electronics

MDPI

Article

A Multi-Object Tracking Approach Combining Contextual Features and Trajectory Prediction

Peng Zhang [1,†], Qingyang Jing [1,*,†], Xinlei Zhao [2], Lijia Dong [2], Weimin Lei [1], Wei Zhang [1] and Zhaonan Lin [1]

[1] School of Computer Science and Engineering, Northeastern University, Shenyang 110169, China; 1910609@stu.neu.edu.cn (P.Z.); leiweimin@mail.neu.edu.cn (W.L.); zhangwei1@mail.neu.edu.cn (W.Z.); 1910692@stu.neu.edu.cn (Z.L.)

[2] Shenyang Er Yi San Electronic Technology Co., Ltd., Shenyang 110023, China; zhaoxinlei@china213.net (X.Z.); donglijia@china213.net (L.D.)

* Correspondence: 2110649@stu.neu.edu.cn

† These authors contributed equally to this work.

Abstract: Aiming to solve the problem of the identity switching of objects with similar appearances in real scenarios, a multi-object tracking approach combining contextual features and trajectory prediction is proposed. This approach integrates the motion and appearance features of objects. The motion features are mainly used for trajectory prediction, and the appearance features are divided into contextual features and individual features, which are mainly used for trajectory matching. In order to accurately distinguish the identities of objects with similar appearances, a context graph is constructed by taking the specified object as the master node and its neighboring objects as the branch nodes. A preprocessing module is applied to exclude unnecessary connections in the graph model based on the speed of the historical trajectory of the object, and to distinguish the features of objects with similar appearances. Feature matching is performed using the Hungarian algorithm, based on the similarity matrix obtained from the features. Post-processing is performed for the temporarily unmatched frames to obtain the final object matching results. The experimental results show that the approach proposed in this paper can achieve the highest MOTA.

Keywords: contextual features; trajectory prediction; trajectory matching; similarity matrix; preprocessing; postprocessing

Citation: Zhang, P.; Jing, Q.; Zhao, X.; Dong, L.; Lei, W.; Zhang, W.; Lin, Z. A Multi-Object Tracking Approach Combining Contextual Features and Trajectory Prediction. *Electronics* **2023**, *12*, 4720. https://doi.org/10.3390/electronics12234720

Academic Editor: Dong Zhang

Received: 27 September 2023
Revised: 21 October 2023
Accepted: 25 October 2023
Published: 21 November 2023

1. Introduction

Object tracking, which combines digital image processing and machine learning, has long been of interest to researchers [1]. It is important in military planning, medical treatment, and even all aspects of life, and has been widely recognized and applied in many fields [2,3]. Research on object tracking has been ongoing since the 1980s. The initial object tracking algorithms primarily focused on analyzing a solitary object within a video and needed to calculate the position and size of the specified object in each frame of the video or image sequence, which is generally called single-object tracking or visual object tracking. Multi-object tracking aims to track all specific types of objects on the screen. In single-object tracking, a particular object is selected for tracking, and its initial position and size are provided in the first frame. The objective of this task is to devise an appearance model or motion model for the object, addressing challenges such as variations in shape and scale, alterations in the background, changes in speed, and occlusions encountered during the tracking process. Conversely, in multi-object tracking, the emphasis lies in determining the number of objects to be tracked while ensuring their individual identities remain distinct [4]. Pedestrians are the most important research object in multi-object tracking and have the characteristics of non-rigidity, slow speed, density, etc. In crowded scenes such as subway stations, shopping malls, and squares, accurate and efficient pedestrian tracking is a rather difficult task, and the study of real-time pedestrian tracking in crowded scenes is

very challenging and of practical value. The occlusion problem is a difficult problem in multi-object tracking, especially in crowded scenes where occlusion between pedestrians is especially frequent. At the time of intersection, just before object occlusion, the similarity of features in the object area is high, making it susceptible to confusion and resulting in the identification of multiple objects as a single entity, ultimately leading to tracking failure. Following object occlusion, the task of multi-object tracking entails not only the immediate detection of the object once it reappears, but also the preservation of the occluding object's identity consistency. In the process of multi-object tracking, new objects will appear, and the newly emerged objects lack historical information modeling. If the newly emerged objects cannot be detected, there will be missed detection, and the subsequent follow-up will be affected [5]. Simultaneously, when an object goes out of the field of view, failure to process the tracking frame within the tracking algorithm can lead to an erroneous focus of the frame on the background, resulting in tracking errors. Furthermore, in multi-object tracking, it is crucial to differentiate between the scenarios of object disappearance and brief occlusion. Additionally, the tracking speed is the standard parameter used to measure whether it can meet the real-time requirements of the algorithm, and also to guarantee the fluency of the video. Object tracking requires the algorithm to be able to replace eyes to overcome more advanced visual challenges [6]. Most of the tracking algorithms found in the current research are in the stage of theoretical analysis, and further research and discussion are still needed to apply them in the actual detection scenarios to meet the needs of social progress. The research in this area is of great importance and of theoretical and practical significance.

