applied sciences

MDPI

Article

Multi-Object Tracking with Grayscale Spatial-Temporal Features

Longxiang Xu and Guosheng Wu *

School of Electronic Information, Qingdao University, Qingdao 266071, China; x9902270@163.com
* Correspondence: xuxlx202311@163.com

Featured Application: Security monitoring and autonomous driving.

Abstract: In recent multiple object tracking (MOT) research, there have not been many traditional methods and optimizations for matching. Most of today's popular tracking methods are implemented using deep learning. But many monitoring devices do not have high computing power, so real-time tracking via neural networks is difficult. Furthermore, matching takes less time than detection and embedding, but it still takes some time, especially for many targets in a scene. Therefore, in order to solve these problems, we propose a new method by using grayscale maps to obtain spatial-temporal features based on traditional methods. Using this method allows us to directly find the position and region in previous frames of the target and significantly reduce the number of IDs that the target needs to match. At the same time, compared to some end-to-end paradigms, our method can quickly obtain spatial-temporal features using traditional methods, which reduces some calculations. Further, we joined embedding and matching to further reduce the time spent on tracking. Our method reduces the calculations in feature extraction and reduces unnecessary matching in the matching stage. Our method was evaluated on benchmark dataset MOT16, and it achieved great performance; the tracking accuracy metric MOTA reached 46.7%. The tracking FPS reached 17.6, and it ran only on a CPU without GPU acceleration.

Keywords: multiple object tracking; tracking-by-detection; grayscale; spatial-temporal features

Citation: Xu, L.; Wu, G. Multi-Object Tracking with Grayscale Spatial-Temporal Features. *Appl. Sci.* **2024**, *14*, 5900. https://doi.org/10.3390/app14135900

Academic Editors: Saul Tovar-Arriaga and Luis Alberto Morales-Hernández

Received: 14 June 2024
Revised: 2 July 2024
Accepted: 5 July 2024
Published: 5 July 2024

1. Introduction

Multiple object tracking (MOT) is an important research field in computer vision, and it is also an integral part of cutting-edge technologies such as action recognition. Recently, MOT has achieved a wide range of applications in artificial intelligence, such as autonomous driving, intelligent traffic management, and video monitoring. It aims to detect the target of interest, assign each target a separate ID, maintain target identities, and generate their respective trajectories throughout the entire video stream. With the development of MOT algorithms, it is still a challenge to improve the accuracy and speed, as the overlap matching between many objects can cause the target detection to fail or slow.

In early computer vision, tracking relied on the features of interest points, which was simple and fast but poorly robust. In recent years, deep learning has developed rapidly and has been widely used in the image field. With the continuous iteration of YOLO [1], Faster R-CNN [2], and other models [3], the speed and accuracy of the target detection algorithm have been significantly improved. So much MOT research is based on tracking by detection (TBD) [4]. However, the TBD model not only takes a long time for target detection but also requires some calculation time for feature extraction and ID matching. In order to solve this problem, a new model that joins detection and embedding (JDE) [5] is proposed. Although the JDE model reduces the amount of calculation compared to the TBD model, it only combines the two parts of target detection and feature extraction, so the complexity of the model is still high. Recently, a new model named joint detection and tracking (JDT) [6] was proposed by combining target detection, feature extraction, and data association in a single network, which simplifies tracking. But the association strategy still

has a lot of complexity and calculation, as each target needs to be matched multiple times. Many JDT models use spatial-temporal features; however, their use of neural networks to extract spatial-temporal features requires extensive computation.

Today, many monitoring devices do not have high computing power and do not support real-time tracking through neural networks. But most of the popular methods today are implemented using deep learning. Therefore, there are two problems to be solved in this paper. The first is how to quickly obtain spatial-temporal features in the traditional way. The second is how to reduce unnecessary feature extraction and matching. In order to solve these problems, we propose a new method to obtain spatial-temporal features, which is convenient only based on grayscale maps. The proposed spatial-temporal feature is called the grayscale spatial-temporal feature. This spatial-temporal feature marks each target with different grayscale values in each frame, which can quickly obtain the spatial-temporal features of the target. At the same time, the matching of grayscale spatial-temporal features is also convenient and fast. We also optimize the association strategy by joint feature extraction and matching to reduce unnecessary feature extraction. Our feature extraction and matching operations are performed in a traditional way without deep learning. In this paper, we extract motion features through the Kalman filter algorithm and predict the target position of the next frame through the state equation and the target position of the historical frame. However, the Kalman filter algorithm is only applicable to linear motion, so it is necessary to extract appearance features to increase the robustness of tracking. The color histogram is used to obtain the appearance features. The color histogram can intuitively reflect the color distribution and color characteristics of the image. The color histogram is more efficient than the scale-invariant feature transform (SIFT) in extracting appearance features and is simpler to calculate. Although the optical flow estimation algorithm has high computational efficiency, it is sensitive to illumination changes and is not suitable for target occlusion or large-scale target tracking. Therefore, we do not need many computation resources and data to train. Its accuracy is comparable to some current deep learning methods on MOT16 benchmarks [7]. In summary, this paper proposed the following four contributions:

- A way to quickly extract spatial-temporal features just by traditional methods. The spatial-temporal features can deliver information perfectly between two frames. This way can quickly obtain the spatial-temporal features of the target. This method does not require the use of deep learning methods.
- A method to reduce ID matching times. This method reduces matching times to less than three times for most targets by using the spatial-temporal features.
- A method joins feature extraction and matching to reduce unnecessary feature extraction and matching. Target matching is performed in three stages. Different strategies are adopted for matching and feature extraction according to each stage.
- A better way to deal with occlusion problems. Based on the spatial-temporal features, we can infer the region where the occluded target disappears. After it is re-detected, matching is performed in this region, which reduces the interference of similar targets outside the area.

2. Related Work

Some research on TBD mainly relied on the motion features or appearance features of the target. Bewley et al. [4] modeled the position and velocity of each target and then, based on the Intersection-over-Union (IoU), matched the target. Due to the long-term loss of the detected target, the Kalman filter parameters deviated from the true value, and tracking might fail. To solve this problem, Cao et al. [8] proposed OC-SORT, which proposed an observation-centric update strategy to reduce the accumulated errors with time. Qin et al. [9] also aimed to solve the problem of long-term tracking and designed a recovery module to stitch together fragmented trajectories. Unmatched detections and trajectories were matched by calculating the velocity–time relationship between them. Yu et al. [10] designed a feature extraction network based on GoogleNet [11] to extract the appearance

features of the target and associated targets through the k-dense neighbor algorithm [12]. However, the tracking capability of these methods, which relied on the motion features, was reduced for scenes with complex motion trajectories, and the appearance model was easily affected by lighting, occlusion, and other conditions, which might cause tracking failure.

