Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 9959178, 11 pages
https://doi.org/10.1155/2023/9959178

Research Article

A Multiple Object Tracker with Spatiotemporal Memory Network

Peng Xiao,[1] **Jiannan Chi**,[2,3] **Zhiliang Wang**,[1] **Fei Yan**,[1] **and Jiahui Liu**[2]

[1]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China*
[2]*School of Automation and Electronic Engineering, University of Science and Technology Beijing, Beijing 100083, China*
[3]*Engineering Research Center of Intelligence Perception and Autonomous Control, Beijing University of Technology, 100124, China*

Correspondence should be addressed to Jiannan Chi; ustbjnc@ustb.edu.cn

Received 16 January 2023; Revised 10 March 2023; Accepted 10 September 2023; Published 26 October 2023

Academic Editor: Mohammad Tawfik

Target tracking is an important application of unmanned aerial vehicles (UAVs). The template is the identity of the target and has a great impact on the performance of target tracking. Most methods only keep the latest template of the target, which is intuitive and convenient but has poor ability to resist the change of target appearance, especially to reidentify a target that has disappeared for a long time. In this paper, we propose a practical multiobject tracking (MOT) method, which uses historical information of targets for better adapting to appearance variations during tracking. To preserve the spatial-temporal information of the target, we introduce a memory pool to store masked feature maps at different moments, and precise masks are generated by a segmentation network. Meanwhile, we fuse the feature maps at different moments by calculating the pixel-level similarity between the current feature map and the masked historical feature maps. Benefiting from the powerful segmentation features and the utilization of historical information, our method can generate more accurate bounding boxes of the targets. Extensive experiments and comparisons with many trackers on MOTS, MOT17, and MOT20 demonstrate that our method is competitive. The ablation study showed that the introduction of memory improves the multiobject tracking accuracy (MOTA) by 2.1.

1. Introduction

The unmanned aerial vehicle is one of the most interesting research directions at present and has a wide range of applications, such as monitoring disasters, long-range telecommunications, relays, Internet network services, and aerial surveillance. Multiple object tracking (MOT) is a key technology in the field of video surveillance. The task of MOT is largely divided into locating multiple objects, maintaining their identities, and yielding their individual trajectories. Depending on how the target is initialized, MOT methods can be categorized into two types: model-free tracking (MFT) and tracking-by-detection (TBD). The MFT framework requires manually initializing a certain number of targets and then tracking them. An obvious disadvantage of the MFT framework is that it cannot handle the case when new objects appear. The TBD framework detects targets at each frame and then matches the detected objects with existing trajectories. If a target matches a trajectory, the target is a known target. If a target fails to match any trajectory, the target is a new target and a new trajectory needs to be initialized. If a trajectory fails to match any target, the target represented by that trajectory is lost. The TBD framework is becoming increasingly popular because it is more flexible and accurate. We are also researching in the direction of TBD. Our method follows the TBD framework and gives deep insight into the multiobject tracking problems such as deformation, scale variation, and object occlusion.

Tracking performance of the TBD method is greatly influenced by the template, which separates the target from the background and other objects. Most tracking research works only keep an up-to-date template. But, the disadvantage of a single template is that it has a poor ability to capture the fast shape change of the target. If the appearance of the target has changed significantly, the current template would fail to match the previous one. Motivated by STMTrack [1], we extend this single-object tracking method to multiple objects by using robust deep network features for

the tracking targets to improve tracking performance. We also design a memory reading and writing strategy to dynamically model the moving target, and the dynamic model can capture long variation features about the moving target's appearance. Our goal is to design a simple and practical MOT method based on multiple templates. There are two core problems to be solved when introducing multiple templates: what to save in memory and how to read and write memory.

The first problem is what kind of features will be saved in memory to model the target. Previous research works use the deep features [2, 3], the residual map [4], and the original raw pixels [1] of the target. These features are suitable for SOT, because SOT only needs to separate the target from the background and does not pay much attention to spatial position changes. Mutual interference between targets in MOT is frequent, so it is necessary to consider the targets' spatial information. We use the global feature map of the complete image while suppressing the background features and highlighting the target features by adding a mask. It should be pointed out that our method is different from STMTrack. STMTrack is a SOT method, while ours is an MOT method which is more challenging. STMTrack stores the original image and the mask of the target. The features extracted by the detection network are nonreusable, so the tracker wastes time calculating the feature maps of the historical frame at each moment. We save masked feature maps in memory to avoid double deep neural network calculation. STMTrack uses a detection network to get a rectangle mask. We use a segmented network, so we can additionally get a segmentation mask and a minimum rectangle mask.

The second problem is how to model fast-moving targets. We design a dynamic model that can be updated by a reading and writing memory strategy. Yang and Chan [4] and Deng and Zheng [5] use long short-term memory network (LSTM) as a memory controller. But the overall performance of these trackers is greatly affected by the quality of the LSTM controller. The memory reading process is to fuse multiple templates into one template, and the key is to calculate the fusion weights of each template. Inspired by the work [1, 3], we calculate the pixel-level similarity between the feature map of the current frame and the historical frames as fusion weights. We use masked feature maps, instead of extracting the feature maps after masking the original image. Researches [6, 7] show that the target information from the first frame and the previous frame plays a significant role in the target location of the current frame. More templates mean more computation, so we limit the memory length. Additionally, we design a writing criterion so that only feature maps satisfying this criterion would be written into memory. It means that the length of memory is flexible.

