Article

An Effectively Finite-Tailed Updating for Multiple Object Tracking in Crowd Scenes

Biaoyi Xu ¹, Dong Liang ^{1,*}, Ling Li ¹, Rong Quan ¹ and Mingguang Zhang ²

¹ MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Collaborative Innovation Center of Novel Software Technology and Industrialization, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China; xubiaoyi@nuaa.edu.cn (B.X.); liling@nuaa.edu.cn (L.L.); quanrong21@nuaa.edu.cn (R.Q.)

² Shanghai Institute of Satellite Engineering, Shanghai Academy of Spaceflight Technology, Shanghai 210016, China; mingguangzhang509@163.com

* Correspondence: liangdong@nuaa.edu.cn

Abstract: Multiple Object Tracking (MOT) focuses on tracking all the objects in a video. Most MOT solutions follow a tracking-by-detection or a joint detection tracking paradigm to generate the object trajectories by exploiting the correlations between the detected objects in consecutive frames. However, according to our observations, considering only the correlations between the objects in the current frame and the objects in the previous frame will lead to an exponential information decay over time, thus resulting in a misidentification of the object, especially in scenes with dense crowds and occlusions. To address this problem, we propose an effectively finite-tailed updating (FTU) strategy to generate the appearance template of the object in the current frame by exploiting its local temporal context in videos. To be specific, we model the appearance template for the object in the current frame on the appearance templates of the objects in multiple earlier frames and dynamically combine them to obtain a more effective representation. Extensive experiments have been conducted, and the experimental results show that our tracker outperforms the state-of-the-art methods on MOT Challenge Benchmark. We have achieved 73.7% and 73.0% IDf1, and 46.1% and 45.0% MT on the MOT16 and MOT17 datasets, which are 0.9% and 0.7% IDf1 higher, and 1.4% and 1.8% MT higher than FairMOT respectively.

Keywords: multiple object tracking; Re-ID; update mechanism; crowd scenes

Citation: Xu, B.; Liang, D.; Li, L.; Quan, R.; Zhang, M. An Effectively Finite-Tailed Updating for Multiple Object Tracking in Crowd Scenes. *Appl. Sci.* **2022**, *12*, 1061. <https://doi.org/10.3390/app12031061>

Academic Editor: Andrea Prati

Received: 15 November 2021

Accepted: 25 December 2021

Published: 20 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Multiple Object Tracking (MOT) [1] is one of the hotspots in the field of computer vision. Object detection and re-identification (Re-ID) are key modules of a MOT pipeline. The solution of MOT can be mainly partitioned into three steps, including locating multiple objects, maintaining their identities, and yielding their trajectories given the input video. The MOT algorithm [2–6] can be widely used in intelligent traffic flow monitoring [7] and autonomous driving [8]. Both the accuracy and the robustness of the MOT algorithm are of great significance to the validity and reliability of these intelligent applications. However, MOT is a very complicated task that is difficult to solve. On one hand, MOT needs to detect objects with similar appearances simultaneously. On the other hand, the scenes of MOT mostly include dense crowds and occlusions. To obtain a promising result in MOT, we need to construct a robust model to generate a group of distinguishable appearance templates for the video frames. However, the appearance update mechanism of most current MOT methods is formulated as a simple linear combination of the appearance template of the previous frame and the Re-ID feature of the current frame, which will cause a serious misidentification of the current object if the appearance template of the previous frame is incredible, owing to some severe occlusions. The problem is worse when similar objects occur in scenes with dense crowds and occlusions. For example, in Figure 1, the ID

of the person with a red arrow changes from 363 to 217 when the object we are tracking is occluded.

Figure 1. Illustration of the object misidentification problem caused by occlusions. The numbers above the pictures represent the frame numbers of the pictures. The red arrow points to the object we are tracking. The number in each object box represents the ID we assigned to the object.

To address this issue, we propose an effective and flexible appearance update mechanism named finite-tailed updating (FTU). Except for the appearance template of the object in the previous frame, FTU also combines the accumulated appearance templates of the objects in earlier frames with the Re-ID feature of the object in the current frame. Specifically, for an input video, we firstly input the video frame into the backbone network to obtain the detected bounding box of each object in the current frames. Then, we use the Re-ID module and our proposed update mechanism to obtain the actual appearance template of each object in the current frame. Finally, a Hungarian algorithm [9] is used to match the objects in different frames to obtain their trajectories. The contributions of this paper are as follows:

- We propose an effective and flexible appearance update mechanism, named finite-tailed updating (FTU), which combines the object's historical accumulated appearance templates in multiple earlier frames with its Re-ID feature in the current frame to improve the identification performance of the object in the current frame.
- We propose an effective MOT solution that can obtain better performance than the state-of-the-art methods. The experimental results have shown that our tracker outperforms the state-of-the-art methods on MOT Challenge Benchmark.