In order to further improve the tracking accuracy, some methods combined motion and appearance features. Li et al. [13] designed a self-correcting KF for predicting the target position and evaluated the similarity between targets through a recurrent neural network. Yang et al. [14] combined motion features with appearance. They used a newly introduced strategy of weak cues (confidence state and height state) to increase the robustness for occlusion. However, due to the high complexity of the network and the large number of calculations, the tracking speed of these algorithms is slow, making it difficult to meet the requirements of real-time tracking.

Therefore, the paradigm of completing object detection and feature extraction in a single network has attracted attention. Wang et al. [5] proposed the JDE864 algorithm, which integrates object detection and feature extraction into one network. Lu et al. [15] added a branch for extracting instance-level Re-ID features based on RetinaNet [16] for subsequent data association. Jin et al. [17] designed a feature decoupling module and proposed a simple three-stage data association. The combination of detection and re-identification modules in these methods can improve efficiency and reduce the time for feature extraction. But object detection and re-identification are two different tasks, so balancing the differences between the two tasks is a difficult problem, and this paradigm still does not consider optimizing matching.

In recent years, the paradigm of completing three subtasks of detection, embedding, and tracking in a single network has attracted the attention of many scholars. Recently, transformer [18] achieved remarkable success in multiple computer vision tasks. Transformer has powerful feature representation capabilities and good parallel computing capabilities. An end-to-end paradigm is proposed based on Transformer. Meinhardt et al. [19] proposed a new MOT framework, TrackFormer. This framework is based on DETR [20] and is an end-to-end framework that uses tracking-by-attention as a new idea to achieve data association between frames. Zhang et al. [21] proposed MOTRv2 based on MOTR [22]. They think that it is conflicting for the DETR network to perform the two tasks of detection and data association at the same time. Therefore, they added an additional object detector based on MOTR, and the model is an end-to-end model. Transformer has powerful feature representation capabilities and good parallel computing capabilities. But its computational complexity is high, and its real-time performance is poor.

There are some other methods to tracking targets. Ballester et al. [23] proposed DOT (dynamic object tracking), which combines instance segmentation and multi-view geometry. Yu et al. [24] proposed trajectory masks for tracking. The trajectory mask network is used to extract multidimensional information, including the estimate, approximate location, and shape of the target. Saleh et al. [25] proposed a probabilistic autoregressive motion model, which learns and generates multimodal distributions of motion trajectories, matching based on trajectories and detection frames.

In these paradigms, there are not many ways to use traditional methods to extract features and optimize ID matching. In the matching stage, as the number of detection targets increases, the matching times will increase significantly, and in order to increase the real-time performance of the program, it is necessary to reduce the use of neural networks to extract features. The grayscale spatial-temporal feature we proposed can reduce a large number of calculations, and it has better robustness during occlusion.

3. Method

3.1. Overview

In order to better explain the proposed method, there are a few symbols that need to be defined:

$\boldsymbol{P_t^i}$: represents a one-dimensional set at frame t, and i is the size of this set, for example, $\boldsymbol{P_t^i} = \left\{ p_t^0, p_t^1, p_t^2, \ldots, p_t^{i-1}, \right\}$.

$\boldsymbol{RcID_{t,i}^k}$: represents a two-dimensional set at frame t: k is the size of a one-dimensional set, and i is the number of one-dimensional sets in this two-dimensional set, for example, $\boldsymbol{RcID_{t,i}^k} = \left\{ \left\{ rcID_{t,0}^0, \ldots, rcID_{t,0}^{k-1}, \right\}, \ldots, \left\{ rcID_{t,i-1}^0, \ldots, rcID_{t,i-1}^{k-1}, \right\} \right\}$.

In the following, the region spatial-temporal feature is abbreviated as region-STF and target spatial-temporal feature is abbreviated as target-STF. Table 1 shows the interpretation of notation in the manuscript.

Table 1. The notation in the manuscript.

Notation	Interpretation
$\boldsymbol{t}$	Frame
$\boldsymbol{P_t^i}$	Detection targets
$\boldsymbol{Pb_t^i}$	Bounding box for each target
$\boldsymbol{Pm_t^i}$	Segmentation mask for each target
$\boldsymbol{FmI_t}$	Foreground mask map
$\boldsymbol{PsI_t}$	Target-STF map
$\boldsymbol{RsI_t}$	Region-STF map
$\boldsymbol{Ps_{t,i+1}^k}$	Matching results of the target-STF
$\boldsymbol{Rrs_{t,i}^k}$	Matching results of the region-STF
$\boldsymbol{Rs_t^i}$	The region to which the target belongs
$\boldsymbol{PcId_t^i}$	The successfully matched IDs
$\boldsymbol{RcID_{t,i}^k}$	The successfully matched IDs contained in each region
$\boldsymbol{RID_{t,i}^k}$	Unmatched IDs of the previous frame in each region
$\boldsymbol{UID_t^i}$	Unmatched IDs of the previous frame do not belong to any region in the current frame

As shown in Figure 1, our framework is based on tracking-by-detection, with embedding and tracking combined.

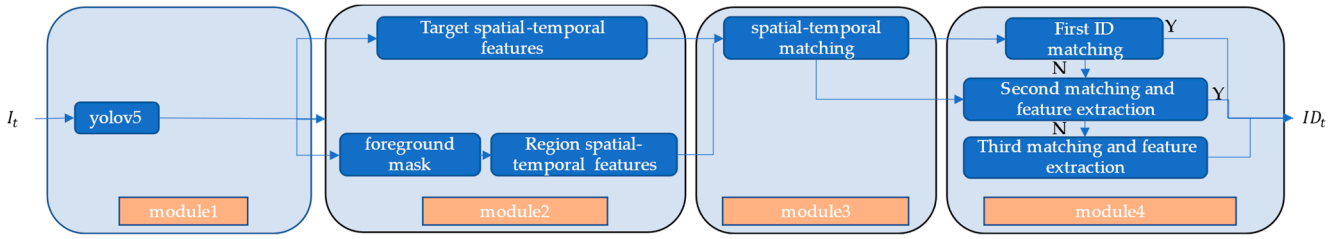

Figure 1. The framework of our method.

It performs online tracking in four steps:

In module1, continuous frames are input into the yolov5s module to obtain the target information. As the yolov5s module has image segmentation function, we can not only obtain the bounding box ($\boldsymbol{Pb_t^i}$) of each target ($\boldsymbol{P_t^i}$) but also obtain the segmentation mask ($\boldsymbol{Pm_t^i}$) of each target.

