Towards Discriminative Representation: Multi-view Trajectory Contrastive Learning for Online Multi-object Tracking

En Yu[1]*, Zhuoling Li[2]*, Shoudong Han[1]†
[1]Huazhong Univerisity of Science and Technology [2]Tsinghua University
{yuen, shoudonghan}@hust.edu.cn lzl20@mails.tsinghua.edu.cn

Abstract

Discriminative representation is crucial for the association step in multi-object tracking. Recent work mainly utilizes features in single or neighboring frames for constructing metric loss and empowering networks to extract representation of targets. Although this strategy is effective, it fails to fully exploit the information contained in a whole trajectory. To this end, we propose a strategy, namely multi-view trajectory contrastive learning, in which each trajectory is represented as a center vector. By maintaining all the vectors in a dynamically updated memory bank, a trajectory-level contrastive loss is devised to explore the inter-frame information in the whole trajectories. Besides, in this strategy, each target is represented as multiple adaptively selected keypoints rather than a pre-defined anchor or center. This design allows the network to generate richer representation from multiple views of the same target, which can better characterize occluded objects. Additionally, in the inference stage, a similarity-guided feature fusion strategy is developed for further boosting the quality of the trajectory representation. Extensive experiments have been conducted on MOTChallenge to verify the effectiveness of the proposed techniques. The experimental results indicate that our method has surpassed preceding trackers and established new state-of-the-art performance.

1. Introduction

As a fundamental vision perception task, multiple object tracking (MOT) has been extensively deployed in broad applications, e.g., autonomous driving, video analysis and intelligent robots [6, 41]. Previous MOT methods mainly adopt the tracking-by-detection paradigm [4, 27, 29, 39], which mainly comprises two parts, i.e, detection and association. For the detection part, a detector is established to localize objects of interest. In the association part, some methods utilize a motion predictor for forecasting the positions of objects in the next frames and rely on the position information to associate them [16]. However, when these methods are applied to the challenging cases where targets are missing for several frames, it is hard to reconnect these targets to the corresponding trajectories correctly.

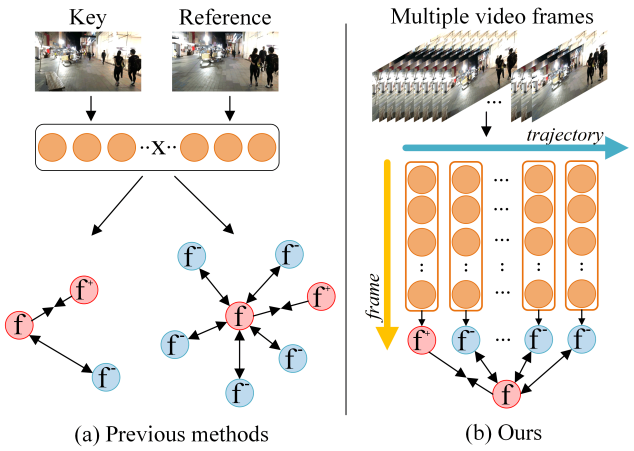


Figure 1. **Comparison between existing methods and our proposed method.** (a) Existing methods only utilize the information in a single or two adjacent frames to learn representation. (b) On the contrary, our method fully exploits the features in the whole trajectories, which contain numerous frames.

To alleviate this problem, existing trackers seek help from appearance-based strategies [39, 47, 51], in which objects are identified based on the similarity of extracted features. Nevertheless, the effectiveness of the appearance-based association strategy is still limited. Many objects with different identities are associated with the same trajectory because they are occluded or blurry, thus causing the learned representation to be indistinguishable. Hence, extracting more meaningful and discriminative representation is desired for enhancing the association accuracy.

In order to improve the quality of the extracted representation, we revisit existing representation learning methods in MOT and observe that they only use samples in a single or neighboring frames to construct loss for training

*Equal contribution
†Corresponding author

(a) Anchor-based (b) Center-based (c) Ours

Figure 2. **Comparison among various strategies for representating targets**: (a) Anchor-based, (b) Center-based, (c) Learnable view sampling (ours).

networks [17, 28, 39, 41, 51], which is illustrated in Fig. 1 (a). However, an object usually appears in many frames of a video, which compose a trajectory. Almost all existing methods fail to make full use of the information contained in the whole trajectories. Given this observation, we raise a new question: *is it feasible to fully exploit the trajectory information for boosting the discriminability of the target representation?*

A possible solution to this question is constructing contrastive loss [18] using all target representation vectors in trajectories. Nevertheless, since the trajectories in a video could include thousands of instances, this solution requires massive computing resource, which is unaffordable. To address this problem, we propose a strategy named *multi-view trajectory contrastive learning* (MTCL). In this strategy, we first model every trajectory as a vector, namely *trajectory center*, and establish a trajectory-center memory bank (TMB) to maintain these trajectory centers. Every trajectory center in this memory bank is updated dynamically during the training process. Afterwards, we regard the target appearance vectors as queries and devise a contrastive loss to draw them closer to their corresponding trajectory centers while farther away from other trajectory centers, which is shown in Fig. 1 (b). In this way, our method is able to exploit the inter-frame trajectory information while only consuming limited memory.

Moreover, we develop a strategy named *learnable view sampling* (LVS), which serves as a subcomponent of MTCL to explore the intra-frame features. As depicted in Fig. 2, LVS represents each target with multiple adaptively selected keypoints rather than anchors or their 2D centers. These keypoints gather at the meaningful locations of the targets and provide richer views to the aforementioned trajectory contrastive learning process. Additionally, LVS has an extra benefit. As illustrated in Fig. 2 (a) and (b), the anchors or 2D centers of targets are occluded by other objects, while LVS can still focus on visible regions adaptively.