The remainder of this paper is organized as follows. We discuss the related works in Section 2. We formulate the problem and describe the proposed method in detail in Section 3. The experimental results are presented in Section 4, and the conclusions, limitations, and future work are presented in Section 5.

2. Related Work

2.1. Multi-Object Tracking Framework

MHT [10] is essentially an extension of the Kalman filter [11] method in MOT tasks. It performs trajectory association by solving the conditional probability model of the association between the existing trajectory and the current detection. NOMT [12] designed an accurate affinity measure to correlate detections and achieve efficient and accurate (approximate) online MOT tracking. With the rapid development of deep learning, the detector of a MOT algorithm begins to use deep neural networks and apply CNN-based object detectors, such as Faster R-CNN [13] and YOLOv3 [14], to localize all objects of interest in the input frame. The algorithm is divided into Detection-Based Tracking (DBT) and Joint Detection Tracking (JDT) methods.

Detection-Based Tracking methods treat object detection and Re-ID as two separate tasks. The MHT-DAM [15] algorithm is an extension of multiple hypotheses tracking (MHT), which incorporates online discriminative appearance modeling, but it cannot be tracked online and the algorithm runs slowly. DeepSort [16] preliminarily realized the balance of accuracy, speed, and ID switches (IDs) of the algorithm, but compared with the latest algorithm, it has a big gap in the index of accuracy and object IDs. The TubeTK [17] algorithm adopted tubes to track multiple objects. This algorithm uses multiple frames

of information at the same time, its spatio-temporal features are highly coupled, and it performs well in the case of low visibility and occlusion, but the algorithm is not an end-to-end network in the strict sense, and the association part is still an algorithm independent of the network and is based on artificial rules. Dang [18] proposed an improved YOLOV3 algorithm to address the problem of low detection accuracy resulting from an exceptionally unbalanced sample distribution of different traffic signs in the TT100K, but it also is not an end-to-end network, and it is not elegant enough. Chen [19] proposed a novel detection framework using multi-object tracking for vehicle-mounted FIR pedestrian detection, but it uses multiple KCF trackers, and cannot achieve real-time tracking.

Joint Detection Tracking methods simultaneously accomplish object detection and identity embedding (Re-ID features) in a single network to reduce the inference time by sharing most of the computation. It is divided into two branches. One is the anchor-based algorithm, Tracktor++ [20], which uses the existing depth-detection algorithm to integrate the tracking function into the detection module to simplify the training on the tracking task. Its advantage is its use of a detection network to realize the tracking function, which greatly improves the accuracy and speed of the algorithm, but it relies too much on detectors. JDE [21] used a network to realize the joint learning of detecting and embedding apparent features. It can run in real time and has good accuracy. However, the presence of the anchor frame leads to a misalignment of the detection and the embedding of the apparent features. The other is the anchor-free algorithm, which offers a larger and more flexible solution space. It eliminates the computational burden associated with the use of anchors, making detection and segmentation tasks faster and more accurate. Huang [22] proposed a Hierarchical Deep High-Resolution Network (HDHNet) and an end-to-end framework for the feature extraction of small targets in UAV scenes, but it does not manage real-time tracking. CenterTrack [23] ran in real time without relying on appearance characteristics, achieving high tracking accuracy, but it has many IDs for long-time tracking.

2.2. Update Object Appearance Template

Most trackers use simple linear interpolation to update the template in every frame. One problem with this is that the appearance information of the object decreases exponentially over time, causing the appearance information of the object to focus only on the current few frames. As the object can change in appearance due to deformation, rapid movement, or occlusion, and very little information is retained about the historical appearance of the object, this can lead to the misidentification of the object, which can result in assigning a wrong ID to the object. Gao [24] proposed a conservative update strategy that does not consider the reliability of tracking objects to reduce the influence of drift under background interference. In the tracking process, the whole sequence is decomposed into multiple long periods, and each long period is subdivided into multiple small periods. Multiple trackers are established instead of one tracker, and the number of each tracker is equal to the number of small-time periods in each large period. Each tracker has a specific update strategy, but common updates must be executed frame by frame within the previous small cycle of each large tracker, so each tracker stops updating after a certain small cycle and then restarts. Yang [25] employed a Long Short-Term Memory (LSTM) to estimate the current template by storing previous templates in memory during online tracking, which is a computationally expensive and rather complex system. Choi [26] also used a template memory, but he uses reinforcement learning to select one of the stored templates. This method fails to accumulate information from multiple frames.

3. Problem Formulation and the Approach

3.1. Revisiting FairMOT

Since our method is based on a FairMOT [27] method with a new update mechanism, in order to have a clear understanding of our method we must first review some technical details of the FairMOT method.

3.1.1. Backbone Network

FairMOT adopts ResNet-34 as a backbone to strike a good balance between accuracy and speed. An enhanced version of Deep Layer Aggregation (DLA) [28] is applied to the backbone to fuse multi-layer features. In addition, the convolution layers in all up-sampling modules are replaced by deformable convolution such that they can dynamically adjust the receptive field according to object scales and poses. These modifications are also helpful to alleviate the alignment issue.