In module2, we obtain the foreground mask map ($\boldsymbol{FmI_t}$), target-STF map ($\boldsymbol{PsI_t}$), and region-STF map ($\boldsymbol{RsI_t}$), which rely on $\boldsymbol{Pb_t^i}$ and $\boldsymbol{Pm_t^i}$.

In module3, we match spatial-temporal features and use the $\boldsymbol{Ps_t}$ and $\boldsymbol{Rrs_t}$ to save the matching results of the target-STF and region-STF, respectively.

In module4, we classify targets by $\boldsymbol{Ps_t}$. According to the classification results, the targets are matched hierarchically. Continuously detected targets are matched first, then re-detected targets are matched. We use a histogram to extract the appearance features of the target and use the yolov5seg model to cut the target. This can remove background

noise interference when extracting appearance features and remove interference between occluded targets.

After matching, we update the trajectory of the target and record the ID for each target $\boldsymbol{P}_t^i$ by using $\boldsymbol{PcId}_t^i$. The successfully matched IDs contained in each region are recorded by the set $\boldsymbol{RcID}_{t,i}^k$. The unmatched targets in the previous frame are stored in $\boldsymbol{RID}_{t,i}^k$ or $\boldsymbol{UID}_t^i$. For the unmatched IDs in each region in the previous frame, if their region spatial-temporal features are not successfully matched in the current frame, these targets are recorded in $\boldsymbol{UID}_t^i$, and if they are matched successfully, they are recorded in $\boldsymbol{RID}_{t,i}^k$. The IDs contained in $\boldsymbol{RID}_t^i$ only keep 200 frames. The IDs contained in $\boldsymbol{UID}_t^i$ only keep 50 frames.

3.2. Grayscale Spatial-Temporal Features

Grayscale spatial-temporal feature includes two features: target-STF and region-STF. These two spatial-temporal features of all targets in each frame are saved with two grayscale images, respectively, and each grayscale value in the map corresponds to a spatial-temporal feature. As shown in Figure 2, the grayscale maps on the left are target-STF maps, and the grayscale maps on the right are region-STF maps. In the target-STF map, each target-STF corresponds to a target. In the region-STF map, each region-STF corresponds to one or more targets.

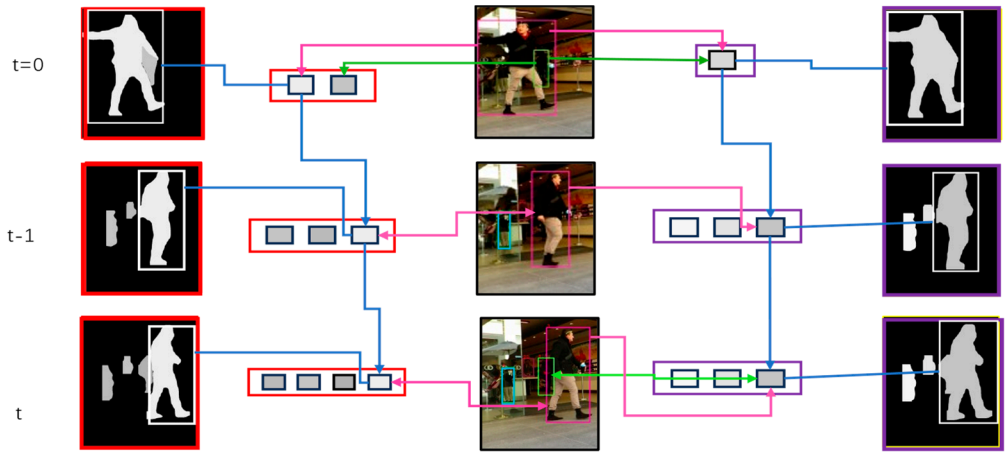

Figure 2. Target spatial-temporal features and region spatial-temporal features. The red bounding box is target-STF, the purple bounding box is region-STF.

The target-STF applies different grayscale values to mark targets. We obtain the bounding box of the target and find the grayscale values of the previous frame within the bounding box. Then, we can quickly find the targets that need to be matched based on these grayscale values. As shown in Figure 2, take the pink target as an example. At frame $t = 0$, assign an ID to the target and obtain the target-STF. At frame $t - 1$ and t, after matching the target-STF, obtain the target ID that needs to be matched by grayscale value, then perform target ID matching.

For re-identified targets, they were not detected and did not have the target-STF in the previous frame. Therefore, we proposed region-STF. The region-STF uses different grayscale values to mark regions. For a re-identified target, we find the region that the target previously belonged to by region-STF and perform target ID matching in this region. As shown in Figure 2, take the green target as an example. At frame $t = 0$, assign an ID to the target and obtain the region-STF. Store the successfully matched target ID in the region where the target is located. A target ID that has not been successfully matched for a long time will be removed from this region. At frame $t - 1$, the green target detection failed. At frame t, the green target is re-detected, and we use region-STF to find the area in which the green target belongs and match the green target with the target ID contained in this area. The region-STF effectively handles the problems caused by occlusion and improves the robustness of tracking.

3.3. Obtain Grayscale Spatial-Temporal Map

The process of obtaining spatial-temporal features is shown in Figure 3. In the image $\boldsymbol{PsI_t}$ and $\boldsymbol{RsI_t}$, for the convenience of observation, different grayscale values are chosen for the target.

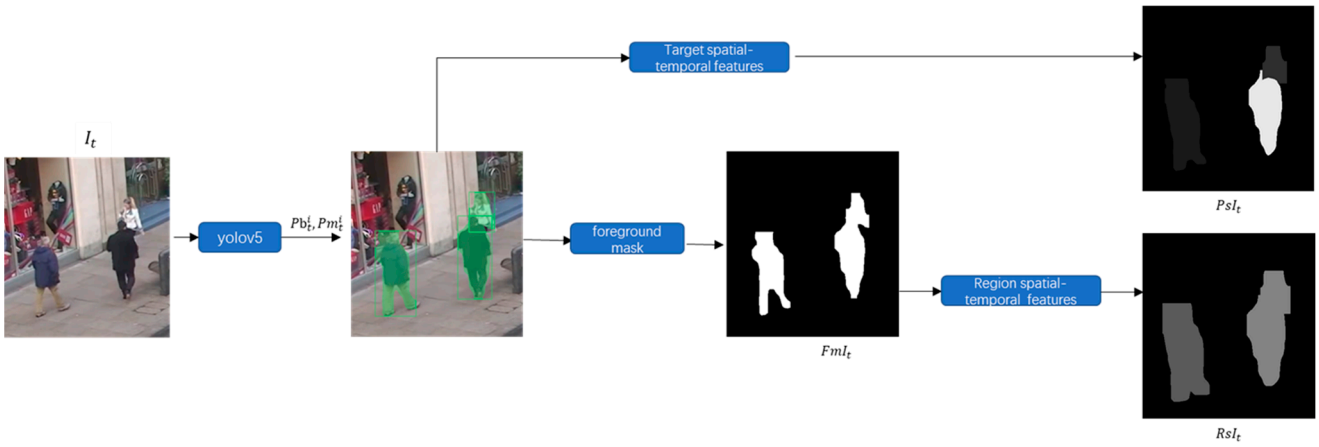

Figure 3. The process of obtaining spatial-temporal features. In the image $\boldsymbol{PsI_t}$ and $\boldsymbol{RsI_t}$, for the convenience of observation, different grayscale values are chosen for the target.