The graph neural network was first proposed in 2005 [7], which aggregates the information of neighboring nodes on object nodes through a recurrent neural architecture, but it is not applicable in large-scale graph models because of the large computational effort required [8]. Aiming to solve the problems of the above existing approaches, this paper uses the graph model to extract the contextual features of object appearances, adds pre-processing and post-processing, and integrates the object appearance information as well as the motion information. The innovations are as follows:

(1) During the tracking process, both the appearance and motion information of the object are taken into account.
(2) Contextual features as well as individual features are used to better represent similar-looking objects.
(3) The graph model uses different weights depending on the distance between objects.
(4) A preprocessing module is applied to categorize the object velocities and exclude unnecessary connections between the graph model objects.
(5) Post-processing of temporarily unmatched trajectories is conducted to reduce the impact of occlusion and other factors on the object tracking results.

The rest of this paper is organized as follows: The second part introduces the common multi-object tracking approaches, which are divided into traditional multi-object tracking and object detection-based multi-object tracking approaches. The third section introduces the extraction of the motion and appearance features after preprocessing. The fourth section introduces the object association matching approach and the post-processing technique for temporarily unmatched frames. The fifth section presents the analysis of the experimental results, and the sixth section summarizes the paper, presenting the limitations of the current approach and discussing future research.

2. Related Works

2.1. Traditional Multi-Object Tracking Methods

The initial multi-object tracking algorithms used traditional tracking techniques such as particle filtering [9], Meanshift [10], and feature-point-based optical flow approaches. The optical flow method is a classical motion estimation algorithm. The concept of optical flow was first proposed by Gibson in 1950, and it describes the instantaneous speed of motion of pixels of space-moving objects on the observed imaging plane. In 1981, Horn

and Schunck connected the two-dimensional velocity field with the gray level of the image, added the global smoothing hypothesis on the basis of the basic constraint equation of optical flow, and proposed the basic model of optical flow calculation, which laid the foundation for the theoretical research and mathematical derivation of the variational method for the estimation of optical flow. In general, the optical flow method needs to meet three assumptions: (1) the brightness between adjacent frames is constant; (2) the movement of objects between adjacent frames is small; and (3) spatial consistency, that is, pixels within a certain neighborhood have the same motion. But these approaches are not well adapted to the complex tracking process; therefore, the technique of correlation filtering was developed. The cyclic matrix-based kernel tracking algorithm [11] (CSK) was proposed to solve the problem of intensive sampling during data acquisition and to speed up detection using Fourier transform [12]. Then, the KCF algorithm and its derivative algorithms were successively proposed. With the gradual increase in feature and scale information considered, the computation volume also gradually increases, but in general, the algorithms in this category have better real-time performance [13]. However, the tracking accuracy is lower when the object moves faster and there are large deformations.

2.2. Multi-Object Tracking Method Based on Object Detection

(1) Tracking based on appearance features

Firstly, the appearance features of all objects in each frame are utilized to detect and obtain the corresponding detection boxes [14]. Subsequently, the detection box in the current frame is matched with the tracking box from the previous frame [15]. Leveraging the impressive feature representation capabilities of deep neural networks can significantly enhance the performance of multi-object tracking, both simply and effectively. The first multi-object tracking method based on deep learning was proposed by Wang et al. in 2014 [16]. This method leverages a two-layer neural network to extract appearance features, subsequently computing the similarity based on appearance. Ultimately, it transforms the multi-object tracking problem into a minimum spanning tree problem for resolution. Experiments show that the appearance features extracted based on a simple two-layer neural network greatly improve the performance of multi-object tracking [17]. With the rapid development of deep learning technology in recent years, most of the current multi-object tracking methods use the appearance feature extraction method based on convolutional neural networks. For example, after Kim et al. integrated CNN features into the classical multi-hypothesis tracking method, its multi-object tracking performance jumped to first place in the MOT15 dataset at that time. The appearance feature extraction of some multi-object tracking methods uses ResNet, GoogLeNet, pedestrian recognition, and other different types of depth features.