3.1.2. Object Detection Branch

The detection branch is based on CenterNet. Three parallel heads are appended to DLA-34 to estimate heatmaps, object center offsets, and bounding box sizes, respectively. Each head is implemented by applying a 3×3 convolution (with 256 channels) to the output features of DLA-34, followed by a 1×1 convolutional layer that generates the final objects.

Heatmap Head: This head is responsible for predicting an object heatmap $H \in [0, 1]^{\frac{H}{R} \times \frac{W}{R} \times 1}$ with a default down-sampling factor R of 4. The response at a location in the heatmap is expected to be 1 if it collapses with the ground-truth object center. The response decays exponentially as the distance between the heatmap location and the object center increases.

Center Offset Head: This head is responsible for positioning the object more accurately. Since the stride of the final feature map is four, it will introduce quantization errors up to four pixels. Because the Re-ID feature needs to be extracted based on the accurate object center, the careful alignment of the Re-ID feature with the object center is critical to performance. To solve the problem of quantization errors, the branch estimates the continuous offset of each pixel relative to the center of the object to reduce the impact of down-sampling.

Box Size Head: This head is responsible for estimating the height and width of the bounding box of the object at each anchor point. The head has no direct relationship with the Re-ID feature, but the positioning accuracy will affect the evaluation of the object detection performance.

3.1.3. Object-Embedding Branch

The branch aims to generate features that can distinguish objects. Ideally, the distance between different objects in different frames should be larger than the distance between the same objects in different frames. In this way, we can make good use of appearance features to distinguish different objects. In the experiment, we apply a convolution layer with 128 kernels based on the backbone features to extract identity-embedding features for each location. The resulting feature map is $E \in R^{128 \times W \times H}$. The Re-ID feature $E_{x,y} \in R^{128}$ of an object at (x, y) is extracted from the feature map.

3.1.4. Updating the Motion Model

The update method of the motion model uses the Kalman filter. The Kalman filter is an efficient recursive filter (autoregressive filter) which can estimate the state of a dynamic system from a series of incomplete and noisy measurements. The Kalman filter is mainly divided into two steps, prediction and correction. The prediction step is to estimate the current state based on the state at the previous moment, and the correction step is to perform a comprehensive analysis based on the observation of the current state and the estimation at the previous moment to estimate the optimal state value of the system. The complete formula for the prediction step is shown below:

$$\hat{x}_k = A_k \hat{x}_{k-1} + B_k \vec{u}_k \quad (1)$$

$$P_k = A_k P_{k-1} A_k^T + Q_k \quad (2)$$

where $\hat{x}_k$ is the system state at time k , and $\bar{u}_k$ is the control vector of the system at time k . The matrix A relates the state at time step $k - 1$ to the state at step k . Matrix B is the control matrix. P_{k-1} represents the correlation between different internal states in $\hat{x}_{k-1}$, and Q_k represents noise in the prediction process. The update formula of the measurement process is as follows:

$$K_k = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1} \quad (3)$$

$$\hat{x}_k' = \hat{x}_k + K_k (z_k - H_k \hat{x}_k) \quad (4)$$

$$P_k = (I - K_k H_k) P_k \quad (5)$$

The matrix H_k represents the scale transition from a predicted state to a measured state, and the R_k represents the measurement error at time k . The z_k represents the status value of the actual measurement. The K_k represents the Kalman gain.

3.1.5. Updating the Appearance Model

If only motion models are used for matching in crowded scenes, the matching process will result in the misidentification of the object, as there will be many other people in the vicinity of the object and no appearance features are used to distinguish it. Thus, we introduce the appearance model, which will be made more accurate by combining it and the motion model for matching. We take f_1 as the appearance template obtained in the previous frame, f_2 as the Re-ID feature obtained in the current frame, and f as the updated appearance template of the current frame. λ is a hyperparameter. Specifically, all of the FairMOT, DeepSort, and CenterTrack algorithms set the value of λ to 0.9. The formula for updating the appearance model is as follows:

$$f = \lambda f_1 + (1 - \lambda) f_2 \quad (6)$$

3.2. The Proposed Updating Scheme

The appearance feature plays an important role in accurately tracking the object. When the object is occluded by the environment or other objects, the update strategy of FairMOT and some other current methods greatly limits us from obtaining a good appearance template. The appearance update mechanism of the FairMOT and some other current methods is to linearly combine the Re-ID feature of the current frame and the accumulated appearance template of the previous frame. This update mechanism is not suitable for MOT in dense scenes, which will cause a serious misidentification of the current object if the appearance template of the previous frame is incredible, owing to some severe occlusions. Compared to the FairMOT algorithm, we propose a novel appearance template update mechanism, which uses the Re-ID feature of the current frame and multiple previous frames' appearance templates to obtain the actual appearance template of the current frame. We use a dynamic method to assign weights to the appearance templates of these frames. In this way, we can get a robust and distinguishable appearance template. Even if the previous frame is incredible, owing to some severe occlusions, our proposed method can still identify the object in the current frame successfully. The comparison between our proposed method and the current method is shown in Figure 2. The T_i here represents the i -th frame to be input into the feature-extraction network.