First, after inputting the current frame $\boldsymbol{I_t}$ into the yolov5 module, we obtain the bounding box and the segmentation mask of each target. Then, we apply these data into the foreground mask module and target-STF module to obtain the foreground mask and target-STF map, respectively. Finally, we use the foreground mask on the region-STF module to obtain the region-STF map.

In the target-STF module, first, we create a single-channel grayscale map $\boldsymbol{PsI_t}$ of the same size as frame $\boldsymbol{I_t}$ with a grayscale value of 0. Second, according to the bounding box and the segmentation mask, we copy the mask of each target to the grayscale map with different grayscale values using Equation (1):

$$\boldsymbol{gv_{Pm_t^i}(x,y) > 0: gv_{PsI_t}(x,y) = i + 1} \tag{1}$$

$\boldsymbol{gv_{pm_t^i}(x,y)}$ is the grayscale value of (x,y) in segmentation mask $\boldsymbol{Pm_t^i}$ and $\boldsymbol{gv_{PsI_t}(x,y)}$ is the grayscale value of (x,y) in map $\boldsymbol{PsI_t}$. After the operation is completed, we will obtain the target-STF map, and the grayscale value of target $\boldsymbol{p_t^i}$ is $i + 1$ in the target-STF map.

The foreground mask module is similar to the target spatial-temporal features module. First, we create a single-channel grayscale map $\boldsymbol{FmI_t}$ of the same size as $\boldsymbol{I_t}$ with a grayscale value of 0. Then, according to the bounding box and the segmentation mask, the mask of each target is copied to $\boldsymbol{FmI_t}$; with a gray value of 255, we will obtain the foreground mask $\boldsymbol{FmI_t}$.

In the region spatial-temporal features module, first, dilate the foreground mask, then fill it with different gray values in each white area in the foreground mask using Equation (2):

$$\boldsymbol{(x,y) \in cont_{FmI_t}^i,\ RsI_t(x,y) = i + 1} \tag{2}$$

$\boldsymbol{cont_{FmI_t}^i}$ is the set of contours in $\boldsymbol{FmI_t}$, and (x,y) is the point included in $\boldsymbol{cont_{FmI_t}^i}$. After the operation is completed, we will obtain the target-STF map.

3.4. Spatial-Temporal Feature Matching

The spatial-temporal feature matching process is shown in Figure 4. The spatial-temporal matching module includes two parts. It can be seen from the visualization results of $\mathbf{Ps_t^i}$ and $\mathbf{Rs_t^i}$. The red rectangular box is the set of target-STF. The green rectangular box is the set of region-STFs. The black rectangular box is the set of the targets.

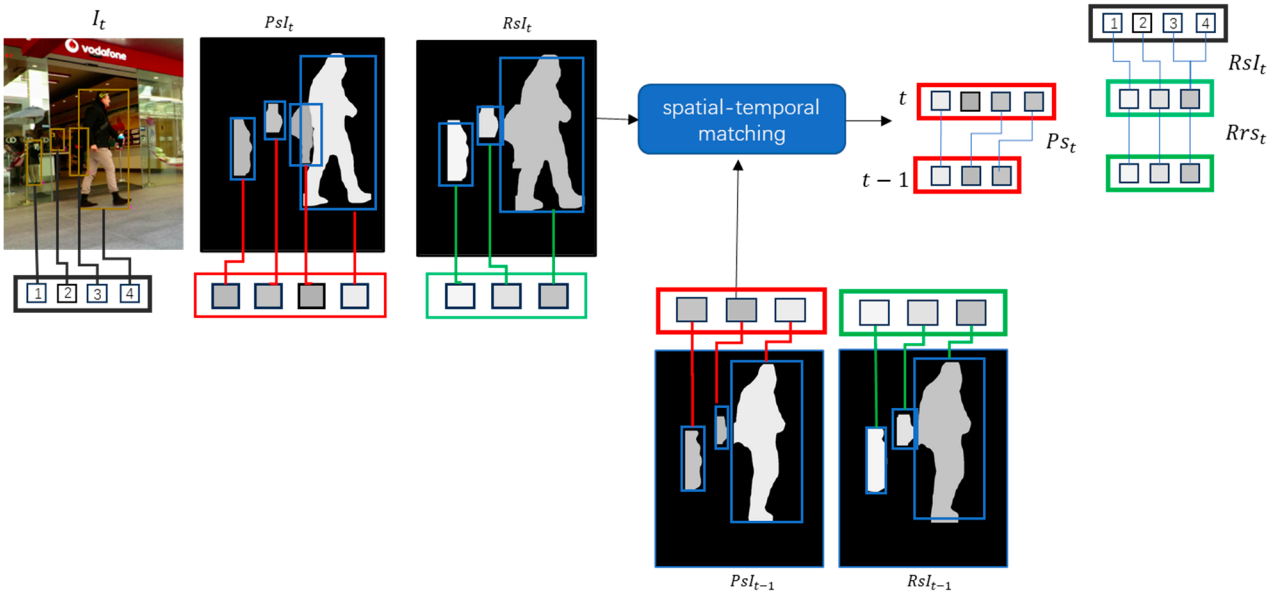

Figure 4. The spatial-temporal feature matching process. The blue rectangular box is the detected target. The red rectangular box is the set of target-STF. The green rectangular box is the set of region-STFs. The black rectangular box is the set of targets.

For the first part, we match the target-STF. According to the number of targets $\boldsymbol{P_t^i}$, we create the set $\boldsymbol{Ps_{t,i+1}^k}$. The grayscale value in the target-ST map of target $\boldsymbol{p_t^j}$ is $j+1$, so the target-STF matching results of target $\boldsymbol{p_t^j}$ is $\boldsymbol{ps_{t,j+1}^k}$.