In summary, the main contributions of the proposed method are as follows:

(1) We extend the STMTrack method from single-object to multiobject tracking which uses global deep feature maps with masks to highlight targets' positions. This strategy greatly enhances MOT tracking performance with only a single deep neural network computation

(2) We design a memory reading and writing strategy to keep the dynamic template model for moving targets. This dynamic model can represent moving objects with big appearance variations for a long time. And, this feature makes our multiobject tracking method more robust and flexible for occlusion and fast deformation

2. Related Work

2.1. Multiple Object Tracking. Multiple object trackers focus on tracking an unknown number of objects from a fixed set of categories. Thanks to the powerful detection capability of the object detector, the TBD framework has become the standard paradigm for most state-of-the-art multiple object trackers. It firstly uses detectors (i.e., such as Faster RCNN [8], SDP [9], and DPM [10]) to localize all objects in the video frames and then associates these objects together to form trajectories [11, 12]. Due to the fact that detector performance can significantly influence tracking results, some MOT datasets, such as MOT Challenge [13–15], provide a standard set of detections [8–10] to allow the performance of different trackers to be fairly compared. For this reason, the majority of MOT methods have been focusing on improving association. The Hungarian method [16] assigns the identity label to each detection in every frame. Due to its high speed, it has been widely used in the past decade [17]. Hungarian is, however, a local optimal method, and its accuracy is not excellent. Some researchers have tried to improve the association process with deep learning models, such as RNN [18, 19], deep multilayer perceptron [20], and deep reinforcement learning agents [21, 22]. Milan et al. [14] used an RNN to predict the probability of the existence of a track in each frame. This helped with the decision of when to initiate or terminate the track. Kieritz et al. [20] took the track score at the previous step and various kinds of information (like association score and detection confidence) about the last associated detection as inputs to an MLP with two hidden layers, and the output track confidence score was then utilized to manage the termination of tracks. Rosello and Kochenderfer [21] used multiple deep RL agents to manage the various tracked targets, deciding when to start and stop tracks and influencing the operation of the Kalman filter. Training a detection network and an association network separately is very computationally intensive and slow in inference, so some works [23–25] try to build a unified multitarget tracking framework. It is becoming increasingly popular to use transformer architecture [25] for visual object tracking [26–28]. To achieve this, an encoder with self-attention is first adopted, and then, a decoder with cross-attention replaces the learnable query features with the previously detected object features. Although it has proven to be powerful and inspiring, the transformer-based approach falls far short of real-time requirements.

2.2. Memory Networks. Because many tasks in natural language processing (NLP) are very concerned with continuity, memory networks were first proposed in the NLP field, such as question answering [29], dialogue systems [30], and text

generation [31]. They are composed of common neural networks and memory components that store historical information. Subsequently, memory networks were introduced to other fields with success, such as shot learning [32, 33], video object segmentation [7, 32], and action recognition [34, 35].

Recently, some tracking methods also use memory networks to enhance the representation ability of objects. MemTrack [13] reads a residual template from the memory network and combines it with the initial template to yield a synthetic template as an updated representation of the target. Later, an improved version was proposed [36]. A negative memory unit is used that stores distractor templates to cancel out wrong responses from the object template, and an auxiliary classification loss is designed to facilitate the tracker's robustness to appearance changes. The reading operation is controlled by an LSTM controller, so the quality of the LSTM will greatly affect overall performance. STMTrack [1] retrieves historical information with the guidance of the current frame and adaptively gets all it needs. SAMN [2] exploits an appearance memory network to capture stable appearance information and a spatial memory network to extract position information. These works focus on single-object tracking (SOT), and the following multiobject tracking (MOT) methods are also worthy of attention. MeMOT [37] refers to the encoder and decoder structures of the transformer. The encoder extracts the core information from the memory for each tracked object. The decoder solves the object detection and data association tasks. MeToS [3] first creates tracklets using instance segmentation and optical flow and then associates the tracklets with a space-time memory network. Inspired by STMTrack [1], our method stores masked feature maps in memory and calculates the pixel-level similarity between the feature maps between the current frame and historical frames. We do not simply transfer this SOT method to the MOT method but improve the utilization of the target's features. Firstly, we save the feature maps instead of the original images. Secondly, we use the mask generated by segmentation instead of the rectangular box. Finally, we directly calculate the similarity between feature maps, instead of extracting the features of each target and then calculating the similarity, which maximizes the exploit of the feature extraction network.

2.3. Instance Segmentation and Tracking. Instance segmentation is closely related to object detection. A classical framework is adding a parallel branch for predicting object masks with bounding boxes to the detection branch. Because it usually provides a more accurate representation of the target, segmentation-based tracking has become increasingly popular. Many multiobject tracking methods adopt the MOTS paradigm, which creates tracklets from segmentation masks and then builds long-term tracks by merging the tracklets [38]. As a result of all their observations and hierarchical clustering, ReMOTS [38] associates tracklets with those that are temporally close and have similar appearance features without any temporal overlap. Choudhuri et al. [39] compute the top-k detection assignments between the consecutive frame pairs using the Hungarian-Murty algorithm [40], then use dynamic programming to obtain a globally optimal minimum-cost path, and finally uncover long-range assignments in a postprocessing step via an assignment formulation over the space of the track. TrackRCNN [23] extends Mask R-CNN by an association head to return an embedding for each detection. MOTSNet [41] proposes a mask pooling layer to Mask R-CNN [42] to improve object association over time. MeNToS [3] associates temporally close segmentation masks between consecutive frames by computing the optical flow; then, an appearance similarity is computed by a memory network. Finally, the tracklets are gradually merged starting with the pair having the highest similarity while respecting the updated constraints. Mask R-CNN [42] is the most used instance segmentation method in MOTS [38, 43]. Some trackers [23, 41] have been improved to make them more suitable for tracking tasks. TrackRCNN [23] extends Mask R-CNN with 3D convolutions to incorporate temporal information and extracts instance embeddings for tracking by ROIAlign. MOTSNet [41] adds a tracking head (TH) that runs in parallel with the region segmentation head (RSH). Other segmentation methods are also used. GridShift [44] presents an extended variant of mean shift, which speeds up by employing a grid-based approach to neighbor search and moving the active grid cells in place of data points toward higher density. D3S2 [45] applies two target models with complementary geometric properties, one invariant to a broad range of transformations, including nonrigid deformations, and the other assuming a rigid object, to simultaneously achieve robust online target segmentation. Our work adopts the TBD paradigm, focusing on how to make full use of segmentation features. To retain location information, we do not use the binary classification features in the segmentation branch, but the features output from the backbone network.

3. Proposed Method

We present a practical multiobject tracking approach equipped with a memory network. In this section, we first give an overall introduction to our algorithm based on the architecture graph (Figure 1) and then describe each module in detail.