Furthermore, in the inference stage, we note that the target features of some frames are unclear and inappropriate to represent trajectories. Correspondingly, we devise a *similarity-guided feature fusion* (SGFF) strategy that adaptively aggregates features based on the historical feature similarity to alleviate the influence of these poor features on the trajectory representation.

Incorporating all the above proposed techniques, the resulting model, namely *multi-view tracker* (MTrack), has been evaluated on four public benchmarks, i.e., MOT15 [23], MOT16 [26], MOT17 [26] and MOT20 [12]. The experimental results indicate that all our proposed strategies are effective and MTrack outperforms preceding counterparts significantly. For instance, MTrack achieves IDF1 of 69.2% and MOTA of 63.5% on MOT20.

2. Related Work

Multiple-object Tracking. Thanks to the rapid development of 2D object detection techniques [8, 30, 35, 53], recent trackers mainly adopt the tracking-by-detection paradigm [4, 6, 27, 29, 39, 41]. The trackers following this paradigm first utilize detectors to localize targets in each frame and then employ an associator to link detected objects of the same identity to form trajectories.

Traditional methods [6] usually perform temporal association via motion-based algorithms, such as Kalman Filter [40] and optical flow [3]. However, these algorithms behave poorly when targets move irregularly. In contrast to these traditional methods, some recent methods employ neural networks to predict the locations and displacements of targets in the next frames jointly [4, 14, 32, 33, 52]. However, when these methods are applied to complex scenarios where objects are occluded and invisible for several frames, the tracking results become unsatisfactory.

To mitigate the aforementioned problems, appearance-based methods [17, 24, 39, 41, 42, 47, 51] are introduced. These methods utilize neural networks to extract features of detected targets. The extracted features are desired to be discriminative, which means the features of objects with the same identity are similar while the ones corresponding to different identities are diverse. Since the appearance-based methods associate targets based on the similarity of extracted features, how to produce discriminative features is critical for them. In this work, we propose the MTCL which enables a model to learn more discriminative features through the proposed trajectory contrastive learning.

Constrastive Learning. Contrastive learning has been widely studied due to its impressive performance in the field of self-supervised learning [10, 18, 19, 38]. Given some images, contrastive learning first transforms every image into various views through random augmentation. Its optimization objective is drawing a view closer to the views augmented from the same image, and pushing it away from the views that originate from other images [18].

Although contrastive learning has been deployed in many fields, such as classification [9, 11] and detection

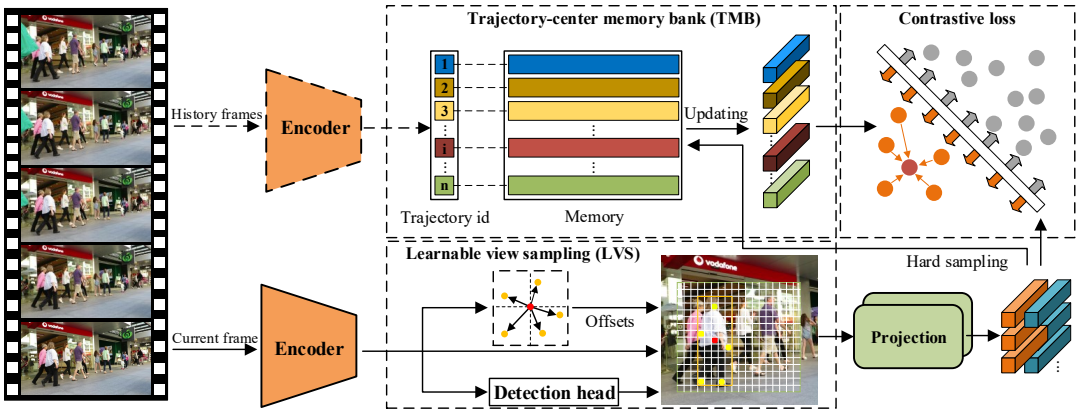


Figure 3. **Overall pipeline of multi-view trajectory contrastive learning**. Given a video $\mathbb{V}_\xi$ with ξ frames I_t (t=1,2,...,ξ), MTCL comprises 4 steps: (1) Employ an encoder (the backbone) to extract feature maps from the current input frame. (2) Use LVS to select informative keypoints from the extracted feature maps, and transform the features of selected keypoints as target appearance vectors by a projection head. (3) Conduct contrastive learning between the appearance vectors and trajectory centers stored in the memory bank. (4) Update trajectory-center memory bank using our hard sampling strategy.

[43, 45], few works applies it to MOT. Recently, QDTrack [28] serves as the first work that utilizes contrastive learning in MOT to improve the learned representation. However, it only uses the samples in adjacent two frames and fails to explore information in the whole trajectories. This is because directly constructing contrastive loss with all samples in trajectories leads to a tremendous computing burden, which is unaffordable. On the contrary, our proposed MTCL can fully exploit the trajectory information with very limited computing resource, which is realized by only storing one vector for every trajectory in the memory.

3. Methodology

This section explains how MTrack is implemented. First of all, Sec. 3.1 presents an overview of MTrack. Afterwards, Sec. 3.2 introduces our proposed MTCL, which includes two subcomponents, i.e., LVS and TMB. Finally, Sec. 3.3 describes the implementation of the SGFF strategy.

3.1. Overview

In this work, we implement MTrack based on CenterNet [53], which represents targets as their center points. DLA34 [48] is adopted as the backbone of the CenterNet. Given a video V_ξ of ξ frames as input, the backbone is firstly applied to extract feature maps. Then, several network heads are established to transform the feature maps into the desired properties of the targets, which include 2D center heatmaps, center offsets and bounding box sizes. Besides, we add an extra embedding head in parallel with the detection heads to extract appearance features. In our implementation, the detection bounding boxes are generated based on the estimated 2D center heatmaps, center offsets and bounding box sizes. The detected objects are associated according to their appearance feature similarities.