Then, select different points $\boldsymbol{D_i}(x,y)$ based on the bounding box of each target using Equation (3):

$$\boldsymbol{D_i(x,y) = S(h_i, w_i, v_i, s_i)} \tag{3}$$

h_i is the height of $\boldsymbol{Pb_t^i}$, w_i is the width of $\boldsymbol{Pb_t^i}$, v_i is the velocity of $\boldsymbol{CPb_t^i}$, and s_i is the area of $\boldsymbol{Pb_t^i}$. $\boldsymbol{S}(h_i, w_i, v_i, s_i)$ is a function to obtain points $\boldsymbol{D_i}(x,y)$.

Then, obtain the grayscale values of these points $\boldsymbol{D_i}(x,y)$ in the target-STF map $\boldsymbol{PsI_{t-1}}$. If the grayscale values of these points are all zero, zero is stored in the set $\boldsymbol{Ps_{t,i}^k}$. If there is a non-zero gray value, the non-zero gray value is stored in the set $\boldsymbol{Ps_{t,i}^k}$, and the same value is recorded only once. After the operation is completed, we will obtain the target-STF matching results $\boldsymbol{Ps_{t,i}^k}$.

For the second part, we match the region spatial-temporal features. This process is similar to the first part; according to the number of target-STF created, set $\boldsymbol{Rrs_t^i}$ to record the region corresponding relationship between two frames. We create set $\boldsymbol{Rs_t^i}$ to record to which region the target belongs. $\boldsymbol{Rs_t^i}$ can be obtained from Equation (4):

$$\boldsymbol{rs_t^i = gv_{RsI_t}(D_i(cty, ctx)), 0 \leq i < P_t.size.} \tag{4}$$

$\boldsymbol{D_i}(cty, ctx)$ is the center point of the bounding box of target $\boldsymbol{p_t^i}$. After the operation is completed, we know the region to which the target belongs.

Last, if the target-STF matching results $\boldsymbol{ps_{t,j+1}^k}$ of target $\boldsymbol{p_t^j}$ are not zero, for target $\boldsymbol{p_t^j}$, select the same points $\boldsymbol{D_i}(x,y)$ as in the first part. Obtain the grayscale values of these points in the target-STF map $\boldsymbol{RsI_{t-1}}$, and store the non-zero and non-repeating grayscale values in $\boldsymbol{Rrs_{t,rs_t^j}^k}$. After this operation is completed, we will obtain $\boldsymbol{Rrs_{t,i}^k}$. According to $\boldsymbol{rs_t^j}$ and $\boldsymbol{Rrs_{t,rs_t^j}^k}$, the region to which target $\boldsymbol{p_t^j}$ belongs in the previous frame can be obtained.

3.5. Matching and Feature Extraction

The ID matching and feature extraction process is shown in Figure 5. The green box represents the continuously detected target. The blue box represents the re-detected target and new target.

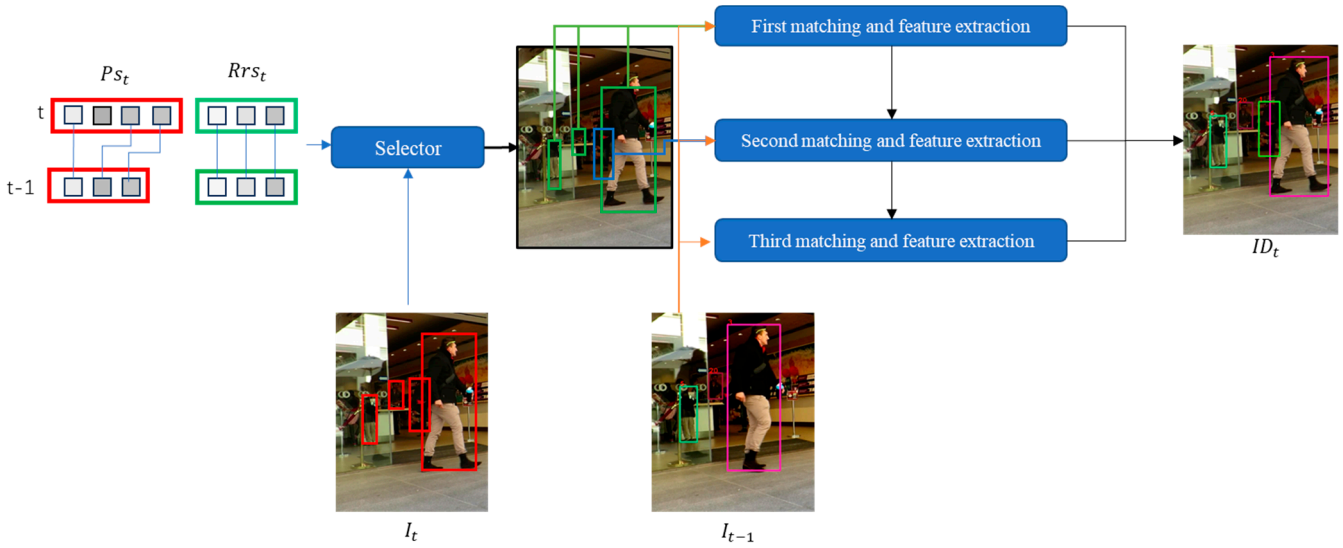

Figure 5. The ID matching process. The red box represents the detected target. The green box represents the continuously detected target. The blue box represents re-detected target or new target.

After obtaining the matching results of these two spatial-temporal features, we can judge whether a target is detected in both the current frame and the previous frame according to $\boldsymbol{Ps}_{t,i}^{k}$:

$\boldsymbol{Ps}_{t,i}^{k}, k > 0$, re-detected target or new target,

$\boldsymbol{Ps}_{t,i}^{k}, k = 0$, continuously detected target.