3.1. Architecture. Given a sequence of video frames, the goal of MOT is to localize a set of objects while following their trajectories. Our tracking framework is based on the TBD paradigm, in which objects are first detected in each frame and then associated over time to form trajectories. In actual processing, we generally predict the targets $P = \{P^1, P^2, \cdots, P^M\}$ in the current frame and then match the predicted targets with the detected targets $D = \{D^1, D^2, \cdots, D^N\}$. M and N do not exceed a set threshold. The essence of this problem is $M \times N$ bipartite graph matching. Different from most existing methods that use a fixed template or the latest template to predict the current state of objects, we use multiple feature maps of each target in the memory pool. Now, our issue is extended to a more complex $(M \times L) \times N$ graph matching problem, where L is the memory length. For simplicity, we fuse multiple templates of each target into a synthesized

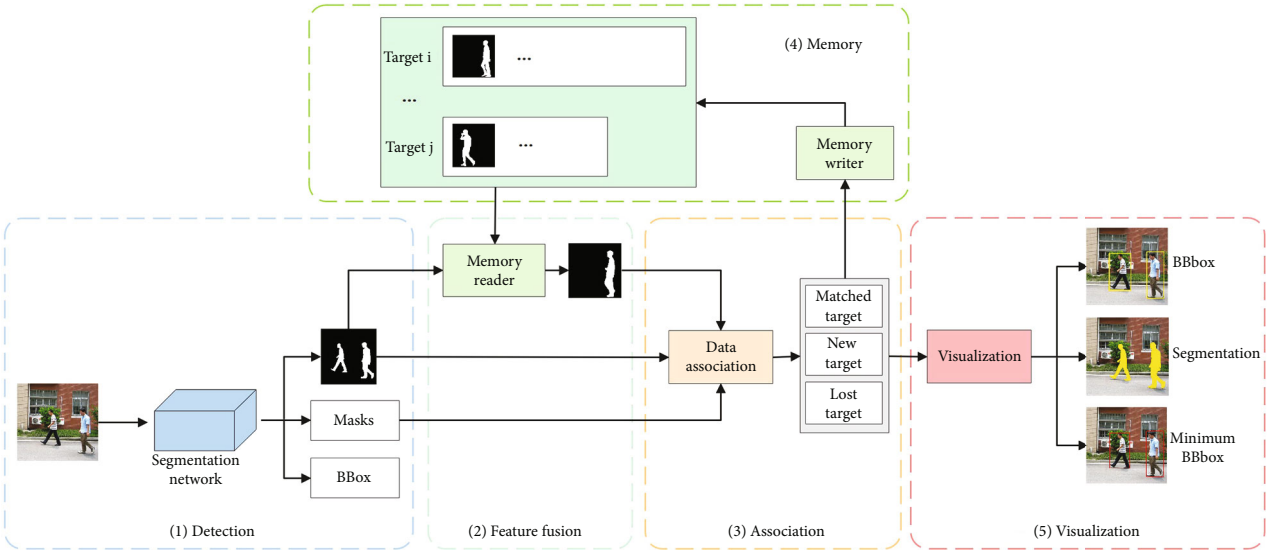

Figure 1: The architecture of our proposed method.

template, which is also a common practice of object trackers with memory networks.

Specifically, as shown in Figure 1, the architecture consists of five components: (1) a memory module maintains a flexible number of masked feature maps for each target. (2) An object detection module detects objects in the current frame with a segmentation network and provides their feature maps, masks, and bounding boxes. Notice that the feature map is the multichannel feature of the whole image rather than the feature of each target, and the mask is a binary map for each target. (3) A feature fusion module fuses multiple masked feature maps of every target into one feature map according to the feature map of the current frame. This part is the key point that distinguishes our method from single template methods. Detailed process will be described in Section 3.2. (4) A data association module matches the existing objects with the detected objects. (5) A visualization module shows tracking results with the type of segmentation, vertical bounding boxes, or tilted bounding boxes.

3.2. Segmentation-Based Detection. We use Mask R-CNN [42] with a feature pyramid network (FPN) [46] backbone as the segmentation network, shown in Figure 2. Firstly, the FPN extracts multiple feature maps of the same size from the original image. Then, the region proposal network (RPN) takes the feature maps as input and outputs a set of rectangular object proposals. Next, the ROIAlign layer ensures alignment between the RoI and extracted features. Finally, a mask branch network calculates the mask of each target, and the predicted branch network calculates the bounding box and category. The network outputs three results: the feature maps of the input image, the mask, and the bounding box coordinates of each target. To preserve spatial location information, we use the feature maps of the whole map, rather than the feature maps of the ROI region. Meanwhile, different masks are used on the feature maps to discriminate each target from its surroundings.

3.3. Feature Fusion. As mentioned in Section 3.1, we need to fuse the historical feature maps with the same size as the feature maps of the current frame. The usual way is to calculate the weight at different moments, the weight of different channels, or the weight at different pixels. Since we use high-precision segmentation features, we chose the last approach to fully utilize them. Figure 3 illustrates the feature fusion process of a target. The historical feature maps are denoted as $f \in R^{THW\times C}$, where T is the number of memory frames, C is the number of channels, and H and W represent the height and width of the feature map, respectively. The feature maps of the current frame are denoted as $q \in R^{C\times HW}$. For the convenience of matrix multiplication in math, we reshape f from $T\times C\times H\times W$ to $THW\times C$ and reshape q from $C\times H\times W$ to $C\times HW$; thus, here, $THW = T\times H\times W$ and $HW = H\times W$.

Inspired by [47], we compute the similarity between every pixel of the historical feature maps f in memory and every pixel of the current feature maps q to obtain a similarity matrix $w \in R^{THW\times HW}$. The similarity of the i-th pixel on f and the j-th pixel on q is computed as follows:

$$w_{ij} = \frac{\exp\left[\left(f_i \odot q_j\right)/s\right]}{\sum_{\forall k}\exp\left[\left(f_k \odot q_j\right)/s\right]}, \tag{1}$$

where the binary operator $\odot$ denotes vector dot product. Following [25], to prevent the exp function from overflowing numerically, we add a scaling factor s. Here, s is set to $\sqrt{C}$, where C is the feature dimensionality of f. Then, the i-th element of g is a weighted sum of the i-th element of each historical feature map:

$$g_i = (f_i)^T \otimes w, \tag{2}$$

where $(f_i)^T \in R^{C\times THW}$ is the transpose of f_i and $\otimes$ denotes