In this work, we focus on how to produce discriminative representation for realizing accurate association. Specifically, MTCL is applied during the training process to improve the ability of generating representative embedding vectors, which is illustrated in Fig. 3. SGFF is developed for improving the quality of the trajectory representation during the inference phase.

3.2. Multi-view Trajectory Contrastive Learning

In this subsection, we elaborate on our main contribution, MTCL. To explain it clearly, we first introduce the LVS strategy in MTCL, which generates multiple appearance vectors adaptively for every target and contributes to exploiting the intra-frame information more efficiently. Then, we describe the trajectory-center memory bank, which enables us to realize trajectory contrastive learning with only limited computing resource. Finally, we present the details of the trajectory-level contrastive loss and the overall process of MTCL.

Learnable view sampling. Existing trackers developed based on CenterNet mainly represent every target as a sole center point on feature maps. As introduced before, this strategy has two critical restrictions: (1) The center points of targets could be occluded by other objects, such as the case shown in Fig. 2 (b). In this case, the produced appearance vectors fail to reflect the characteristics of targets. (2) Representating every target with only one vector cannot provide sufficient samples to the contrastive learning algorithm.

To address the above restrictions, we devise LVS that represents a target as multiple adaptively selected keypoints. Specifically, given the t_{th} frame of a video as input, denoted as I_{t}, we first transform it into feature maps F_{t} using the backbone, and recognize the center points of all

targets in I_{t} like former trackers [51]. Denoting the center point coordinate of a target $q \in I_t$ on F_{t} as $Z^q = (x^q, y^q)$, we take out the vector at the coordinate of Z^q in F_{t} and represent this vector as r^q. The vector r^q contains the appearance information of q. Therefore, we can regress the offsets from Z^q to the potential informative keypoints in I_{t} by applying a linear transformation W. This process can be formulated as

$$\triangle Z^q = W r^q, \tag{1}$$

where $\triangle Z^q = \{\triangle Z_i^q\}_{i=1}^{N_k}$, N_k represents the total number of selected keypoints and $\triangle Z_i^q$ is the offset from Z^q to the i_{th} keypoint. Hence, the coordinate of the i_{th} selected keypoint is determined as:

$$Z_i^k = \Phi(Z^q + \triangle Z_i^q), \tag{2}$$

where $\Phi(\cdot)$ denotes an operator that guarantees all the keypoints to fall within the generated 2D bounding boxes. Specifically, if a keypoint is out of a 2D box, it will be clipped to the boundary of this box.

With the coordinate of the i_{th} keypoint Z_i^k, we can obtain its corresponding feature vector v_i^k from F_{t}. Then, v_i^k is further transformed into a more representative appearance vector $\widetilde{v}_i^k$ by applying a projection head consisting of 4 fully connected layers. Since N_k keypoints are selected for every target, we can obtain N_k appearance vectors $\widetilde{v}_i^k$ corresponding to different positions of a target. Representing every detected target as these N_k vectors provides richer intra-frame samples to the contrastive learning process. Notably, during the inference stage, the N_k appearance vectors are contatenated as a single one to represent their corresponding target.

Trajectory-center memory bank. Compared with intra-frame samples, inter-frame samples provide more informative features. However, directly utilizing all samples in the historical frames to construct contrastive loss consumes much computing resource. To alleviate this problem, we propose to represent every trajectory as a vector named *trajectory center*, and maintain all the trajectory centers in a memory bank.

Assuming there are N trajectories in a training dataset, the memory bank is initialized as a set containing N zero vectors $\{c_i\}_{i=1}^{N}$ at the beginning of a training epoch (a training epoch includes many iterations, and every iteration corresponds to an input data batch). This memory bank and its contained trajectory centers are not reinitialized until the end of this epoch. Since we do not save the gradient information of all iterations for saving memory, the trajectory centers cannot be updated by gradient-based optimizers, such as Adam [21]. To solve this problem, we develop a momentum-based updating strategy that dynamically updates trajectory centers in each iteration without requiring historical gradient information.

Hereby, we explain how the momentum-based updating strategy is implemented. For an input data batch comprising several frames, the network recognizes the instances of interest in all frames and generates N_k appearance vectors for each instance. During the training stage, every detected instance is labeled with a trajectory ID. We gather all appearance vectors extracted from the instances with the same trajectory ID to update their corresponding trajectory centers.

Specifically, the trajectory centers are updated after the parameters of the model are optimized based on the back-propagated gradient. Denoting all appearance vectors in a batch corresponding to the l_{th} trajectory as $\mathrm{P}^l = \{p_i^l\}_{i=1}^{N_l}$, we select the hardest sample in P^l to update c_l. The hard levels of samples are reflected by the cosine similarities [37] with their corresponding trajectory centers, and the sample with the minimum similarity is the hardest one. Mathematically, the cosine similarity s_i^l between p_i^l and c_l is formulated as:

$$s_i^l = \frac{p_i^l \cdot c_l}{\|p_i^l\|_2 \times \|c_l\|_2}, \tag{3}$$

where $\cdot$ and $\times$ represent the dot product between two vectors and the normal product between two floating numbers, respectively. $\|\cdot\|_2$ is a L_2 normalization operator.

Denoting the appearance vector in P^l with the minimum cosine similarity as p_m^l, c_l is updated given:

$$c_l \leftarrow \alpha c_l + (1-\alpha) p_m^l, \tag{4}$$

where α is a hyper-parameter falling between [0, 1].