For the continuously detected target, we input it into the first matching module for priority matching. For the continuously detected target, the target does not need to be matched with all targets in the previous frame. We can quickly obtain the ID set $\boldsymbol{Ids}_{t,i}^{k}$ that the target needs to match using Equation (5):

$$\boldsymbol{ps}_{t,i+1}^{k}, \boldsymbol{k > 0}, \boldsymbol{Ids}_{t,i}^{k} = \boldsymbol{PcId}_{t-1}^{\boldsymbol{ps}_{t,i+1}^{k}-1} \tag{5}$$

$\boldsymbol{PcId}_{t}^{i}$ records the ID for each target $\boldsymbol{P}_{t}^{i}$ in the t frame, and the gray value of the target-STF of each target $\boldsymbol{p}_{t}^{j}$ is j+1 in frame t. Therefore, when we know the $\boldsymbol{ps}_{t,j+1}^{k}$ and $\boldsymbol{PcId}_{t-1}^{i}$, we can quickly obtain the ID set $\boldsymbol{Ids}_{t,j}^{k}$ that the target needs to match. According to $\boldsymbol{Ids}_{t,i}^{k}$, prioritize matching with k values equal to 1 during matching. We only need to calculate the IoU without calculating the appearance features and motion features when matching the ID for a continuously detected target. The probability $\boldsymbol{p}$ can be obtained by Equation (6):

$$\boldsymbol{p} = \boldsymbol{Max}\left(\boldsymbol{U}\left(\boldsymbol{p}_{t}^{i}, \boldsymbol{Ids}_{t,i}^{k}\right)\right) \tag{6}$$

$U\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}^{k}\right)$ is a function to obtain the IoU. If the $\boldsymbol{p}$ is greater than the set threshold, the match is successful, and then, we update the trajectory of the target and record the ID for target $\boldsymbol{P}_{t}^{i}$ and region using $\boldsymbol{PcId}_{t}^{i}$ and $\boldsymbol{RcID}_{t,i}^{k}$. If the $\boldsymbol{p}$ is less than the threshold, then put this target into the second stage of matching. According to the different videos, we can choose a different threshold.

For re-detected targets, new targets, and continuously detected targets that failed to match in the first stage, we input them into the S matching and feature extraction module

for second matching. We need to use appearance features and motion features in this module. The ID set $\boldsymbol{Ids}_{t,i}^{k}$ that the target needs to match can be obtained by Equation (7):

$$\boldsymbol{Ids}_{t,i}^{k} = \boldsymbol{RID}^{k}_{\boldsymbol{t-1},\boldsymbol{Rrs}_{t}^{Rs_{t,i}}} \cup \boldsymbol{RcID}^{k}_{\boldsymbol{t-1},\boldsymbol{Rrs}_{t}^{Rs_{t,i}}} \tag{7}$$

We find the region where the target belongs in the previous frame according to the region-STF matching results: $\boldsymbol{Rs}_{t,i}$ and $\boldsymbol{Rrs}_{t}^{i}$. According to $\boldsymbol{Ids}_{t,i}^{k}$, if $k = 0$, the target $\boldsymbol{p}_{t}^{i}$ does not belong to the region in the previous frame, so put this target $\boldsymbol{p}_{t}^{i}$ into the next stage for matching. After we obtain $\boldsymbol{Ids}_{t,i}^{k}$, we can do ID matching based on appearance features and motion features. The probability $\boldsymbol{p}$ can be obtained by Equation (8):

$$\boldsymbol{p} = \boldsymbol{Max}\begin{pmatrix} \boldsymbol{w_1} * \boldsymbol{U}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}\right) + \boldsymbol{w_2} * \boldsymbol{HG}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}\right) \\ + \boldsymbol{w_3} * \boldsymbol{MT}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}\right) \end{pmatrix} \tag{8}$$

w_1, w_2, w_3 are the weights of each feature.

$\boldsymbol{U}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}\right)$ is a function to obtain the IoU. $\boldsymbol{HG}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}^{k}\right)$ is a function to compare two histograms. $\boldsymbol{MT}\left(\boldsymbol{P}_{t}^{i}, \boldsymbol{Ids}_{t,i}^{k}\right)$ is a function to obtain the motion features by KF.

If the p is greater than the set threshold, the match is successful, and then, we update the trajectory of the target and record the ID for the target $\boldsymbol{P}_{t}^{i}$ and region by $\boldsymbol{PcId}_{t}^{i}$ and $\boldsymbol{RcID}_{t,i}^{k}$. If the match fails, we put this target in the last stage to match.

In the third matching and feature extraction module, the matching process is similar to the second stage, except for the difference in the matching range. The matching range of the target in the second stage is in the region where the target belongs, while the third stage matches outside of this range. Therefore, we need to use $\boldsymbol{UID}_{t}^{i}$:

$$\boldsymbol{Ids}_{t,i}^{k} = \boldsymbol{UID}_{t-1}^{i} \tag{9}$$

The matching method is the same as Equation (8). But when the probability p is greater than the set threshold, the matching stops. Then, we update the trajectory of the target and record the ID for the target $\boldsymbol{p}_{t}^{i}$ and region by $\boldsymbol{PcId}_{t}^{i}$ and $\boldsymbol{RcID}_{t,i}^{k}$. If a target fails to match, we regard this target as a new target and record the ID for the target $\boldsymbol{p}_{t}^{i}$ and region using $\boldsymbol{PcId}_{t}^{i}$ and $\boldsymbol{RcID}_{t,i}^{k}$.

Finally, after all matches are completed, store the unmatched target IDs of the previous frame into $\boldsymbol{RID}_{t,i}^{k}$ or $\boldsymbol{UID}_{t}^{i}$. If a region disappears, the $\boldsymbol{RID}_{t-1,i}^{k}$ of this region is stored in $\boldsymbol{UID}_{t}^{i}$. For a re-detected target, the ID of this target is removed from the sets $\boldsymbol{RID}_{t,i}^{k}$ and $\boldsymbol{UID}_{t}^{i}$. Also, in the sets $\boldsymbol{RID}_{t,i}^{k}$ and $\boldsymbol{UID}_{t}^{i}$, after some frames, this ID is eliminated.

4. Results

4.1. Datasets, Evaluation Metrics, Environment, and Implementation Details

Datasets. We conduct the experiments on the MOT16 pedestrian tracking dataset. MOT16 contains seven training sequences and seven test sequences, and these videos were captured in high-density scenes with heavy occlusion. The video frame rate is around 30 FPS.

Evaluation metrics. We use the official evaluation tools. These evaluation tools contain many evaluation metrics. There are two most used tracking performance evaluation metrics, which are multiple object tracking accuracy (MOTA) and IDF1.

MOTA focuses more on the detection side, while IDF1 can better reflect the tracking side. IDF1 reflects the tracker ability to maintain identities over time and emphasizes association accuracy.

In addition, the percentage of mostly tracked targets (MT) and the percentage of mostly lost targets (ML) are also commonly used metrics.

TFPS is tracking FPS, which does not consider the detection time.

Environment. The proposed approach is implemented in visual studio 2019, and the program is performed on a laptop with a CPU i7-11800H and GPU RTX3060 laptop.

Implementation Details. We use the public detector deformable parts model (DPM) provided by MOT16 to detect the target, and yolov5 is only used for target cutting.