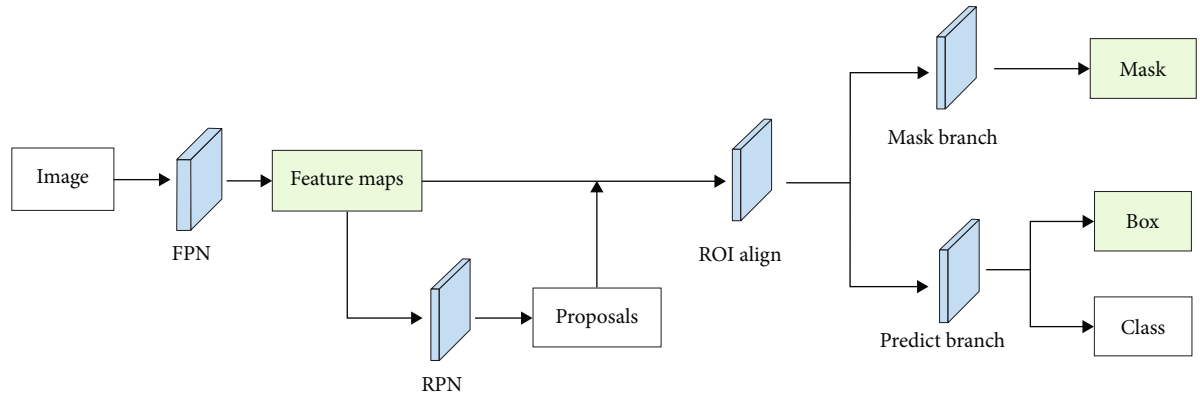

Figure 2: The architecture of segmentation network. Light green boxes (feature maps, mask box) are the results to be output.

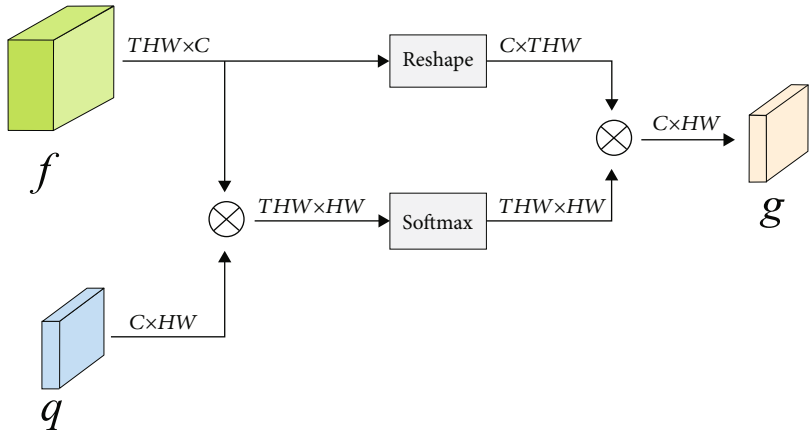

Figure 3: The process of feature fusion.

matrix multiplication. Obviously, g has the same dimensions with q.

3.4. Data Association. Here, we adopt the simple but practical Hungarian algorithm [48] to find the association between the detected targets and the known targets. Notice that we obtain feature maps of the whole frame and masks for every target in the detection step. So the feature maps of each target P^i can be easily obtained by multiplying the feature maps of the current frame f with the target's mask. There are three results for matching. (1) For a matched target, if a detected target matches an existing target, we consider it as a known target and update the feature maps in memory. (2) For a new target, we consider a detected target to be a new target if it does not match any existing target. (3) For a missing target, if the existing target does not match any detected target, we consider the target to be lost and retain its template for a period of time. Next, we will update the memory based on the matched results as well as display the tracking results.

3.5. Memory Update. In general, the more templates you use, the more feature information of the target will be available, but the more computation you will need to do. Memory length design and memory writing control are challenging issues. For each target, we store up to L feature maps. According to existing work [49, 50], the first frame information and previous frame information play an important role in determining the current frame's target location. Specifically, the target from the first frame provides the most reliable information, and the previous frame has the most similar appearance to the target in the current frame. For the left $L-2$ feature maps, we tend to choose features that are salient, so we limit the similarity, and only feature maps with a similarity less than a threshold are retained in the memory. Our memory-updating strategy is summarized below. When a new target appears, we save its masked feature map in memory. For subsequent frames, we first save its masked feature maps, then calculate the similarity of the feature maps between each intermediate frame and the initial frame and the current frame, and sort these similarities in descending order. If the similarity is greater than a threshold, the feature maps at that moment are removed. If the memory size is exceeded, the earliest feature maps are discarded (except for the initial frame). If a target is lost in K consecutive frames, all its feature maps will be deleted.

3.6. Visualization. Instance segmentation is suitable for detecting targets with variational shapes. In this paper, we use the high-precision Mask R-CNN as the segmentation network. It is easy to get the minimum bounding box of the target according to the mask. Most MOT methods only output vertical rectangular bounding boxes, but our tracker

can also provide tilted boxes and masked targets. Figure 4 shows the three styles of tracking results. Clearly, the results in the last column show tilted targets. When the shape of the target changes only slightly, it is reasonable to use the minimum bounding box to approximate the rotation angle of the target. If the shape of the target changes sharply, such as gymnasts, it is meaningless to estimate the rotation angle because of the lack of a comparison basis.

4. Experiments

In this section, we present the tracking results of our method on two common and challenging MOT benchmarks, namely, MOT17 [14] and MOT20 [15].

4.1. Experimental Setup

4.1.1. Dataset. The MOT17 [14] and MOT20 [15] are the most popular benchmarks for multiple object tracking, which mainly focus on the task of pedestrian tracking in dense scenes. The MOT17 [14] contains 7 scenes of indoor and outdoor public places. The video of each scene is separated into train set and test set. To exclude the influence of different detectors on the tracking results, three sets of publicly available detections with varying quality are provided, namely, DPM [10], Faster R-CNN [8], and SDP [9]. MOT20 contains 8 new scenes with much more crowded targets, and the types of targets are not only pedestrians but also cars, bicycles, etc. There is only one public detector available: Faster R-CNN. MOTS [23] is a multitarget tracking and segmentation dataset, with only two types of targets, pedestrians and cars, with a total of 8 scenes.