During the training phase, updating trajectory centers with hard samples contributes to the efficiency of training a network. This issue has been confirmed by our experimental results.

Trajectory-level contrastive loss. LVS and TMB provide rich intra-frame and inter-frame samples for constructing the contrastive loss, respectively. The following problem is how to implement the contrastive loss, and train the embedding head of the network for producing discriminative representation.

For the k_{th} appearance vector $\tilde{v}_l^k$ produced by LVS and its corresponding trajectory center c_l, the optimization objective is drawing $\tilde{v}_l^k$ closer to c_l while pushing $\tilde{v}_l^k$ away from all other trajectory centers. Following [18], we employ the InfoNCE loss to realize this objective. Mathematically, the loss of $\tilde{v}_l^k$ can be formulated as:

$$L_{NCE}^k = -log \frac{exp(\tilde{v}_l^k \cdot c_l)/\tau}{\sum_{i=0}^{N_t} exp(\tilde{v}_l^k \cdot c_i)/\tau}, \tag{5}$$

where $\tau \in (0, 1]$ denotes a hyper-paramter and N_t is the total number of the trajectories. To fully exploit inter-frame

information, we calculate the trajectory-level contrastive loss for every appearance vector based on Eq.(5), and the embedding head loss L_{tcl} is formulated as:

$$L_{tcl} = \frac{1}{N_a}\sum_{k=1}^{N_a} L_{NCE}^{k}, \tag{6}$$

where N_a is the total number of appearance vectors. Overally, the total loss for training MTarck is:

$$L = \frac{1}{2}(\frac{1}{e^{\eta_1}}L_{det} + \frac{1}{e^{\eta_2}}L_{tcl} + \eta_1 + \eta_2), \tag{7}$$

where L_{det} represents the detection loss. η_1 and η_2 are learnable weights for balancing L_{det} and L_{tcl}.

Additionally, we give the pseduo code of MTCL in Algorithm 1 to present its process clearly.

Algorithm 1: Training Procedure of MTCL

Require: The feature encoder σ_θ;
Momentum rate α;
Temperature parameter τ;
Input : Training videos $\mathbf{V} = \{V_1, V_2, ...V_N\}$;

```
for each epoch do
    Initialize a trajectory-center memory bank B;
    for each mini_batch do
        Extract feature maps F_b by σ_θ(V_b);
        Detect all targets in V_b given F_b;
        Generate multiview appearance vectors
          by LVS with Eq. (1) and Eq. (2);
        Compute trajectory-level contrastive loss
          L_tcl with Eq. (5) and Eq. (6);
        Update B with Eq. (4);
    end
end
```

3.3. Similarity-guided Feature Fusion

In the inference phase, existing appearance-based trackers associate targets with trajectories based on the appearance similarities. First of all, these trackers compute a pairwise appearance affinity matrix between the targets and trajectories. After that, they associate the targets to the trajectories based on a greedy matching strategy, such as the Hungarian algorithm [22]. In this process, the discriminability of the trajectory representation is critical for the association accuracy.

Existing methods update the representation of trajectories by fusing the features of historical frames. Denoting the representation of the l_{th} trajectory in the $(t-1)_{\text{th}}$ frame as f_l^{t-1}, these methods employ a momentum-based updating strategy [51] to update f_l^{t-1} with the feature vector z_l^t, which is extracted from the newly matched object. Then, f_l^t is obtained. This process is formulated as follows:

$$f_l^t = (1-\beta)f_l^{t-1} + \beta z_l^t, \tag{8}$$

where β is a hyper-parameter.

Setting β to a constant is effective when z_l^t is informative. However, when targets are occluded or blurry, z_l^t is contaminated by noise and contains little valuable information. Under this condition, updating f_l^{t-1} with z_l^t is harmful to the trajectory representation.

To deal with this problem, we propose the SGFF that adjusts β for each frame adaptively, which is shown in Fig. 4. Specifically, assuming a target is clear in most recent frames, we can measure the quality of z_l^t by computing its similarities with the feature vectors in the latest Q frames. If z_l^t is similar to them, we can speculate that z_l^t is informative and β should be a large value. Therefore, denoting β in the t_{th} frame as β^t, β^t can be derived as:

$$\beta^t = \max\{0, \frac{1}{Q}\sum_{i=1}^{Q}\Psi_d(z_l^t, z_l^{t-i})\}, \tag{9}$$

where $\Psi_d(\cdot)$ represents an operator that computes the cosine similarity described in Eq. (3).

With the SGFF, β^t becomes a tiny value if z_l^t is of poor quality. Thus, this strategy reduces the effect of the poor feature vectors.

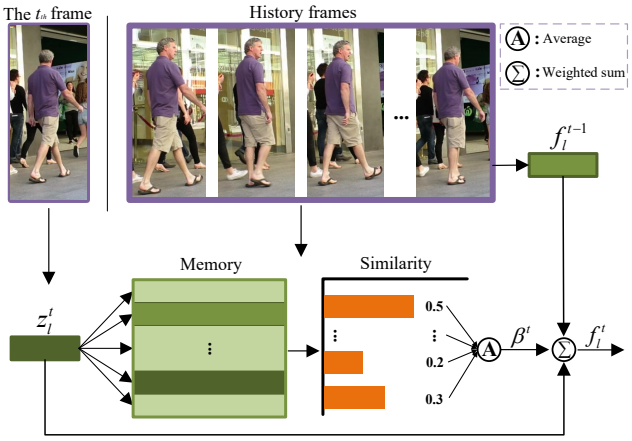


Figure 4. **Illustration of the similarity-guided feature fusion strategy.** In this strategy, β^t is adjusted adaptively for each frame according to the similarities between z_l^t and the features extracted from the recent frames.