Herein, we propose a dynamic weight-distribution method. In this method, we first set the weight of the T_n frame γ to 0.1, so that the appearance template we obtained can adapt to the changes in the appearance of the object. Then, we set the weight of the T_0 frame μ_0 as high as possible to ensure that its appearance information will not decay too quickly over time. μ_0 is set to 0.6 in our experiment. For the rest of the frames, we dynamically distribute their weights according to the values of γ and μ_0 . Through this weight-distribution strategy, we can get a more robust appearance template to improve the identification performance of the object in the current frame. The specific formula for our update mechanism is shown below.

$$f'_n = \gamma f_n + \sum_{i=0}^{N-1} \mu_i f_i \quad (7)$$

$$\gamma + \sum_{i=0}^{N-1} \mu_i = 1 \quad (8)$$

We take f_i as the appearance template obtained in the T_i frame, f_n as the Re-ID feature obtained in the current frame, and f'_n as the updated actual appearance template of the current frame. γ and μ_i are hyperparameters.

Figure 2. The detailed information of the appearance template update mechanism of the current method and the appearance template update mechanism of our proposed method. λ , γ , and μ_i are hyperparameters.

We explore how many frames the finite-tailed window has in our update mechanism in the following experiments. The specific experimental results can be seen in the Section 4.3 in the experimental part.

3.3. The Architecture of the Proposed MOT Framework

We propose an effectively finite-tailed updating (FTU) strategy to generate the appearance template, which exploits the local temporal context of the object in videos and dynamically combines them to enhance the feature representation of the object. Figure 3 is the overview architecture of our MOT framework. The T_i here represents the i -th frame to be input into the backbone network. The input frames are obtained directly without any processing. We first input the T_0 frame into the network and then obtain the motion and appearance features of all objects in the frame. Then, the T_i frame is input into the network to obtain the motion and appearance features of all objects. Finally, the current frame is input into the network to obtain the motion and appearance features of all objects. We use the cosine distance to calculate the similarity between the appearance features of all the objects in the T_{i-1} frame and the appearance features of all the objects in the T_i frame. The Mahalanobis distance is used to calculate the similarity between the movement features of all objects in the T_{i-1} frame and the movement features of all objects in the T_i frame. We use a hyperparameter λ to combine the cosine distance and Mahalanobis distance to obtain the cost matrix. We use the Hungarian algorithm to match all the objects in the T_{i-1} frame and the T_i frame. If the object is successfully matched, in order to obtain better appearance and movement features, we need to update the appearance and movement features to adapt to the changes in the appearance and movement of the object in different frames.

Figure 3. The architecture picture of our approach, T_0 represents the initial frame, T_n represents the current frame, and λ is a hyperparameter for calculating the cost matrix. Bounding boxes and identities are marked in the frames. Bounding boxes with different colors represent different identities.

The updating methods of all the object motion models are as follows. We use the Kalman filter to predict the movement features of objects in the T_i frame through the actual movement features of objects in the T_{i-1} frame, and then update the actual movement features of objects in the T_i frame by combining the movement features measured in the T_i frame with the movement features predicted in the T_{i-1} frame. Then, the movement features of the object in the current frame are predicted by the actual movement features of the object in the T_i frame. Finally, the actual movement features of the object in the current frame are updated by combining the predicted movement features of the object in the T_i frame with the measured movement features of the object in the current frame.

The updating method of the appearance features of objects is as follows. Firstly, we input the T_{i-1} frame into the network, and then combine it with the detection box of the object obtained by the backbone network. We intercept all the objects in the frame and then extract the appearance features of all the objects in the T_{i-1} frame through the Re-ID feature extraction network. Then, the T_i frame is input into the backbone network, we repeat the process of the T_{i-1} frame, and extract the appearance features of all the objects in the T_i frame. Finally, we dynamically combine the historical accumulated appearance templates of objects in multiple frames and the Re-ID features of the current frame to update the actual appearance template of the current frame.

4. Experiment

4.1. Experimental Setting

Datasets: Similar to some earlier work [21,27], we use ETH [29], CityPerson [30], CalTech [31], MOT17 [32], CUHK-SYSU [33], and PRW [34] as our training datasets. We only use the ETH and CityPerson datasets to train the detection branches because they only have box annotations. There are both box annotations and identity annotations in the other datasets. We use them to train the Detection and Re-ID branches at the same time. Because some data in ETH also appear in the test set of MOT16, we removed these data from the ETH training set to be fair.