4.1.2. Metrics. The MOT index uses the CLEAR index [51] and other track quality indexes [52] to evaluate tracking performance, such as MOTA, IDF1, MT, ML, FP, FN, IDS, and Hz. MT indicates mostly tracked targets and measures the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span. ML indicates mostly lost targets and measures the ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life span. FP indicates the total number of false positives. FN indicates the total number of false negatives (missed targets). IDS indicates the number of identity switches. MOTA indicates multiobject tracking accuracy, which is the result of comprehensive FP, FN, and IDS. IDF1 indicates the IDF1 score, which measures the ratio of correctly identified detections over the average number of ground-truth and computed detections measured. In these metrics, MOTA and IDF1 are generally considered to be the most significant. The commonly used evaluation indexes of MOTS include sMOTSA, MOTSA, and MOTSP. sMOTSA (mask-based soft multiobject tracking accuracy), which is a soft version of MOTSA, accumulates the mask overlaps of true positives instead of only counting how many masks reach an IoU of more than 0.5. MOTSA indicates mask-based multiobject tracking accuracy, which is a variant of MOTA, evaluated based on mask overlap (mask IoU). MOTSP indicates mask overlap-based variant of multiobject tracking precision, which is a variant of MOTP, evaluated based on mask IoU instead of bounding box IoU.

4.1.3. Training. We use an FPN background to Mask R-CNN and pretrain it on COCO [53] and Mapillary [54]. To increase the diversity of training examples, we adopt a random radiation transformation strategy for data augmentation. The translation is randomly performed from $-0.15\ S$ to $0.15\ S$, and the resizing scale varies between [0.8,1.2]. Here, S is the cropping size, and we set S to the scale of 4 times the target bounding box. The net was trained for 40 epochs with 1000 iterations per epoch, and the learning rate set of ADAM is set to 10^{-3} and with 0.2 decay every 15 epochs. We implement the segmentation net in PyTorch and train it on 2 GeForce RTX 3090 Ti GPUS. Each batch has 96 images (each GPU holds 48 images).

4.2. Benchmark Results

4.2.1. MOTS. We first evaluate our method on MOTS [23], and the results are shown in Table 1. The selected trackers are published and peer-reviewed. The symbols "↑" and "↓," respectively, mean the higher the better and the smaller the better. The top two best results are highlighted in bold. The performance of our method is not outstanding, and the most critical metric, sMOTA, scores only 61.6, which ranks in the middle. The intrinsic reason is that MOTS metrics measure the combined effect of segmentation and tracking, and most MOTS methods focus on improving the quality of segmentation and tracklet association. In contrast to these trackers, our motivation is to take advantage of multiple historical feature maps, focusing on memory reading and writing. Thus, we use the Mask R-CNN method and the Hungarian algorithm as the basic models of segmentation and data association. Our method is naturally unfeasible in comparison to these well-designed MOTS methods. For the subsequent analysis, we will compare it with the methods in the MOT Challenge.

4.2.2. MOT17. The most important difference between the MOTS methods and the MOT methods is that the MOTS methods use a mask image to represent the target, while the MOT methods use a rectangle image. Clearly, it is easy to get the rectangle bounding box according to the mask for a MOTS method, so the MOT metrics can also be applied to MOTS methods. At first, we evaluated our tracker against state-of-the-art for multiple objects tracking on the MOT17 dataset. For a fair comparison, we chose methods that use private detectors and have been published in MOT17. The tracking performance is shown in Table 2. We provide two sets of results of our method, one using the segmentation mask (ours_seg) and the other using the rectangle mask (ours_rec). As we can see, our method achieves the best and the second best score in most metrics. Although the speed of our method is not the fastest, considering that we use the feature map of multiple images, this speed is still satisfactory.

4.2.3. MOT20. Under the same experimental conditions, we evaluated our method on MOT20, and the results are shown

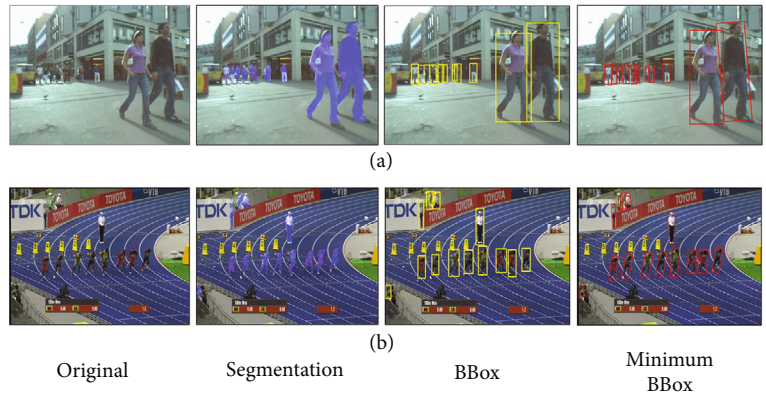

Figure 4: Three styles of tracking results with our proposed method in different scenes. The columns from left to right are the original image, segmentation result, bounding box, and minimum bounding box.

Table 1: Tracking results on MOTS.

Tracker	sMOTA↑	IDF1↑	MOTSA↑	MOTSP↑	IDS↓
ReMOTSv2 [55]	**70.4**	**75.8**	**84.4**	84.0	**229**
EMNT [56]	**70.0**	**77.0**	83.7	**84.1**	261
MAF_HDA [57]	69.9	67.0	**83.8**	84.1	401
OPITrack [58]	63.5	45.4	75.5	**84.6**	342
MPNTrackSeg [59]	58.6	68.8	73.7	80.6	**202**
SORTS [60]	55.0	57.3	68.3	81.9	552
TrackRCNN [23]	40.6	42.4	55.2	76.1	567
Ours	61.6	62.3	74.9	82.2	256

Table 2: Tracking results using private detectors on MOT17.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
TransCenter [61]	**73.2**	62.2	960	435	**23112**	**123738**	4614	1.0
LMOT [62]	72.0	70.3	**1068**	**408**	28113	126704	3071	28.6
MOTPrivate [61]	70.0	62.1	915	480	28119	136722	4647	1.0
TraDeS [63]	69.1	63.9	858	507	**20892**	150060	3555	**66.9**
LCC [64]	68.8	70.2	960	417	38457	135006	**2805**	0.9
XJTU [65]	68.2	64.6	924	570	24747	152388	**2262**	**99.2**
ours_seg	**72.3**	**70.8**	**986**	**394**	24979	**126584**	4738	26.5
ours_rec	72.1	**70.4**	967	412	25135	127404	4801	26.6

in Table 3. Similar to the results on MOT17, our tracker achieves high scores in most metrics except for IDS and Hz. As the targets in MOT20 are more crowded and harder to track, it is normal that the scores on MOT20 are less impressive. The MOTA scores of {TransCenter [61], LMOT [66], MOTPrivate [61], ours_rec, ours_seg, XJTU [65], LCC [64]} decrease by {14.7, 12.9, 12.7, 5.7, 5.4, 3.9, 2.8}, respectively.