4. Experiments

This section presents the experimental details. Specifically, Sec. 4.1 introduces the adopted datasets as well as the evaluation metrics. Sec. 4.2 describes the implementation details of our method. Then, Sec. 4.3 demonstrates the

superiority of MTrack by comparing it with existing state-of-the-art (SOTA) MOT methods. Sec. 4.4 reveals the effectiveness of the proposed techniques through various ablation experiments. Finally, Sec. 4.5 visualizes some extracted features and suggests that our method boosts the discriminability of the learned representation significantly.

4.1. Datasets and Evaluation Metrics

Datasets: We conduct extensive experiments on four public MOT benchmarks, i.e., MOT15 [23], MOT16 [26], MOT17 [26] and MOT20 [12]. Specifically, MOT15 comprises 22 sequences, one half for training and the other half for testing. This dataset in all contains 996 seconds of videos, which includes 11286 frames. MOT16 and MOT17 are composed of the same 14 videos, 7 for training and the other 7 for testing. The 14 videos cover various scenarios, viewpoints, camera poses and weather conditions. Compared with MOT16, MOT17 provides more detection bounding boxes produced by various detectors, which include DPM [15], SDP [46], Faster-RCNN [30]. MOT20 is the most challenging benchmark in these datasets. It consists of 8 video sequences captured in 3 crowded scenes. In some frames, more than 220 pedestrians are included simultaneously. Meanwhile, the scenes in MOT20 are very diverse, which could be indoor or outdoor, in the day or at night.

Evaluation metrics: MTrack is evaluated based on the CLEAR-MOT Metrics [5], which include ID F1 score (IDF1), multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), mostly tracker rate (MT), mostly lost rate (ML), false positives (FP), false negatives (FN) and identity switches (IDS). Among them, IDF1 and MOTA are the most important indexes for comparing performance.

4.2. Implementation Details

We adopt DLA-34 as the backbone and the detection branch of MTrack is pre-trained on Crowdhuman [31]. The parameters are updated using the Adam optimizer [21] with the initial learning rate of 10^{-4}. The learning rate is reduced to 10^{-5} at the 20_{th} epoch. The model is trained for 30 epochs totally. During the training process, the batch size is set as 8 and the resolution of every input image is 1088×608. The adopted image augmentation operations follow FairMOT [51], which include random rotation, scaling, translation and color jittering. For LVS, we set N_k to 9, and choose the center point and the eight points surrounding it as the initial sampling locations. In TMB, the temperature parameter τ is 0.05 and the momentum update factor α is 0.2. During the inference stage, Q is set as 30.

4.3. Comparison with Preceding SOTAs

In this part, we compare the performance of MTrack with preceding SOTA methods on four widely adopted benchmarks, i.e., MOT15, MOT16, MOT17 and MOT20. The results are reported in Tab. 1. Notably, some methods utilize numerous extra data with identity labels to improve their abilities on generating discriminative identity embedding. For fair comparison, we do not use extra training data from the other tasks (such as person search or ReID) to boost the tracking performance. According to the results in Tab. 1, our method surpasses all the compared counterparts significantly on IDF1, MOTA and the other metrics. Besides, the proposed strategies do not affect the inference speed significantly, and the inference speed of MTrack on RTX2080Ti is 23 FPS.

Method	IDF1	MOTA	MT	ML↓	FP↓	FN↓	IDS↓
MOT15 private detection							
DMT [20]	49.2	44.5	34.7%	22.1%	8,088	25,335	684
TubeTK [27]	53.1	58.4	39.3%	18.0%	**5,756**	18,961	854
CDADDAL [2]	54.1	51.3	36.3%	22.2%	7,110	22,271	544
TRID [25]	61.0	55.7	40.6%	25.8%	6,273	20,611	**351**
RAR15 [13]	61.3	56.5	**45.1%**	**14.6%**	9,386	**16,921**	428
MTrack	**62.1**	**58.9**	38.1%	17.5%	6,314	18,177	750
MOT16 private detection							
IoU [7]	46.9	57.1	23.6%	32.9%	**5,702**	70,278	2,167
JDE [39]	55.8	64.4	35.4%	20.0%	-	-	1544
CTracker [29]	57.2	67.6	32.9%	23.1%	8,934	48,305	1,897
TubeTK [27]	59.4	64.0	33.5%	19.4%	10,962	53,626	1,117
LMCNN [1]	61.2	67.4	38.2%	19.2%	10,109	48,435	931
DeepSort [41]	62.2	61.4	32.8%	18.2%	12,852	56,668	781
MAT [16]	63.8	70.5	44.7%	17.3%	11,318	41,592	928
TraDeS [42]	64.7	70.1	37.3%	20.0%	8,091	45,210	1,144
MOTR [49]	67.0	65.7	37.2%	20.9%	16,512	45,340	648
QDTrack [28]	67.1	69.8	41.6%	19.8%	9,861,	44,050	1,097
GMTCT [17]	70.6	66.2	29.6%	30.4%	6,355	54,560	701
MTrack	**74.3**	**72.9**	**50.6%**	**15.7%**	19,236	**29,554**	**642**
MOT17 private detection							
DAN [34]	49.5	52.4	21.4%	30.7%	25,423	234,592	8,431
Tracktor+CTdet [4]	57.2	56.1	25.7%	29.8%	44,109	210,774	2,574
CTracker [29]	57.4	66.6	32.2%	24.2%	22,284	160,491	5,529
TubeTK [27]	58.6	63.0	31.2%	19.9%	27,060	177,483	5,727
TransCener [44]	62.1	70.0	38.9%	20.4%	28,119	136,722	4,647
MAT [16]	63.1	69.5	43.8%	18.9%	30,660	138,741	2,844
TraDeS [42]	63.9	69.1	37.3%	20.0%	20,892	150,060	3,555
CenterTrack [52]	64.7	67.8	34.6%	24.6%	18,489	160,332	3,039
MOTR [49]	66.4	65.1	33.0%	25.2%	45,486	149,307	2,049
GMTCT [17]	68.7	65.0	29.4%	31.6%	**18,213**	177,058	2,200
QDTrack [28]	68.7	66.3	40.6%	21.9%	26,589,	146,643	3,378
MTrack	**73.5**	**72.1**	**49.0%**	**16.8%**	53,361	**101,844**	**2,028**
MOT20 private detection							
MLT [50]	54.6	48.9	30.9%	22.1%	**45,660**	216,803	**2,187**
TransCener [44]	50.4	61.9	49.4%	15.5%	45,895	146,347	4,653
FairMOT [51]	67.3	61.8	68.8%	7.6%	103,440	88,901	5,243
MTrack	**69.2**	**63.5**	**68.8%**	**7.5%**	96,123	**86,964**	6,031