Metrics: The performance of our algorithm is evaluated based on the datasets of the MOT Challenge Benchmark. We extensively evaluate a variety of factors of our approach on the testing sets of three benchmarks: MOT16, MOT17, and MOT20. We adopt the CLEAR [35] MOT metrics, MOTA, IDF1, HOTA, ML, and MT, for our experimentation and comparison against the state-of-the-art methods. Multi-Object Tracking Accuracy (MOTA) denotes standard deviation across all sequences. This measure combines three error sources: false positives, false negatives, and identity switches. IDF1 denotes the ratio of correctly identified detections over the average number of ground-truth and computed detections. The formula for MOTA and IDF1 are shown in Formulas (9) and (10). The GT_t , FP_t , FN_t , and $IDSW_t$ represent the number of ground-truth bounding boxes, false positives, false negatives, and identity switches at time t , respectively. The IDTP, IDFP, and IDFN are

the total number of true positive, false positive, and false negative object identifications, respectively. The Higher-Order Tracking Accuracy (HOTA) is a newly defined indicator. The previous indicator overemphasized the importance of detection or correlation. To solve this problem, we use the new MOT evaluation index, HOTA, which balances the effects of performing accurate detection, correlation, and positioning into a single unified index for comparing trackers. MT denotes mostly tracked objects, it represents the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life spans. ML denotes mostly lost objects. It represents the ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life spans.

$$MOTA = 1 - \frac{\sum_t (FP_t + FN_t + IDSW_t)}{\sum_t GT_t} \quad (9)$$

$$IDF1 = \frac{2IDTP}{2IDTP + IDFP + IDFN} \quad (10)$$

It can be seen from the above formula that the MOTA indicator focuses on the IDSW. The greater the number of IDSW, the lower the score of MOTA. However, while it simply considers the number of IDSW, it cannot determine whether the ID has changed back to the correct ID. The IDF1 indicator focuses on the identification of the ID. The higher the IDF1 score, the higher the proportion of the correct ID number. Therefore, in this paper, we pay more attention to the IDF1 indicator than the MOTA indicator

Implementation details: Our encoder-decoder is a fully-convolutional DLA-34 variant. The model parameters pre-trained on the COCO dataset [36] are used to initialize our model. We train our model with an Adam optimizer [37] for 30 epochs with a starting learning rate of $1e^{-4}$. The batch size is set to be 12. The input frame is resized to 1088×608 and the feature map resolution is 272×152 . The confidence threshold for tracking is set to 0.4. The hyperparameter of the control motion model in the cost matrix is set to 0.9815. For the first association between objects, we integrate the motion model to find a match and set the threshold to 0.4. If there is no association, we use the iou distance between the objects for the second match, and we set the threshold to 0.6. For objects that have only appeared once in multiple frames, we match the iou distance between the objects, and we set the threshold to 0.7.

4.2. Comparison with State-of-the-Art Methods

Herein, we compare our approach to the state-of-the-art trackers, including the one-step methods and the two-step methods. We submitted our method to Multiple Object Tracking Benchmark, which is the official website of MOT. Table 1 shows the comparison of our method with some state-of-the-art methods on MOT16, MOT17, and MOT20 datasets. All the results are from Multiple Object Tracking Benchmark.

As shown in Table 1, compared with the state-of-the-art methods that have been published on Multiple Object Tracking Benchmark, our method ranks first in the IDF1, HOTA, MT, and ML indicators on the MOT17 datasets. The IDF1 indicators measure a tracker's ability to maintain long consistent tracks. HOTA is a newly proposed indicator, which balances the effects of performing accurate detection, correlation, and positioning into a single unified indicator. It can better represent a human visual evaluation of tracking performance. On the MOT16 and MOT20 datasets, our method has also achieved very good results on HOTA, IDF1, MT, ML, and MOTA indicators, compared with the state-of-the-art methods.

4.3. Hyperparameter Comparison and Analysis Experiments

How many frames should be selected? How should the features of these frames for updating be combined? How should the weight of each frame be allocated? In order to solve these problems, we conduct a lot of experiments to find an optimal update strategy to obtain the best appearance template.

Table 1. Comparison of the state-of-the-art methods on the Multiple Object Tracking Benchmark.