4.3. Ablation Studies. To further explore the characteristics of our approach, we conducted more experiments on MOT17 with ours_seg.

4.3.1. Memory Length and Template Filtering. To explore the impact of memory size and template filtering on tracking, we conducted multiple experiments on MOT17. We tested two models with different memory writing methods, filtering similar templates (ours_seg) and not filtering similar templates (ours_seg_nf). Five sets of experiments were carried out for each model, and the memory size ranged from 1 to 5. If the memory size is 1, we only save the feature map of the latest frame. If the memory size is 2, we save the feature maps of both the initial frame and the latest frame. If the memory size is greater than 2, the ours_seg model filters similar templates, but ours_seg_nf does not filter similar

Table 3: Tracking results using private detectors on MOT20.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
LCC [64]	66.0	67.0	699	165	43938	129584	**2237**	0.2
RekTCL [66]	65.2	**70.1**	761	**131**	61209	**114709**	4139	22.4
XJTU [65]	64.3	66.6	626	174	40780	140565	3379	**33.0**
LMOT [62]	59.1	61.1	312	286	**13526**	196673	**1398**	22.4
TransCenter [61]	58.5	49.6	603	185	64217	146019	4695	1.0
MOTPrivate [61]	57.3	46.7	444	223	42271	173903	5014	1.0
ours_seg	**66.9**	**69.7**	**821**	**154**	**37297**	**128974**	5075	20.3
ours_rec	**66.4**	68.5	**836**	161	38315	130160	5124	20.2

Table 4: The performance of memory length and template filtering on MOT17.

		1	2	3	4	5
ours_seg	MOTA↑	70.2	71.6	72.3	72.4	72.1
	FPS↑	35.1	29.5	26.5	21.6	15.9
ours_seg_nf	MOTA↑	70.2	71.6	72.4	72.3	72.2
	FPS↑	35.1	29.5	22.5	15.2	10.1

Table 5: The performance on segmentation mask and rectangle mask.

Dataset	Tracker	MOTA↑	FP↓	FN↓	IDS↓
MOT17	ours_seg	72.3	24979	126584	4738
	ours_rec	72.1	25135	127404	4801
MOT20	ours_seg	66.9	37297	128974	5075
	ours_rec	66.4	38315	130160	5124

templates and keeps the most recent few templates. MOTA and FPS are used as metrics for accuracy and speed, respectively. As shown in Table 4, increasing the number of reference frames does not always improve the MOTA score, but it does decrease speed. When the memory size exceeds 2, the MOTA score barely changes, so the optimal value of memory length is 3. By comparing the results of ours_seg and ours_seg_nf, we can see that filtering similar templates has no significant effect on MOTA but can significantly improve tracking speed. It should be noted that in multitarget tracking, new targets may appear at any moment, so the number of templates for each target is not strictly equal. This means that memory size limits the maximum number of templates, and these metrics are statistical values.

4.3.2. Segmentation Mask vs. Rectangle Mask. Because global feature maps are used, in order to avoid introducing too much background information, we exploit masks to highlight the target. We design two kinds of masks, the segmentation mask and the rectangle mask. Segmentation masks filter foreground information, and rectangle masks filter local information containing background information. Which one is better? Table 5 shows the results. As we can see, the segmentation mask performs slightly better than the rectangle mask. Compared with the results in Table 4, the influence of background on feature maps is not very significant.

4.3.3. Matching Strategy of Tracking Boxes. In the association step of the TBD-based MOT method, the detected objects are matched with known targets. The known targets are usually the targets in the previous frame or the predicted targets in the current frame. We set up three groups of experiments. The first group is ours_rec, which locates the target in the current frame using historical information. The second group predicts the target in the current frame with its bounding box in the previous frame. The third group just used the target in the previous frame. Table 6 shows the results of different tracking boxes on MOT17. It can be seen that our method is better than the "current" group, and the "current" group is better than the "previous" group. This shows that our method is much more accurate at predicting the targets' location.

4.3.4. What Is the Bottleneck of Accuracy? In Section 4.2, we notice that our method has a higher IDS score than other methods, while we get the second-best MOTA score. With the high accuracy of the segmentation network, we have a lower probability of identifying the target incorrectly, thus lowering the number of false positives (FP) and false negatives (FN). Table 7 shows the statistics of false recognition.

Table 6: The performance of matching strategy of tracking boxes.

Matching	MOTA↑	FP↓	FN↓	IDs↓
ours_rec	72.1	25135	127404	4801
Current	68.2	26064	148556	4986
Previous	67.3	27789	151734	5062

Table 7: Comparison of false recognition on MOT17.

Tracker	MOTA↑	FP+FN↓	IDS↓
TransCenter [61]	**73.2**	**146850**	4614
LMOT [62]	72.0	154817	3071
MOTPrivate [61]	70.0	164841	4647
TraDeS [63]	69.1	170952	3555
LCC [64]	68.8	173463	**2805**
XJTU [65]	68.2	177135	**2262**
ours_seg	**72.3**	**151563**	4738
ours_rec	72.1	152539	4801

IDS is undoubtedly the bottleneck of multiobject tracking accuracy (MOTA) in our method. In other words, data association greatly affects overall performance. Since we use the Hungarian algorithm [48], there is much room for improvement in the future.