4.2. Ablation

To demonstrate the efficiency of our method, we employ a conventional tracking module as a baseline, which uses the public detector DPM, yolov5seg, appearance features, and motion features for tracking. Yolov5 is only used for target cutting. Baseline + GSP adds the grayscale spatial-temporal features (GSP). The tracking algorithms are performed in the CPU without GPU accelerating. The ablation experiment results are shown in Table 2.

Table 2. Ablation experiment results on MOT16. ↑ indicates that higher is better, ↓ indicates that lower is better. The best results are shown in **bold**.

Module	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	TFPS ↑
Baseline	20.5	13.6	7.4	51.6	15.4
Baseline + GSP	**46.7**	**52.0**	**21.5**	**34.1**	**17.6**

As can be seen from Table 2, after adding grayscale spatial-temporal features, everything improved. The TFPS increased to 17.6, and the FPS not only did not decrease but increased. This is because, in the TBD paradigm, the number of IDs often increases when the detection time increases, which will result in more time to match the ID. This is the case for the baseline method, which spends a lot of time on ID matching. After adding grayscale spatial-temporal features, most targets only need to match between two IDs, which greatly reduces the matching time. It can be seen that our algorithm reduces the time of tracking, and the tracking accuracy is improved. At the same time, the accuracy is also improved. First, because the range of target matching is reduced, the interference of similar targets is removed. Second, as shown in Figure 6, the grayscale spatial-temporal features are robust for long-term occlusion.

Grayscale spatial-temporal features can solve the problems of ID switching and tracking target loss caused by occlusion. As shown in Figure 6, our method can effectively handle occlusion. The top row is the visualization result of the tracker on MOT16-09. The bottom row is the visualization result of the tracker on MOT16-02. When matching, we give priority to matching continuously detected targets and, secondly, match re-detected targets, which effectively avoids ID switching between targets. When occlusion occurs, the occluded target is recorded in the region where it belongs. When the occluded target is re-detected, matching IDs are found based on the region where the target belongs. This reduces the interference of other similar targets.

4.3. Evaluation Comparison with Other State-of-the-Art Trackers

We compare the performance of our tracker with other trackers on the MOT16 benchmark. Table 3 lists the performance of our method on MOT16; also, comparisons with other models are listed.

We can find that the performance of trackers based on private detectors is better than those based on public detectors. Detectors are divided into public detectors provided by the dataset and private detectors customized by the algorithm. The public detector provided by the MOT16 dataset is the deformable parts model (DPM). These private detectors can be trained by yourself or the state-of-the-art detectors proposed by others. The reason is that these private detectors obtain more accurate detection results compared to public detectors. Tracking performance tends to improve with better detection quality; thus, tracking performance can be improved by using state-of-the-art detectors. But data association is the main part of tracking, so we use public detector DPM. Compared to the methods using a public detector in Table 3 on the MOT16 benchmark, our method achieved

the best score in metric ML and metric MT. The ML and MT metrics reached 21.5 and 34.1, respectively. This means that we successfully tracked more targets and lost fewer targets than the other methods. In metric TFPS, the tracking algorithm of ours was performed using a CPU without GPU accelerating, but it was better than the other trackers, which reached 17.6. We only used traditional methods for feature extraction and ID matching, and the MOTA and IDF1 scores were comparable to these state-of-the-art trackers. Our metric MOTA was only 0.5 lower than the highest metric MOTA of AMIR in Table 3 but higher than the other methods. Our metric IDF1 was only lower than the metrics of DMAN and DM but higher than the other methods. Therefore, it can be concluded that our method maintains tracking accuracy while improving operational efficiency.

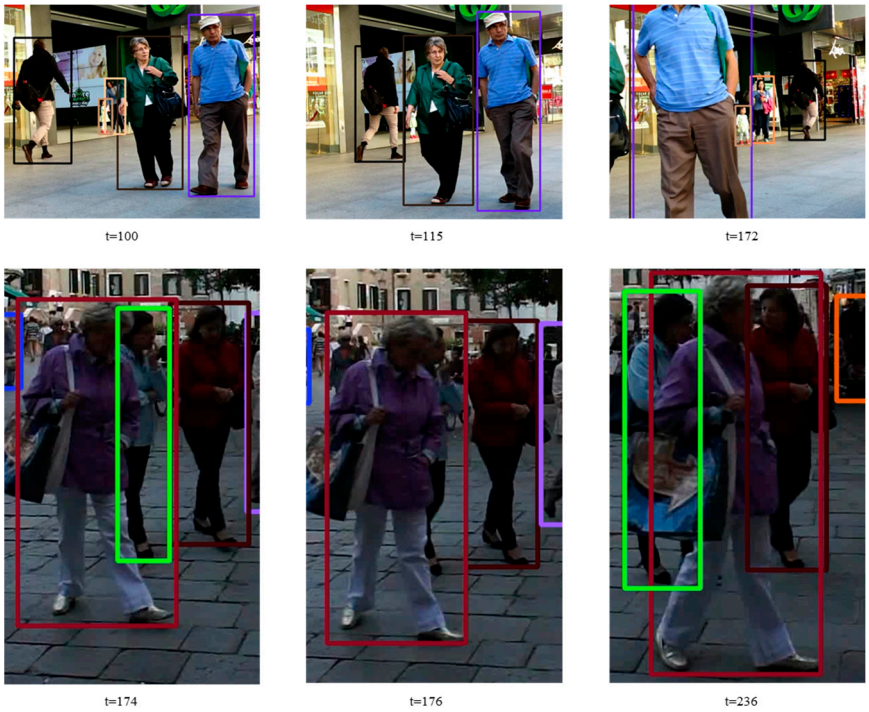

Figure 6. Examples of long-term occlusion. The top row is the visualization result of the tracker on MOT16-09. The bottom row is the visualization result of the tracker on MOT16-02.

Table 3. Comparison of multi-object tracking methods on MOT16. ↑ indicates that higher is better, ↓ indicates that lower is better. The best results are shown in **bold**.

