.1	424	417	/	3739	/

Table 3. 'Public' model ablation experiment. '*' means that the model data is evaluated by motchallenge-devkit. The best result for each indicator will be bold and red. Note: The training set assessment tool does not provide the results of the FAF and Frag indicators

MOT2016 Test						
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	ID.Sw↓
VMaxx[35]	62.6	49.2	78.3	32.7%	21.1%	1389
RAR16wVGG[11]	63.0	63.8	78.8	39.9%	21.1%	482
JDE[36]	64.4	55.8	/	35.4%	20%	1544
CNNMTT[24]	65.2	62.2	78.4	32.4%	20%	946
POI[41]	66.1	65.1	79.5	32.4%	20.8%	805
SORTwHOD16[3]	59.8	53.8	79.6	25.4%	22.7%	1423
Tube TK POI[28]]	66.9	62.2	78.5	39.0%	16.1%	1236
CTracker[29]	67.6	57.2	78.4	32.9%	23.1%	1897
FairV1[43]	68.7	70.4	80.2	39.5%	19.0%	953
Ours	69.8	71.1	80.0	42.0%	15.8%	912
MOT2017 Test						
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	ID.Sw↓
SST[33]	52.4	49.5	76.9	21.4%	30.7%	8431
Tube TK[28]]	63.0	58.6	78.3	31.2%	19.9%	4137
CTracker[29]	66.6	57.4	78.2	32.2%	24.2%	5529
FairV1[43]	67.5	69.8	80.3	37.7%	20.8%	2868
Ours	68.8	70.2	80.0	40.8%	17.7%	2805

Table 4. Comparative experiment of MO20T16 and MOT2017. The best result for each indicator will be bold and red

problem of detector instability and makes up for the missed object tracking, so that these missed object can be effectively tracked. Our tracker can not only accurately compensate for missed object, but also reduce unnecessary ID switching.

For the sake of further demonstrating the data association performance of the compensation tracker, we used Sort, a 'Pubic' tracker, for ablation experiments.

As can be seen in Table 3, in the MOT2020 test sets, after using the tracker of this design, the performance of Sort can be further improved. Among them, MOTA increased by 0.6%, and MT increased by 0.9%. Especially in ID switching and Frag indicators, the reductions are 1499 and 10313 respectively. In the MOT2020 training sets, the performance of our tracker has been greatly improved. Among them, MOTA increased by 7.1%, IDF1 increased by 18.1%, and ID switching decreased by 9253. In addition, it can be seen that the two tracking performance indicators in MT and ML have also been greatly improved. This shows that

the compensation tracker is very effective in data association and can further improve the tracking performance of the models.

4.3. Compare with the state-of-art Models

In this part, our method will be compared with the current state-of-art multi-object tracking models. These models, including one-stage model and two-stage model, belong to the 'Private' ranking list.

Comparative Experiment. As can be seen in Table 4, the test results of this method on the MOT2016 and MOT2017 datasets are outstanding. And there are improvements in MOTA, IDF1 and MT and ML. In the MOT2016 test sets, the MOTA indicator is 69.8%, the IDF1 indicator is 71.1%, the MT indicator is 42%, and the ML indicator is 15.8%. In addition, in the MOT2017 test sets, MOTA, IDF1, MT and ML are 68.8%, 70.2%, 40.8%, and 17.7%, respectively.

The effect of using our tracker is more obvious on

MOT2020 Test								
Models	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FAF↓	ID.Sw↓	Frag↓
FairV1[43]	58.7	63.7	77.2	66.3%	8.5%	24.7	6013	8140
Ours	66.0	67.0	77.8	56.3%	13.3%	9.8	2237	4154

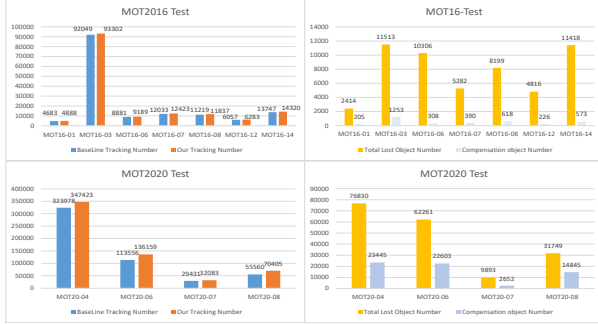Table 5. Comparative experiment of MOT2020. The best result for each indicator will be bold and red.

Figure 8. MOT2020 Test and MOT2016 Test Data Statistics

C_F	MOTA↑	IDF1↑	ID.sw↓	σ_m	MOTA↑	IDF1↑	ID.sw↓
10	46.5	66.0	284	1	45.5	65.2	278
20	46.2	66.3	266	3	45.7	64.9	284
30	46.2	65.6	263	5	46.0	65.5	282
40	46.0	65.8	269	7	46.2	64.8	274

Table 6. Analysis of different compensation frame value C_F and matching threshold σ_m on MOT15 Train. The experiment uses σ_m=5 and C_F=30 as fixed values respectively. The best result for each indicator will be bold and red

MOT2020. It can be clearly seen in Table 5 that compared with the baseline model, after using the compensation tracker, the effect has been greatly improved. Especially in the two comprehensive indicators in MOTA and IDF1, after using the compensation tracker, it can increase by 7.3% and 3.3% respectively. The average number of false alarms per frame FAF has dropped by 14.9% overall. Frag indicators have been greatly eased, reducing the trajectory fragmentation of 3986 object. The total number of ID Switches has dropped from 6013 to 2237. Therefore, combining the performance on the three datasets, our tracker can effectively improve the tracking performance without harming the performance of the existing models.