4.4. Discussion. In this section, we evaluate our approach to the popular MOT Challenge dataset. While our overall performance in MOTS was poor, our trials on MOT17 and MOT20 showed that our approach was competitive. According to the ablation study, the following conclusions can be drawn: (1) Memory contributes the most to tracking. From Tables 4 and 6, we can see that when we use the memory network, MOTA will improve by about 3 points. This indicates that the fused feature can better represent the target than the original feature. Using a split or rectangular mask has an insignificant impact on overall performance. (2) Tracking performance is not always better when the length of memory is longer. Because the mask contains the position of the target in the image, too many reference frames will introduce a large amount of outdated spatial information. This will adversely affect the estimation of the target position. (3) Data association is the bottleneck. Our starting point is to design a mechanism that can use target information in multiple frames. Therefore, in terms of data association, we adopt the basic method without further optimization, which has a significant impact on the performance of multitarget tracking.

5. Conclusion

The aim of the present research is to improve the adaptability of objects to template changes, thereby improving multiobject tracking performance. Our solution is to introduce dynamic multiple templates, the core of which is the reading and writing of memory. We store high-precision masked feature maps in memory, which are obtained from a segmentation net. Feature fusion is performed by computing the pixel-level similarity between the historical feature maps and the feature maps of the current frame. To alleviate the meaningless computation caused by similar features, we filter the feature maps in memory. Experiments on MOTS, MOT17, and MOT20 demonstrate that tracking is conducive to the number of templates increasing. But more templates would slow down tracking and introduce more unimportant target information that leads to poor tracking performance. How to select some representative templates from long time frames should be a critical issue in multitarget tracking research. Our work provides a way to exploit multiple templates in MOT and contributes to a better understanding of the relationship between tracking and templates. The major limitation of this study is the data association step. We use the basic Hungarian algorithm to solve this problem. Integration of data association into an end-to-end neural network and solving the tracking problem as a whole would be a good idea. It would be our future work focus.

Data Availability

The image datasets used to support the findings of this study can be downloaded from the public website whose links are provided in this paper.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of Beijing Municipality (Grant No. 4212023); the National Science Foundation for Young Scientists of China (Grant No. 62206016); the Foundation of Engineering Research Center of Intelligence Perception and Autonomous Control, Ministry of Education, P. R. China; and the Guangdong Basic and Applied Basic Research Foundation.

References

[1] Z. Fu, Q. Liu, Z. Fu, and Y. Wang, "STMTrack: Template-free visual tracking with space-time memory networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13774–13783, 2021.

[2] F. Xie, W. Yang, K. Zhang, B. Liu, G. Wang, and W. Zuo, "Learning spatio-appearance memory network for high-performance visual tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2678–2687, 2021.

[3] M. Miah, G.-A. Bilodeau, and N. Saunier, "Multi-Object Tracking and Segmentation with a Space-Time Memory Network," 2021, http://arxiv.org/abs/2110.11284.

[4] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 152–167, Munich, Germany, 2018.

[5] Y. Deng and H. Zheng, "Memory network for tracking with deep regression," in *2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS)*, pp. 273–278, Sophia Antipolis, France, 2018.

[6] P. Voigtlaender, J. Luiten, P. H. Torr, and B. Leibe, "Siam R-CNN: Visual tracking by re-detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6578–6588, 2020.

[7] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9226–9235, Seoul, South Korea, 2019.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in neural information processing systems*, vol. 28, Curran Associates, Inc., 2015.

[9] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2129–2137, Las Vegas, NV, USA, 2016.

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[11] Z. Pi, H. Qin, C. Gao, and N. Sang, "Jointly detecting and multiple people tracking by semantic and scene information," *Neurocomputing*, vol. 412, pp. 244–251, 2020.

[12] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, pp. 3069–3087, 2021.

[13] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," 2015, http://arxiv.org/abs/1504.01942.

[14] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A benchmark for multi-object tracking," 2016, http://arxiv.org/abs/1603.00831.

[15] P. Dendorfer, H. Rezatofighi, A. Milan et al., "MOT20: A benchmark for multi object tracking in crowded scenes," 2020, http://arxiv.org/abs/2003.09003.

[16] V. K. Singh, B. Wu, and R. Nevatia, "Pedestrian tracking by associating tracklets using detection residuals," in *2008 IEEE Workshop on Motion and video Computing*, pp. 1–8, Copper Mountain, CO, USA, 2008.

[17] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5620–5629, Honolulu, HI, USA, 2017.

[18] C. Ma, C. Yang, F. Yang et al., "Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for multiple object tracking," in *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, San Diego, CA, USA, 2018.

[19] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[20] H. Kieritz, W. Hubner, and M. Arens, "Joint detection and online multi-object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1459–1467, Salt Lake City, UT, USA, 2018.

[21] P. Rosello and M. J. Kochenderfer, "Multi-agent reinforcement learning for multi-object tracking," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 1397–1404, Stockholm, Sweden, 2018.

[22] K. Lin, S. Wang, and J. Zhou, "Collaborative deep reinforcement learning," 2017, http://arxiv.org/abs/1702.05796.

[23] P. Voigtlaender, M. Krause, A. Osep et al., "MOTS: Multi-object tracking and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7942–7951, Long Beach, CA, USA, 2019.

[24] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, "Towards real-time multi-object tracking," in *Computer Vision–ECCV 2020: 16th European Conference*, Springer, Cham, Glasgow, UK, 2020.

[25] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[26] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8126–8135, 2021.

[27] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8844–8854, New Orleans, LA, USA, 2022.

[28] F. Ma, M. Z. Shou, L. Zhu et al., "Unified transformer tracker for object tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8781–8790, New Orleans, LA, USA, 2022.

[29] C. Xiong, S. Merity, and R. Socher, "Dynamic memory networks for visual and textual question answering," in *Proceedings of The 33rd International Conference on Machine Learning*, pp. 2397–2406, Anaheim, California, USA, 2016.

[30] C.-S. Wu, R. Socher, and C. Xiong, "Global-to-local memory pointer networks for task-oriented dialogue," 2019, http://arxiv.org/abs/1901.04713.

[31] X. Yi, M. Sun, R. Li, and Z. Yang, "Chinese poetry generation with a working memory model," 2018, http://arxiv.org/abs/1809.04306.

[32] X. Lu, W. Wang, M. Danelljan, T. Zhou, J. Shen, and L. V. Gool, "Video object segmentation with episodic graph memory networks," in *Computer Vision–ECCV 2020: 16th European Conference*, Glasgow, UK, 2020.