Dataset	Tracker	MOTA↑ (%)	IDF1↑ (%)	HOTA↑	MT↑ (%)	ML↓ (%)
MOT16	FairMOT [27]	74.9	72.8	-	44.7	15.9
	TMOH [38]	63.2	63.5	50.7	27.0	31.0
	mfi_tst [39]	59.9	58.7	46.9	24.1	30.8
	MPNTrack [40]	58.6	61.7	48.9	27.3	34.0
	LPC_MOT [41]	58.8	67.6	51.7	27.3	35.0
	Tracktor++v2 [20]	54.4	52.5	42.3	19.0	36.9
	Ours	74.4	73.7	60.2	46.1	15.2
MOT17	FairMOT [27]	73.7	72.3	-	43.2	17.3
	TMOH [38]	62.1	62.8	50.4	26.9	31.4
	mfi_tst [39]	60.1	58.8	47.2	26.0	27.2
	MPNTrack [40]	58.8	61.7	49.0	28.8	33.5
	LPC_MOT [41]	59.0	66.8	51.5	29.9	33.9
	Tracktor++v2 [20]	56.3	55.1	44.8	21.1	35.3
	Ours	73.1	73.0	59.7	45.0	16.8
MOT20	FairMOT [27]	61.8	67.3	-	68.8	7.6
	TMOH [38]	60.1	61.2	48.9	46.7	17.8
	mfi_tst [39]	59.3	59.1	47.1	41.1	17.3
	MPNTrack [40]	57.6	59.1	46.8	38.2	22.5
	LPC_MOT [41]	56.3	62.5	49.0	40.9	17.8
	Tracktor++v2 [20]	52.6	52.7	42.1	29.4	26.7
	Ours	59.7	67.8	54.0	66.5	7.7

Comparison of different backbones: In order to verify the effect of our update method on which backbone network is better, we use different backbone networks, such as vanilla ResNet, Feature Pyramid Network, High-Resolution Network, and DLA-34. In order to ensure the fairness of the experiment, except for the different backbone networks, the other experimental conditions are consistent. For all backbone networks, we set the stride of the final feature map to be four. In order to make the vanilla ResNet's stride of the final feature map four, we add three upsampling operations. The result is shown in Table 2.

Table 2. Comparison of different backbones on the validation set of MOT17 dataset.

Backbone	MOTA↑ (%)	IDF1↑ (%)	IDs↓	AP↑	TPR↑
ResNet-34	63.7	67.4	427	75.2	90.8
ResNet-50	63.8	67.9	506	75.8	92
ResNet-34-FPN	64.5	70.2	367	77.9	94.3
HRNet-W18	67.6	74.5	317	80.7	94.8
DLA-34	69.0	73.5	304	81.4	94.3

Comparison of different Re-ID feature dimensions: It is shown in FairMOT [27] that feature dimensionality plays an important role in balancing detection and tracking accuracy. In order to verify which dimensional Re-ID features are most effective in our update method, we conduct a series of experiments on the Re-ID feature dimensions 512, 256, 128, and 64. To maintain the fairness of the experiment, the experimental conditions

are kept constant, except for the different dimensions of the Re-ID features. The result is shown in Table 3.

Table 3. Comparison of different Re-ID feature dimensions on the validation set of MOT17 dataset.

Backbone	dim	MOTA↑(%)	IDF1↑ (%)	IDs↓	FPS↑
DLA-34	512	68.6	73.9	314	18.68
DLA-34	256	68.5	72.5	339	19.91
DLA-34	128	69.0	73.5	304	20.29
DLA-34	64	69.2	72.8	299	21.27

Comparison of different numbers of appearance template frame: In order to verify how many frames the finite-tailed window has in our update mechanism, we set N in Formula (7) to two, three, and up to nine for the experiment, respectively. We use a dynamic weight-distribution method. For the different frames used for the update, we fixedly set γ in Formula (7) to 0.1. We set the value of μ_0 in Formula (7) as large as possible and set the same value to the rest of μ_i . From the experimental results, we can find that when the N is two, the IDF1, MOTA, MT, and ML indicators can obtain a good result, especially the IDF1 indicator, which obtains the maximum value of 73.5. When the N increases from two to four, the IDF1, MOTA, MT, and ML indicators begin to decline, and obtain the lowest score when the N is four. When the N is increased from four to seven, the IDF1 and MOTA indicators begin to rise. When the N exceeds seven, all indicators tend to stabilize. Therefore, we select N as two in our update mechanism. Next, we explore how to distribute the weight of the T_0 frame. We set the value of μ_0 to 0.1–0.8. We find that when the value of μ_0 is 0.6, the IDF1 indicator achieves the highest score. Thus, we set the value of μ_0 to 0.6 in our update mechanism. Therefore, the final settings of our experiment are $N = 2$, $\mu_0 = 0.6$, $\mu_1 = 0.3$, and $\gamma = 0.1$. The specific experimental results are shown in Figures 4 and 5.

Figure 4. The change of IDF1, MT, ML, and MOTA indicators as N changes.

Figure 5. The change of IDF1 indicator as u_0 changes.

Parameter discussion: We also verified the influence of the tracking confidence threshold on the IDF1, MT, ML, and MOTA metrics obtained by our method. We adjust the tracking confidence threshold from 0.03 to 0.96, adding 0.03 to the tracking threshold for each experiment. The rules of the IDF1, MT, ML, and MOTA indicators, following the change of the tracking confidence threshold, are shown in Figure 6.

Figure 6. The change of IDF1, MT, ML, and MOTA indicators as the tracking confidence threshold changes.