The result of object detection will seriously affect the effect of subsequent tracking. With the progress of deep learning technology in recent years, object detection techniques such as YOLO and Faster RCNN can meet most of the needs both in terms of accuracy and speed [18]. The association matching can be performed by calculating the Intersection over Union (IOU) ratio between objects or the distance between the centroids of object boxes in two adjacent frames [19]. However, when the object has large deformation or moves fast, the association is often accomplished by constructing a deep network to learn the matching score directly or by comparing the distance of the deep features of the object [20].

(2) Tracking based on trajectory prediction

Common trajectory prediction approaches include Kalman filtering [21] and LSTM. In 2016, Bewley et al. proposed the Sort algorithm, which adopts the Faster R-CNN algorithm for object detection and uses the intersection ratio between the predicted trajectory and the current frame detection results by Kalman filter to perform Hungarian matching to complete the data association. Some multi-object tracking methods use recurrent neural networks to extract appearance features or motion features. For example, Maksai et al. considered both the appearance features based on the ReID model and the appearance features based on the LSTM [22]. There are also some multi-eye tracking methods that

use twin networks to extract deeper features with more discriminating power [23]. The characteristics learned make it easier to distinguish between different goals. Kim et al. also proposed to use a twin network to monitor the depth features of the learning object [24], in which the two images to be matched, the intersection ratio, and the area ratio are input, and the matching loss between the two images is output [25]. Finally, the similarity score between the two objects is determined by the Euclidean distance based on the above depth features, the intersection ratio of the object frame, and the area ratio.

Because the Sort algorithm only uses simple IOU for data association, it will cause a large number of identity switches in crowded scenarios. In order to solve this problem, they also proposed the Deepsort algorithm, which added cascade matching on the basis of the IOU matching of the Sort algorithm to form a two-level data association mechanism. Meanwhile, a pre-trained pedestrian re-recognition network was used to extract the appearance features of the object to assist data association, effectively solving the identity switching problem in the case of occlusion. It lays the foundation for subsequent research. The above methods of trajectory matching based only on detection results are greatly affected by the accuracy of the detector, and the tracking performance will be significantly reduced when the detector has unreliable detection, such as missing or false detection. For example, Bytetrack uses the Kalman trajectory prediction approach, which treats the high and low scoring boxes separately according to the detection box score [26]. Initially, the high-scoring boxes are matched with pre-existing tracked trajectories to assign identities to each high-scoring box. Subsequently, the low-scoring boxes are matched with the tracking trajectories that were not initially associated with the high-scoring boxes, effectively extracting the true objects from the low-scoring boxes.

In this paper, trajectory prediction is combined with object contextual features, i.e., both appearance and motion features of the object are employed for better object tracking.

3. Feature Extraction Methods

3.1. Kalman Trajectory Prediction

The Kalman filter is a powerful filter that uses an autoregressive model to estimate and predict the state of a system in the presence of continuous inputs containing noise. By observing the measured value of the system state, the Kalman filter constantly updates the prediction results and thus optimally estimates the system state. The Kalman filter is quite effective in predicting the position of an object moving regularly and has certain characteristics of anti-noise information interference.

The Kalman filter consists of two parts: prediction and update [27].

1. The predicted value of the state in the $k-1$ frame and the predetermined equations of motion are used to estimate the state of the object, and the estimated value of the state of the object is obtained.

   (1) Calculate the predicted values:

   $$\hat{x}_{\overline{k}} = A_k \hat{x}_{k-1} + B_k u_k \tag{1}$$

   (2) Calculate the a priori covariance matrix between the true and predicted values at the current moment:

   $$P_{\overline{k}} = A_k P_{k-1} A_k^T + Q_k \tag{2}$$

2. Correct the state estimate with the current moment's detection result to obtain the final object state prediction.

   (3) Calculate the Kalman gain:

   $$K_k = P_{\overline{k}} H_k^T / H_k P_{\overline{k}} H_k^T + R \tag{3}$$

   (4) Estimate the true state of the object based on measured values and calculate predicted values:

   $$\hat{x}_k = \hat{x}_{\overline{k}} + K_k(z_k - H_k \hat{x}_{\overline{k}}) \tag{4}$$

(5) Calculate a posteriori covariance matrix for the error between the true and estimated values:

$$P_k = (I - K_k H_k) P_{\overline{k}} \tag{5}$$

where the subscript k denotes the current frame, $k-1$ denotes the previous frame, $\overline{k}$ denotes the predicted value of the current frame, x_k is the state vector, $\hat{x}_k$ is the optimal estimation of the a posteriori state, $\hat{x}_{\overline{k}}$ is the predicted value of the Kalman filtering algorithm, A_k is the state transfer matrix, B_k is the control variable matrix, u_k is the state control vector, Q denotes the system noise covariance matrix, H_k and denotes the state vector-to-measurement vector conversion matrix.