[33] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1842–1850, Anaheim, California, USA, 2016.

[34] C. Y. Wu, C. Feichtenhofer, H. Fan, K. He, P. Krahenbuhl, and R. Girshick, "Long-term feature banks for detailed video understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 284–293, Long Beach, CA, USA, 2019.

[35] M. Xu, Y. Xiong, H. Chen et al., "Long short-term transformer for online action detection," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1086–1099, 2021.

[36] T. Yang and A. B. Chan, "Visual tracking via dynamic memory networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 360–374, 2019.

[37] J. Cai, M. Xu, W. Li et al., "MeMOT: multi-object tracking with memory," in *Proceedings of the IEEE/CVF Conference on*

Computer Vision and Pattern Recognition, pp. 8090–8100, New Orleans, LA, USA, 2022.

[38] F. Yang, X. Chang, C. Dang et al., "ReMOTS: Self-supervised refining multi-object tracking and segmentation," 2020, http://arxiv.org/abs/2007.03200.

[39] A. Choudhuri, G. Chowdhary, and A. G. Schwing, "Assignment-Space-based Multi-Object Tracking and Segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13598–13607, 2021.

[40] K. G. Murty, "An algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, no. 3, pp. 682–687, 1968.

[41] L. Porzi, M. Hofinger, I. Ruiz, J. Serrat, S. R. Bulo, and P. Kontschieder, "Learning multi-object tracking and segmentation from automatic annotations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6846–6855, 2020.

[42] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961–2969, Venice, Italy, 2017.

[43] J. Luiten, I. E. Zulfikar, and B. Leibe, "UnOVOST: Unsupervised offline video object segmentation and tracking," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2000–2009, Snowmass Village, CO, USA, 2020.

[44] A. Kumar, O. S. Ajani, S. Das, and R. Mallipeddi, "GridShift: A Faster Mode-seeking Algorithm for Image Segmentation and Object Tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8131–8139, New Orleans, LA, USA, 2022.

[45] A. Lukežič, J. Matas, and M. Kristan, "A Discriminative Single-Shot Segmentation Network for Visual Object Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 9742–9755, 2021.

[46] T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, Honolulu, HI, USA, 2017.

[47] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, Salt Lake City, UT, USA, 2018.

[48] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[49] M. Muller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, "Trackingnet: A large-scale dataset and benchmark for object tracking in the wild," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 300–317, Munich, Germany, 2018.

[50] G. Wang, C. Luo, X. Sun, Z. Xiong, and W. Zeng, "Tracking by instance detection: A meta-learning approach," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6288–6297, 2020.

[51] R. Stiefelhagen, K. Bernardin, R. Bowers, J. Garofolo, D. Mostefa, and P. Soundararajan, "The CLEAR 2006 evaluation," in *Multimodal Technologies for Perception of Humans. CLEAR 2006. Lecture Notes in Computer Science, vol 4122*, R. Stiefelhagen and J. Garofolo, Eds., pp. 1–44, Springer, Berlin, Heidelberg, 2007.

[52] B. Wu and R. Nevatia, "Tracking of multiple, partially occluded humans based on static body part detection," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pp. 951–958, New York, NY, USA, 2006.

[53] T. Y. Lin, M. Maire, S. Belongie et al., "Microsoft COCO: Common Objects in Context," in *Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., pp. 740–755, Springer, Cham, 2014.

[54] G. Neuhold, T. Ollmann, S. Rota Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4990–4999, Venice, Italy, 2017.

[55] F. Yang, Z. Wang, Y. Wu, S. Sakti, and S. Nakamuram, "Tackling multiple object tracking with complicated motions—Redesigning the integration of motion and appearance," *Image and Vision Computing*, vol. 124, article 104514, 2022.

[56] S. Wang, H. Sheng, D. Yang, Y. Zhang, Y. Wu, and S. Wang, "Extendable multiple nodes recurrent tracking framework with RTU++," *IEEE Transactions on Image Processing*, vol. 31, pp. 5257–5271, 2022.

[57] Y. M. Song, Y. C. Yoon, K. Yoon, H. Jang, N. Ha, and M. Jeon, "Multi-object tracking and segmentation with embedding mask-based affinity fusion in hierarchical data association," *IEEE Access*, vol. 10, pp. 60643–60657, 2022.

[58] Y. Gao, H. Xu, Y. Zheng, J. Li, and X. Gao, "An Object Point Set Inductive Tracker for Multi-Object Tracking and Segmentation," *IEEE Transactions on Image Processing*, vol. 31, pp. 6083–6096, 2022.

[59] G. Brasó, O. Cetintas, and L. Leal-Taixé, "Multi-Object Tracking and Segmentation Via Neural Message Passing," *International Journal of Computer Vision*, vol. 130, no. 12, pp. 3035–3053, 2022.

[60] M. Ahrnbom, M. Nilsson, and H. Ardö, "Real-time and online segmentation multi-target tracking with track revival re-identification," in *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2021.

[61] Y. Xu, Y. Ban, G. Delorme, C. Gan, D. Rus, and X. Alameda-Pineda, "Transcenter: Transformers with dense queries for multiple-object tracking," 2021, http://arxiv.org/abs/2103.15145.

[62] R. Mostafa, H. Baraka, and A. Bayoumi, "LMOT: Efficient Light-Weight Detection and Tracking in Crowds," *IEEE Access*, vol. 10, pp. 83085–83095, 2022.

[63] J. Wu, J. Cao, L. Song, Y. Wang, M. Yang, and J. Yuan, "Track to detect and segment: An online multi-object tracker," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12352–12361, 2021.

[64] Z. Zou, J. Huang, and P. Luo, "Compensation tracker: Reprocessing lost object for multi-object tracking," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 307–317, Waikoloa, HI, USA, 2022.

[65] X. Wan, S. Zhou, J. Wang, and R. Meng, "Multiple object tracking by trajectory map regression with temporal priors embedding," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 1377–1386, New York, NY, USA, 2021.

[66] W. Li, Y. Xiong, S. Yang, M. Xu, Y. Wang, and W. Xia, "Semi-TCL: Semi-supervised track contrastive representation learning," 2021, http://arxiv.org/abs/2107.02396.