Table 1. Performance comparison with preceding SOTAs on the testing splits of the MOT15, MOT16, MOT17 and MOT20 benchmarks under the private detection protocols. The best results are marked in bold and our method is highlighted in pink.

MOT15: According to Tab. 1, MTrack obtains the metric IDF1 of 62.1% and MOTA of 58.9%, which significantly outperforms all compared methods without using ex-

tra training data. Despite MOT15 containing many false annotations, MTrack still achieves outstanding performance.

MOT16 and MOT17: Tab. 1 shows our main results on MOT16 and MOT17. Since MOT16 and MOT17 contain more data and the annotations are more precise compared with MOT15, MTrack outperforms the compared methods by larger margins. For instance, MTrack surpasses the tracker CenterTrack, which is also built upon CenterNet, by $8.8\%(73.5\%-64.7\%)$ on IDF1 and $4.3\%(72.1\%-67.8\%)$ on MOTA in the MOT17 benchmark. Compared with QDTrack, a method that also applies contrastive learning to MOT, MTrack surpasses it by $4.8\%(73.5\% - 68.7\%)$ on IDF1 and $5.8\%(72.1\% - 66.3\%)$ on MOTA. The results demonstrate that learning from the entire trajectories in videos is more likely to empower the model to learn discriminative representation than learning from neighboring frames. Moreover, it can be observed that MTrack also behaves well on the metric of FN and IDS, which means the the generated trajectories are very continuous.

MOT20: To further prove the effectiveness of our method, we evaluate MTrack on the challenging MOT20 benchmark. As shown in Tab. 1, MTrack obtains the metric IDF1 of 69.2% and MOTA of 63.5%. It behaves the best compared with the counterparts that do not employ extra training data. Notably, MTrack performs better than FairMOT, which is pre-trained on numerous external training datasets, which include ETH, CityPerson, CalTech, CUHK-SYSU and PRW. MTrack surpasses FairMOT by $1.9\%(69.2\% - 67.3\%)$ on IDF1 and $1.7\%(63.5\%-61.8\%)$ on MOTA. The results further confirm the superiority of Mtrack, especially in very crowded scenarios.

4.4. Ablation Study

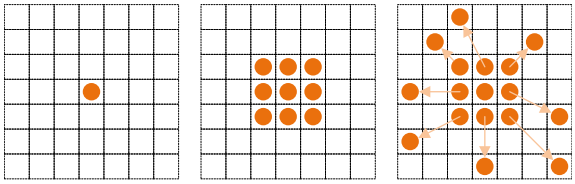
In this subsection, we verify the effectiveness of the proposed strategies separately through ablation studies. All the experiments are conducted on the MOT17 dataset. Since MOT Challenge does not provide the validation set, we divide the MOT17 datasets into two parts, $\frac{3}{4}$ as the training set and the other $\frac{1}{4}$ as the validation set. All the models are trained for 30 epochs on the training set of MOT17.

Analysis of the components of MTrack. In this part, we verify the effectiveness of various components in MTrack through an ablation study. The results are reported in Tab. 2. According to the results, all the components have boosted the tracking performance effectively. Incorporating all of them, MTrack (row 5) outperforms the baseline (row 2) by 3.1% on IDF1 and 2.2% on MOTA. Among these components, TMB (row 4) boosts the results with the largest margin. Specifically, it improves IDF1 by 1.1% and MOTA by 0.8%. This issue suggests that taking all the information in the whole trajectories into consideration is valuable for the tracking precision.

According to row 3, adding an extra projection head enhances the result on IDF1 significantly, which is a metric mainly reflecting the association accuracy. This observation is consistent with the conclusion drawn by previous publications about contrastive learning [18]. In addition, it can be noticed that the SGFF (row 5) also enhances the results notably (0.5% on IDF1 and 0.6% on MOTA), although it does not demand any modification to the training process.

	Method	*Components* LVS	Proj.	TMB	Loss	SGFF	*Metrics* IDF1	MOTA
1	Base	✗	✗	✗	CE	✗	78.4	71.6
2		✓	✗	✗	CE	✗	79.1	72.3
3		✓	✓	✗	CE	✗	79.9	72.4
4		✓	✓	✓	TCL	✗	81.0	73.2
5	MTrack	✓	✓	✓	TCL	✓	**81.5**	**73.8**

Table 2. **Ablation study of the components in MTrack.** The resulting MTrack that combines all the components is highlighted in pink (LVS: learnbale view sampling, TMB: trajectory-center memory bank, Proj.: projection, CE: cross-entropy, TCL: trajectory-level contrastive loss, SGFF: similarity-guided feature fusion).