3.2. Graph Neural Networks

A graph convolutional neural network (GCN) is a specialized type of convolutional neural network designed for graph-structured data [28]. Similar to convolutional neural networks (CNNs), GCNs serve as feature extractors, but they are specifically tailored for processing graph data. GCNs employ a clever mechanism to extract meaningful features from the graph structure, which makes them suitable for learning representations of player context features and measuring player similarity.

The core operation of graph convolution in GCN is fundamentally similar to the convolution operation in CNN, as it involves aggregating information from neighboring nodes using a convolution kernel. However, there is a key difference: the convolution kernel in CNN has a fixed length, whereas the convolution kernel size in GCN is variable. This flexibility enables GCN to adapt to varying numbers of adjacent nodes within graph-structured data, addressing the issue of inconsistent neighborhood sizes. A graph is defined as follows: $G = (V, E, A)$, where V is the set of nodes, E is the set of edges, A is the adjacency matrix, $v_i \in V$ describes a point and $e_{ij} = (v_i, v_j) \in E$ describes an edge between two nodes [28]. The Laplacian matrix is defined as $L = D - A$ where D is the degree matrix and A is the adjacency matrix, which represents the adjacency between any two nodes, 1 for adjacency and 0 for non-adjacency. The node i in the $l+1$ layer of the graph neural network is updated as follows:

$$h_i^{(l+1)} = \sigma\left(h_i^l w_0^l + \sum_{j \in N(i)} c_{ij} h_j^l w_j^l \right) \tag{6}$$

where h_i^l represents the features of node i in layer l, $j \in N(i)$ is the neighbor node of node i, c_{ij} is the normalized coefficient of the neighbor node features, σ is the nonlinear activation function, w_j^l is the convolution kernel parameter of layer l. The $h_i^l w_0^l$ is the information of its own node, and the $\sum_{j \in N(i)} c_{ij} h_j^l w_j^l$ is the information of the neighbor node after normalization.

Graph neural networks can be further simplified as follows:

$$\mathbf{H}^{(l+1)} = \sigma(\mathbf{A}\mathbf{H}^l\mathbf{W}^l) \tag{7}$$

where $\mathbf{H}^l$ is the node features of layer l and $\mathbf{W}^l$ is the weight parameter matrix of layer l.

For the update of the edge, the node feature and the original edge weight are updated through the MLP process. It can be seen that the graph neural network can capture the global relationship of objects between neighboring frames and improve the precision of multi-object tracking.

3.3. Processing Methods for Contextual Features

The paper proposes a trajectory prediction approach based on velocity preprocessing, which aims to reduce the computational load of trajectory prediction and narrow the scope of the graph model in the object matching process. Specifically, this approach involves excluding unnecessary connections between objects, thereby streamlining the prediction process and improving the model's efficiency. We preprocess the objects based on observed

historical object information and classify trajectories according to different velocities to improve the accuracy of subsequent tracking.

By incorporating contextual information, we can more accurately represent the appearance features of objects.

Firstly, we construct a contextual graph $G = \{V, E\}$ for each object and its neighboring objects. Then, we use graph neural networks to integrate the contextual information of neighboring objects into a new feature for the designated object, referred to as the contextual feature. The integration of contextual information from surrounding objects can better represent the object's features.

The closer two objects are, the greater the weight of the edge connecting them. The formula for the edge weight is as follows:

$$\mathbf{X}^{(l+1)} \in \sigma(\widetilde{\mathbf{D}}^{-1}\widetilde{\mathbf{A}}\mathbf{X}^{(l)}\mathbf{W}^{(l)}) \tag{8}$$

where $\widetilde{\mathbf{D}}$ represents the distance matrix, and each element in the matrix represents the geometric distance between the object and its neighboring objects. In order to make the distance negatively correlated with the edge weights, the inverse of $\widetilde{\mathbf{D}}$ is taken to obtain $\widetilde{\mathbf{D}}^{-1}$.