5. Analysis of Tracking Results

In the tracking by detection model, whether it is end-to-end or non-end-to-end, it is very dependent on the performance and stability of the detector. When the model can track a certain object in the previous frame, but the object cannot be detected in the current frame due to the lack of stability of the detector. The object will be judged to lost tracking. This behavior will affect the tracking continuity and tracking effect of the object.(As shown in Figure 1)

As can be seen from Figure 8.Each data set has the situation of losing the targets. These targets include those that are not in the tracking area and those that cannot be tracked in the current frame. Objects not tracked in the current frame can be divided into two categories: objects missed by the detector and objects blocked by objects or the same kind object. The compensation tracker designed by us is to re-track the missed target of the detector.

For MOT2016 dataset, the compensation tracker screened out some missed targets in each sub-data set and lost objects. As can be seen in the first row of the chart, in terms of the total output quantity, the total target quantity output by the compensation tracking method is equal to the baseline output quantity plus the number of targets considered missed by the detector after motion compensation and object selection.

In the MOT2020 data set of dense scenes, our tracker compensates more targets. This shows that the detector is unstable in dense pedestrian scene. However, our compensation tracker can compensate these missed targets (As shown in Figure 1), It can improve the tracking performance. Because we use the historical information (ID, appearance information, etc.) of this kind of target, we also reduce unnecessary ID switching.

Hyperparameter Experiment. There are two adjustable parameters for the compensation tracker, namely the compensation frame number C_F and the feature point matching threshold σ_m. We perform parameter sensitivity verification in the MOT2015 training set. It can be seen from Table 6 that the two parameters have little effect on the tracking effect. In the experiment, the value of the number of compensation frames should not be too low. We use 30 in all experiments. Otherwise, the number of ID switching will increase. In addition, the matching threshold should be selected appropriately, and the value of 5 or 7 can be selected in the experiment.

6. Conclusion

In this paper, we propose a simple compensation tracker that can be ported to other non-end-to-end models. Although the existing detection-based tracking methods have overall preference for the effect of multi-object tracking,

their problems in data association and missed detection are still prominent. The detection-based method is easy to make the tracking performance of the whole model worse because of the detector's omission. Our method can effectively solve this problem to a certain extent. In the compensation tracker, the historical information is used to compensate and track the lost target to improve the overall performance of the detector. According to the experimental results, the tracker proposed in this paper can improve different baseline networks in different degrees.

Acknowledgements

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. 1

[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. 6

[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. 3, 7

[4] 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. 1

[5] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2

[6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017. 2

[7] 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. 6

[8] 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. 5

[9] 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. 6

[10] 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. 6

[11] 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. 7

[12] Weitao Feng, Zhihao Hu, Wei Wu, Junjie Yan, and Wanli Ouyang. Multi-object tracking with multiple cues and switcher-aware classification. *arXiv preprint arXiv:1901.06129*, 2019. 2

[13] 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. 5

[14] 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. 2

[15] 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. 2

[16] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960. 2, 3

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

[18] 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. 5

[19] 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. 2

[20] Chen Long, Ai Haizhou, Zhuang Zijie, and Shang Chong. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*, 2018. 1

[21] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 4, 5

[22] 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. 1

[23] 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. 2

[24] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools and Applications*, 78(6):7077–7096, 2019. 2, 7

[25] 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. 5, 6

[26] Anton Milan, Seyed Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. *arXiv preprint arXiv:1604.03635*, 2016. 2

[27] Alejandro Newell, Zhiao Huang, and Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. In *Advances in neural information processing systems*, pages 2277–2287, 2017. 2

[28] 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. 2, 7

[29] 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. 2, 7

[30] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1, 2

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2

[32] Bing Shuai, Andrew G Berneshawi, Davide Modolo, and Joseph Tighe. Multi-object tracking with siamese track-rcnn. *arXiv preprint arXiv:2004.07786*, 2020. 2

[33] 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. 7

[34] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv preprint arXiv:1905.11946*, 2019. 2

[35] Xingyu Wan, Jinjun Wang, Zhifeng Kong, Qing Zhao, and Shunming Deng. Multi-object tracking using online metric learning with long short-term memory. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 788–792. IEEE, 2018. 7

[36] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2019. 1, 2, 6, 7

[37] 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. 1, 3

[38] 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. 6

[39] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3988–3998, 2019. 2

[40] 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. 2

[41] 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. 2, 7

[42] 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. 2

[43] Yifu Zhan, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*, 2020. 1, 2, 6, 7, 8

[44] 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. 6

[45] 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. 6

[46] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020. 1

[47] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2

[48] 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. 2

[49] 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. 2, 3