(a) Center point (b) Center area (c) LVS (ours)

Figure 5. Illustration of three positive sampling strategies. (LVS: learnable view sampling)

Sampling Method	IDF1	MOTA	FP↓	FN↓	IDS↓
Center point	80.0	72.7	1835	**5345**	213
Center area	79.8	72.9	**1458**	5675	211
LVS	**81.5**	**73.8**	1524	5393	**183**

Table 3. Comparisions between different sampling strategies.

Analysis of LVS. In this part, we conduct an in-depth analysis on LVS. We compare it with two other keypoint sampling strategies, the center point and center area based sampling strategies, which are illustrated in Fig. 5 (a) and (b), respectively. Similar to LVS, the center area based sampling strategy provides 9 appearance vectors for constructing the contrastive learning loss. However, the 9 keypoints are pre-defined and not adjusted according to the content of the input image.

The experimental results are reported in Tab. 3. According to the two primary metrics, IDF1 and MOTA, the results obtained by the center point and center area based sampling strategies are similar. This phenomenon implies

that directly incorporating more appearance vectors into the training process cannot boost the performance of the model, and selecting keypoints adaptively (Fig. 5 (c)) is important.

Analysis of the operator $\Phi(\cdot)$. In this part, we analyze how the operator $\Phi(\cdot)$ in Eq. (2), which restricts the selected keypoints to fall within the estimated 2D bounding box, affects the training process. The curves of the embedding head loss L_{tcl} corresponding to the models trained with and without $\Phi(\cdot)$ are illustrated in Fig. 6. We can observe that $\Phi(\cdot)$ decreases the loss value and accelerates the convergence process significantly. This observation suggests that sampling keypoints only in the estimated 2D bounding boxes is valuable because it forces the generated keypoints to reflect the target appearance information, and employing these keypoints to implement contrastive learning leads to a model with better feature extraction ability.

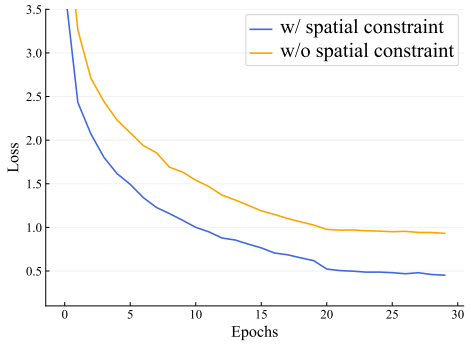


Figure 6. The loss curves of the models trained with and without the spatial constraint operator $\Phi(\cdot)$ on the MOT17 benchmark.

Analysis of the trajectory center updating strategy. As introduced in Sec. 3.2, we select the hardest appearance vector of a target to update its corresponding trajectory center. To analyze whether there exists a better strategy, we compare it with other three strategies, "Random", "Average" and "Easy". The results are reported in Tab. 4.

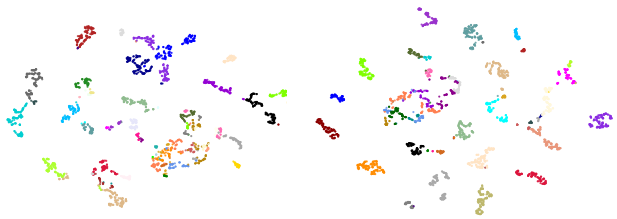
As shown in Tab. 4, "Hard" leads to the best results, and the performances of the other three strategies are similar. We speculate that this is because most appearance vectors of the same target generated from neighboring frames are similar, and employing them to update the trajectory center does not fully explore all information available. On the contrary, utilizing the hardest appearance vector alleviates this problem.

4.5. Embedding Visualization

In this subsection, we visualize the target appearance vectors produced by models with and without MTCL based on the t-SNE algorithm [36]. The model with MTCL is the same as MTrack, and the model without MTCL employs the cross-entropy loss to train the embedding head of CenterNet like [51]. The visualization results are given in Fig. 7.

Updating strategy	IDF1	MOTA	FP↓	FN↓	IDS↓
Random	80.3	73.1	1712	5382	190
Average	80.6	73.2	1669	5389	208
Easy	80.2	73.0	2058	**5071**	189
Hard	**81.5**	**73.8**	**1524**	5393	**183**

Table 4. Comparison between different trajectory center updating strategies. Random: randomly select an appearance vector; Average: Take the average of all appearance vectors; Easy: Select the appearance vector of the maximum cosine similarity with the trajectory center; Hard: Select the appearance vector with the minimum cosine similarity with the trajectory center.



(a) without MTCL (b) with MTCL

Figure 7. Visualizing the appearance vectors of some targets in MOT17 using the t-SNE algorithm. The points in the same color are with the same identity.

As shown in Fig. 7, the representation generated by the model with MTCL is more discriminative. The vectors corresponding to the same identity are well clustered, and the ones with different identities are clearly distinguished. Hence, MTCL is effective on improving the feature extraction ability of a network.

5. Conclusion

In this work, we have argued that the discriminability of the extracted representation is critical for MOT. However, existing work only exploits the features in neighboring frames, and the information in the whole trajectories is ignored. To bridge this gap, we have proposed a strategy named multi-view trajectory contrastive learning, which fully exploits the intra-frame features and inter-frame features at the cost of limited computing resource. In the inference stage, a strategy named similarity-guided feature fusion has been developed to alleviate the negative influence of poor features caused by occlusion and blurring. We have verified the effectiveness of the proposed techniques on 4 public benchmarks, i.e., MOT15, MOT16, MOT17 and MOT20. The experimental results indicate that these techniques can boost the tracking performance significantly. We hope this work can serve as a new solution to producing discriminative representation in MOT.