Module	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	TFPS ↑
MotionTrack [9] (private)	81.1	80.1	-	-	-
FairMOT [26] (private)	68.7	70.4	39.5	19.0	-
MOTRv2 [21] (private)	78.6	75.0	-	-	-
STAM16 [27]	46.0	50.0	14.6	43.6	0.2
DMAN [28]	46.1	**54.8**	17.4	42.7	0.3
DASOT [29]	46.1	49.4	14.6	41.6	9.0
TAMA [30]	46.2	49.4	14.1	44.0	6.3
DM [31]	46.5	52.3	-	35.6	10.1
AMIR [32]	**47.2**	46.3	14.0	41.6	1.0
Ours	46.7	52.0	**21.5**	**34.1**	**17.6**

5. Conclusions

We propose a novel method on tracking via traditional methods, which are more suitable to run on monitors than deep learning methods. Compared to deep learning methods, our method has very low requirements for the computing power. Using this

method, we obtain target spatial-temporal features and region spatial-temporal features by grayscale mapping. Our spatial-temporal features are simple and fast to match between two frames, and we reduce matching times by more than three times for most targets by using these two spatial-temporal features. This can reduce unnecessary ID matching, which solves the problem of taking some time to calculate when matching IDs. Our proposed method is based on traditional methods, so our approach does not require data to train. Especially for the tracking-by-detection paradigm, if using our method, it will not only reduce some feature extraction calculations but also reduce some unnecessary ID matching. Our method maintains or even improves tracking accuracy while reducing the computations, as demonstrated on the MOT16 benchmark set. For future works, first, we need to further reduce the number of computations while ensuring accuracy. We will study how to effectively extract the appearance features of the target through the histogram without using the yolov5seg cutting function. Second, during the night, motion features and grayscale spatiotemporal features can still be used, but the appearance features extracted by a color histogram are useless. Therefore, we will study how to extract effective appearance features from infrared cameras in the night. We hope that our work can help MOT research to achieve real-time performance.

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

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in MOT16 at https://motchallenge.net/data/MOT16/, accessed on 1 July 2022, reference number [7]. These data were derived from the following resources available in the public domain: https://motchallenge.net/data/MOT16/, accessed on 1 July 2022.

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

References

1. Redmon, J.; Divvala, S.; Divvala, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.
2. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems, Montreal, QC, USA, 7–12 December 2015; Volume 28.
3. Liu, W.; Anguelov, D.; Erhan, D. SSD: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherland, 11–14 October 2016; pp. 21–37.
4. Bewley, A.; Ge, Z.; Ott, L. Simple online and realtime tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
5. Wang, Z.; Zheng, L.; Liu, Y. Towards real-time multi-object tracking. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020; pp. 107–122.
6. Bergmann, P.; Meinhardt, T.; Leal-Taixe, L. Tracking without bells and whistles. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 941–951.
7. Milan, A.; Leal-taixé, L.; Reid, I.; Roth, S.; Schindler, K. MOT16: A benchmark for multi-object tracking. *arXiv* **2016**, arXiv:1603.00831.
8. Cao, J.; Pang, J.; Weng, X. Observation-centric sort: Rethinking sort for robust multi-object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 9686–9696.
9. Qin, Z.; Zhou, S.; Wang, L. Motiontrack: Learning robust short-term and long-term motions for multi-object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 17939–17948.
10. Yu, F.; Li, W.; Li, Q. POI: Multiple object tracking with high performance detection and appearance feature. In Proceedings of the Computer Vision–ECCV 2016 Workshops, Amsterdam, The Netherlands, 11–14 October 2016; pp. 36–42.

11. Szegedy, C.; Liu, W.; Jia, Y. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
12. LIU, H.; Yang, X.; Latecki, L. Dense Neighborhoods on Affinity Graph. *Int. J. Comput. Vis.* **2012**, *98*, 65–82. [CrossRef]
13. LI, Z.; Cai, S.; Cai, X. Multiple Object Tracking with GRU Association and Kalman Prediction. In Proceedings of the International Joint Conference on Neural Networks, Virtual, 18–22 July 2021; pp. 1–8.
14. Yang, M.; Han, G.; Yan, B. Hybrid-sort: Weak cues matter for online multi-object tracking. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 6504–6512.
15. Lu, Z.; Rathod, V.; Votel, R. Retinatrack: Online single stage joint detection and tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 14668–14678.
16. Lin, T.; Goyal, P.; Girshick, R. Focal loss for dense object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
17. Jin, Y.; Gao, F.; Shuang, F. Multi-Object Tracking: Decoupling Features to Solve the Contradictory Dilemma of Feature Requirements. *IEEE Trans. Circuits Syst. Video Technol.* **2023**, *33*, 5117–5132. [CrossRef]
18. Vaswani, A.; Shazeer, N.; Parmar, N. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5999–6009.
19. Meinhardt, T.; Kirillov, A.; Leal-taixe, L. Trackformer: Multi-object tracking with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8844–8854.
20. Carion, N.; Massa, F.; Synnaeve, G. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Virtual, 23–28 August 2020; pp. 213–229.
21. Zhang, Y.; Wang, T.; Zhang, X. MOTRv2: Bootstrapping end-to-end multi-object tracking by pretrained object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 22056–22065.
22. Zeng, F.; Dong, B.; Zhang, Y. MOTR: End-to-end multiple-object tracking with transformer. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; pp. 659–675.
23. Ballester, I.; Fontán, A.; Civera, J. DOT: Dynamic Object Tracking for Visual SLAM. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation, Xi'an, China, 30 May–5 June 2021; pp. 11705–11711.
24. Yu, M.; Diao, H.; Ling, X. Online multi-object tracking method based on trajectory mask. *J. Shandong Univ.* **2023**, *53*, 61–69.
25. Saleh, F.; Aliakbarian, S.; Gould, S. Probabilistic Tracklet Scoring and Inpainting for Multiple Object Tracking. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 14324–14334.
26. 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**, *129*, 3069–3087. [CrossRef]
27. Chu, Q.; Ouyang, W.; Li, H.; Wang, X.; Liu, B.; Yu, N. Online Multi-object Tracking Using CNN-Based Single Object Tracker with Spatial-Temporal Attention Mechanism. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 4846–4855.
28. Zhu, J.; Yang, H.; Liu, N.; Kim, M. Online Multi-Object Tracking with Dual Matching Attention Networks. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 379–396.
29. Chu, Q.; Ouyang, W.; Li, H.; Wang, X.; Liu, B.; Yu, N. DASOT: A Unified Framework Integrating Data Association and Single Object Tracking for Online Multi-Object Tracking. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 10672–10679.
30. Yoon, Y.; Kim, D.Y.; Song, Y. Online multiple pedestrians tracking using deep temporal appearance matching association. *Inf. Sci.* **2021**, *561*, 326–351. [CrossRef]
31. Sun, Z.; Chen, J.; Mukherjee, M. Online multiple object tracking based on fusing global and partial features. *Neuro Comput.* **2022**, *470*, 190–203. [CrossRef]
32. Sadeghian, A.; Alahi, A.; Savarese, S. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 300–311.