4.4. Comparison to Baseline Method FairMOT

In order to verify that our finite-tailed updating (FTU) strategy can maintain a better continuity after being occluded than the standard linear update mechanism of the baseline method, we compare it with the baseline method FairMOT. After inputting the same video data of MOT16, we use the baseline method FairMOT and our method for tracking, respectively. As shown in Figure 7, our method can effectively maintain the consistency of the object ID after being occluded. In the baseline method FairMOT, the ID of the object changes after being occluded. We also submitted our method to Multiple Object Tracking Benchmark and compared it with the experimental results of the baseline method FairMOT using the current update mechanism. The specific results are shown in Table 4.

Figure 7. Comparison of our method and the baseline method. We compare the same video data of MOT16. The red arrow points to the object we used for the tracking-performance comparison. It can be seen from the figure that, when using the FairMOT method for tracking, the ID of the object changes from 363 to 217 after occlusion. However, when using our method for tracking, the ID remains at 366 after occlusion.

Table 4. Comparison of the baseline method on Multiple Object Tracking Benchmark.

Dataset	Tracker	MOTA↑ (%)	IDF1↑ (%)	HOTA↑	MT↑ (%)	ML↓ (%)
MOT16	FairMOT [27]	74.9	72.8	-	44.7	15.9
	Ours	74.4	73.7	60.2	46.1	15.2
MOT17	FairMOT [27]	73.7	72.3	-	43.2	17.3
	Ours	73.1	73.0	59.7	45.0	16.8
MOT20	FairMOT [27]	61.8	67.3	-	68.8	7.6
	Ours	59.7	67.8	54.0	66.5	7.7

5. Conclusions

In this work, we have proposed a novel appearance template update mechanism, named FTU, in the MOT field. By combining the accumulated appearance templates of the objects in multiple earlier frames with the Re-ID feature of the current frame, FTU succeeds in identifying the occluded objects even in scenes with dense crowds and occlusions. We have conducted extensive experiments on three widely used datasets, including the MOT16, MOT17, and MOT20 datasets. The experimental results have demonstrated the superiority of our proposed method over other state-of-the-art methods. In addition, we have conducted several analysis experiments to show the effectiveness of the proposed strategy as well as the parameter settings of our method.

We believe that this method still has some room for improvement. For example, the current update window parameter is fixed for the entire video. We believe that adaptive update windows can further improve system performance. In the future, we will try to integrate this update strategy with new multi-object tracking algorithms and apply it to multi-modal tracking tasks.

Author Contributions: Conceptualization, B.X. and D.L.; methodology, B.X.; software, B.X.; validation, B.X., D.L. and L.L.; formal analysis, B.X.; investigation, B.X.; resources, M.Z.; data curation, B.X.; writing—original draft preparation, B.X.; writing—review and editing, R.Q.; visualization, B.X.; supervision, D.L.; project administration, D.L.; funding acquisition, D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by AI+ Project of NUAU (XZA20003), National Science Foundation of China (61772268)

Institutional Review Board Statement: Ethical review and approval were waived for this study, due to the datasets are public datasets, and other paper such as JDE and FairMOT use the public datasets.