References

[1] Maryam Babaee, Zimu Li, and Gerhard Rigoll. A dual cnn–rnn for multiple people tracking. *Neurocomputing*, 368:69–83, 2019. 6

[2] Seung-Hwan Bae and Kuk-Jin Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *TPAMI*, 2018. 6

[3] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *Int J Comput Vis*, 56(3):221–255, 2004. 2

[4] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 1, 2, 6

[5] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP J Image Video Process*, 2008:1–10, 2008. 6

[6] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, pages 3464–3468. IEEE, 2016. 1, 2

[7] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, 2017. 6

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 2

[9] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020. 2

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607. PMLR, 2020. 2

[11] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2

[12] 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. 2, 6

[13] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *WACV*, 2018. 6

[14] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, pages 3038–3046, 2017. 2

[15] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2009. 6

[16] Shoudong Han, Piao Huang, Hongwei Wang, En Yu, Donghaisheng Liu, Xiaofeng Pan, and Jun Zhao. Mat: Motion-aware multi-object tracking. *arXiv preprint arXiv:2009.04794*, 2020. 1, 6

[17] Jiawei He, Zehao Huang, Naiyan Wang, and Zhaoxiang Zhang. Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking. In *CVPR*, pages 5299–5309, 2021. 2, 6

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 2, 4, 7

[19] Qianjiang Hu, Xiao Wang, Wei Hu, and Guo-Jun Qi. Adco: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries. In *CVPR*, pages 1074–1083, 2021. 2

[20] Han-Ul Kim and Chang-Su Kim. Cdt: Cooperative detection and tracking for tracing multiple objects in video sequences. In *ECCV*, pages 851–867. Springer, 2016. 6

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4, 6

[22] Harold W Kuhn. The hungarian method for the assignment problem. *Nav. Res. Logist.*, 2(1-2):83–97, 1955. 5

[23] 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. 2, 6

[24] Chao Liang, Zhipeng Zhang, Yi Lu, Xue Zhou, Bing Li, Xiyong Ye, and Jianxiao Zou. Rethinking the competition between detection and reid in multi-object tracking. *arXiv preprint arXiv:2010.12138*, 2020. 2

[25] Santiago Manen, Michael Gygli, Dengxin Dai, and Luc Van Gool. Pathtrack: Fast trajectory annotation with path supervision. In *ICCV*, pages 290–299, 2017. 6

[26] 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. 2, 6

[27] Bo Pang, Yizhuo Li, Yifan Zhang, Muchen Li, and Cewu Lu. Tubetk: Adopting tubes to track multi-object in a one-step training model. In *CVPR*, June 2020. 1, 2, 6

[28] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, pages 164–173, 2021. 2, 3, 6

[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 *ECCV*, pages 145–161. Springer, 2020. 1, 2, 6

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 28:91–99, 2015. 2, 6

[31] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018. 6

[32] Bing Shuai, Andrew Berneshawi, Xinyu Li, Davide Modolo, and Joseph Tighe. Siammot: Siamese multi-object tracking. In *CVPR*, pages 12372–12382, 2021. 2

[33] 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. 2

[34] ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Deep affinity network for multiple object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(1):104–119, 2019. 6

[35] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, pages 9627–9636, 2019. 2

[36] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *J Mach Learn Res*, 9(11), 2008. 8

[37] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, pages 5265–5274, 2018. 4

[38] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *CVPR*, pages 3024–3033, 2021. 2

[39] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. In *ECCV*, 2020. 1, 2, 6

[40] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter. 1995. 2

[41] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, pages 3645–3649. IEEE, 2017. 1, 2, 6

[42] Jialian Wu, Jiale Cao, Liangchen Song, Yu Wang, Ming Yang, and Junsong Yuan. Track to detect and segment: An online multi-object tracker. In *CVPR*, pages 12352–12361, 2021. 2, 6

[43] Enze Xie, Jian Ding, Wenhai Wang, Xiaohang Zhan, Hang Xu, Peize Sun, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. In *ICCV*, pages 8392–8401, 2021. 3

[44] Yihong Xu, Yutong Ban, Guillaume Delorme, Chuang Gan, Daniela Rus, and Xavier Alameda-Pineda. Transcenter: Transformers with dense queries for multiple-object tracking. *arXiv preprint arXiv:2103.15145*, 2021. 6

[45] Ceyuan Yang, Zhirong Wu, Bolei Zhou, and Stephen Lin. Instance localization for self-supervised detection pretraining. In *CVPR*, pages 3987–3996, 2021. 3

[46] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, pages 2129–2137, 2016. 6

[47] En Yu, Zhuoling Li, Shoudong Han, and Hongwei Wang. Relationtrack: Relation-aware multiple object tracking with decoupled representation. *arXiv preprint arXiv:2105.04322*, 2021. 1, 2

[48] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *CVPR*, pages 2403–2412, 2018. 3

[49] Fangao Zeng, Bin Dong, Tiancai Wang, Cheng Chen, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. *arXiv preprint arXiv:2105.03247*, 2021. 6

[50] Yang Zhang, Hao Sheng, Yubin Wu, Shuai Wang, Wei Ke, and Zhang Xiong. Multiplex labeling graph for near-online tracking in crowded scenes. *IEEE Internet Things J.*, 7(9):7892–7902, 2020. 6

[51] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *Int J Comput Vis*, pages 1–19, 2021. 1, 2, 4, 5, 6, 8

[52] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, 2020. 2, 6

[53] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2, 3