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Luo, W.; Xing, J.; Milan, A.; Zhang, X.; Liu, W.; Kim, T.-K. Multiple object tracking: A literature review. *Artif. Intell.* **2020**, *293*, 103448. [CrossRef]
2. Yu, F.; Li, W.; Li, Q.; Liu, Y.; Shi, X.; Yan, J. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 36–42.
3. Bochinski, E.; Eiselein, V.; Sikora, T. High-speed tracking-by-detection without using image information. In *Proceedings of the 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Lecce, Italy, 29 August–1 September 2017; pp. 1–6.
4. Hornakova, A.; Henschel, R.; Rosenhahn, B.; Swoboda, P. Lifted disjoint paths with application in multiple object tracking. In *Proceedings of the International Conference on Machine Learning (ICML)*, Vienna, Austria, 12–18 July 2020; pp. 4364–4375.
5. Tokmakov, P.; Li, J.; Burgard, W.; Gaidon, A. Learning to track with object permanence. *arXiv* **2021**, arXiv:2103.14258.
6. Zhou, X.; Koltun, V.; Krähenbühl, P. Tracking objects as points. In *European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 474–490.
7. Tian, B.; Yao, Q.; Gu, Y.; Wang, K.; Li, Y. Video processing techniques for traffic flow monitoring: A survey. In *Proceedings of the 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, Washington, DC, USA, 5–7 October 2011; pp. 1103–1108.
8. Brown, M.; Funke, J.; Erlien, S.; Gerdes, J.C. Safe driving envelopes for path tracking in autonomous vehicles. *Control Eng. Pract.* **2017**, *61*, 307–316. [CrossRef]
9. Kuhn, H.W. The hungarian method for the assignment problem. *Nav. Res. Logist. Q.* **1955**, *2*, 83–97. [CrossRef]
10. Blackman, S.S. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerosp. Electron. Syst. Mag.* **2004**, *19*, 5–18. [CrossRef]
11. Welch, G.; Bishop, G. An Introduction to the Kalman Filter. Chapel Hill, NC, USA, 1995. Available online: <https://perso.crans.org/club-krobot/doc/kalman.pdf> (accessed on 17 September 1997).
12. Choi, W. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 7–13 December 2015; pp. 3029–3037.
13. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 91–99. [CrossRef] [PubMed]
14. Redmon, J.; Farhadi, A. Yolo3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
15. Kim, C.; Li, F.; Ciptadi, A.; Reh, J.M. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 7–13 December 2015; pp. 4696–4704.
16. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In *Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP)*, Beijing, China, 17–20 September 2017; pp. 3645–3649.
17. Pang, B.; Li, Y.; Zhang, Y.; Li, M.; Lu, C. Tubetk: Adopting tubes to track multi-object in a one-step training model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020; pp. 6308–6318.
18. Park, Y.; Dang, L.M.; Lee, S.; Han, D.; Moon, H. Multiple object tracking in deep learning approaches: A survey. *Electronics* **2021**, *10*, 2406. [CrossRef]
19. Chen, H.; Cai, W.; Wu, F.; Liu, Q. Vehicle-mounted far-infrared pedestrian detection using multi-object tracking. *Infrared Phys. Technol.* **2021**, *115*, 103697. [CrossRef]
20. Bergmann, P.; Meinhardt, T.; Leal-Taixe, L. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea, 27 October–2 November 2019; pp. 941–951.
21. Wang, Z.; Zheng, L.; Liu, Y.; Li, Y.; Wang, S. Towards real-time multi-object tracking. In *European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 107–122.
22. Huang, W.; Zhou, X.; Dong, M.; Xu, H. Multiple objects tracking in the uav system based on hierarchical deep high-resolution network. *Multimed. Tools Appl.* **2021**, *80*, 13911–13929. [CrossRef]
23. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as points. *arXiv* **2019**, arXiv:1904.07850.
24. Gao, Y.; Hu, Z.; Yeung, H.W.F.; Chung, Y.Y.; Tian, X.; Lin, L. Unifying temporal context and multi-feature with update-pacing framework for visual tracking. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 1078–1091. Available online: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8660578> (accessed on 28 April 2020). [CrossRef]

25. Yang, T.; Chan, A.B. Learning dynamic memory networks for object tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 152–167.
26. Choi, J.; Kwon, J.; Lee, K.M. Visual tracking by reinforced decision making. *arXiv* **2017**, arXiv:1702.06291v1.
27. Zhang, Y.; Wang, C.; Wang, X.; Zeng, W.; Liu, W. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *Int. J. Comput. Vis.* **2021**, 1–19. [[CrossRef](#)]
28. Yu, F.; Wang, D.; Shelhamer, E.; Darrell, T. Deep layer aggregation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 2403–2412.
29. Ess, A.; Leibe, B.; Schindler, K.; Gool, L.V. A mobile vision system for robust multi-person tracking. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
30. Zhang, S.; Benenson, R.; Schiele, B. Citypersons: A diverse dataset for pedestrian detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3213–3221.
31. Dollár, P.; Wojek, C.; Schiele, B.; Perona, P. Pedestrian detection: A benchmark. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, FL, USA, 20–25 June 2009; pp. 304–311.
32. Milan, A.; Leal-Taixé, L.; Reid, I.; Roth, S.; Schindler, K. Mot16: A benchmark for multi-object tracking. *arXiv* **2016**, arXiv:1603.00831.
33. Xiao, T.; Li, S.; Wang, B.; Lin, L.; Wang, X. Joint detection and identification feature learning for person search. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3415–3424.
34. Zheng, L.; Zhang, H.; Sun, S.; Chandraker, M.; Yang, Y.; Tian, Q. Person re-identification in the wild. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1367–1376.
35. Bernardin, K.; Stiefelwagen, R. Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP J. Image Video Process.* **2008**, 2008, 1–10. Available online: <https://link.springer.com/content/pdf/10.1155/2008/246309.pdf> (accessed on 23 April 2008). [[CrossRef](#)]
36. Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 740–755.
37. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
38. Stadler, D.; Beyerer, J. Improving multiple pedestrian tracking by track management and occlusion handling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 10958–10967.
39. Yang, J.; Ge, H.; Yang, J.; Tong, Y.; Su, S. Online multi-object tracking using multi-function integration and tracking simulation training. *Appl. Intell.* **2021**, 2021, 1–21. [[CrossRef](#)]
40. Brasó, G.; Leal-Taixé, L. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020; pp. 6247–6257.
41. Dai, P.; Weng, R.; Choi, W.; Zhang, C.; He, Z.; Ding, W. Learning a proposal classifier for multiple object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 2443–2452.