Contextual features contain features of the object itself as well as neighboring objects and therefore have stronger feature representation capabilities. The similarity can be represented by the cosine distance of the object contextual features and learned using the Cosine Embedding Loss (CEL) function and the similarity labels. The formula and the schematic diagram in Figure 1 are as follows:

$$\text{loss}(x_1, x_2, y) = \begin{cases} 1 - \cos(x_1, x_2), & y = 1 \\ \max(0, \cos(x_1, x_2) - \text{margin}), & y = -1 \end{cases} \tag{9}$$

where x_1, x_2 are the feature vectors, y is the similarity label, 1 means similar, -1 means not similar, and margin is taken as 0–1.

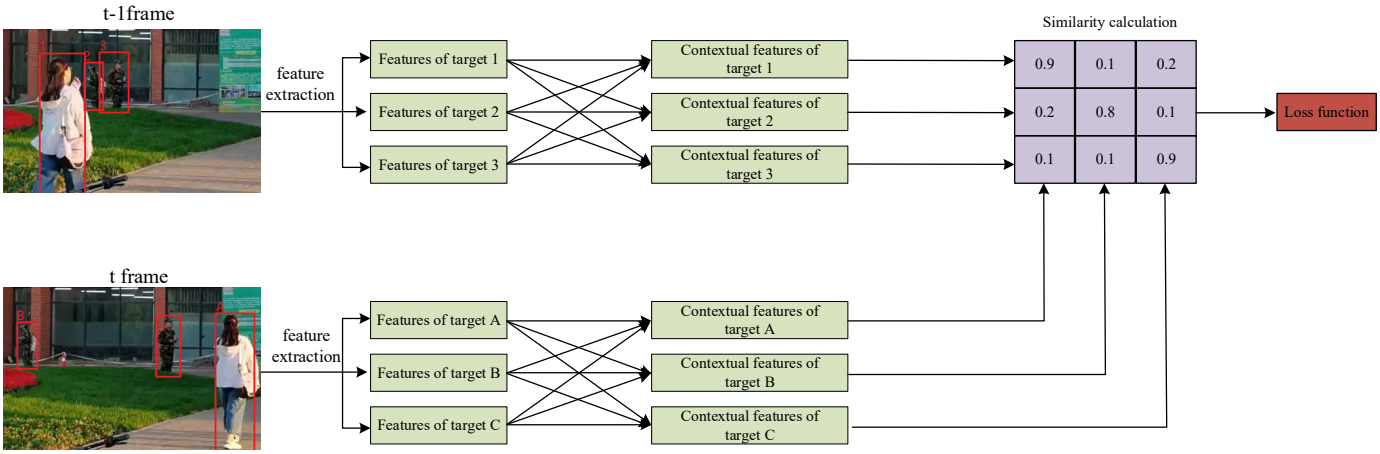


Figure 1. Contextual feature map.

In daily life, humans often rely on contextual information to distinguish objects. For instance, when an object is obstructed and cannot be effectively tracked, it is often possible to identify the object by considering the surrounding objects. In this paper, we use contextual graph modeling to better learn the appearance features of objects and then solve the problem of identity exchange of similar-looking objects in the tracking process.

4. Object Linkages

4.1. Hungarian Algorithm

The Hungarian algorithm is a combinatorial optimization method for solving task assignment problems in polynomial time. The key of this algorithm is to find an augmentation path to obtain the maximum matching of the bipartite graph. The algorithm flow is as follows:

(1) First, the bipartite graph is constructed, the vertices are numbered, and the relationship between the vertices of the two sets is represented by the adjacency list.
(2) Take the unmatched vertices in set M, i, and then traverse the vertices connected to it in set N. If the vertices in set N do not match, then match the two points. Then, continue to traverse the unmatched vertices in set M using the same principle. If the vertex in set N is already matched, then try to recursively match the matching point of the matched vertex in set N with another vertex match.
(3) Perform step (2) recursively until all vertices in M have been traversed to obtain the maximum match [29].

In frame $t-1$ we obtain $\widetilde{c}$ real frames based on the existing trajectories, and in frame t we obtain $\widetilde{a}$ detection frames [30], and we construct an IOU matrix of size $\widetilde{c} \times \widetilde{a}$, which is solved as shown in Equation (10):

$$\mathrm{IOU} = (\mathrm{Area}(\partial_1) \cap \mathrm{Area}(\partial_2)) / (\mathrm{Area}(\partial_1) \cup \mathrm{Area}(\partial_2)) \tag{10}$$

where ∂_1, ∂_2 are the object box and Area is the area of the object box.

4.2. Trajectory Post-Processing

To address the association matching problem more effectively, the paper proposes obtaining accurate motion features or appearance features of the objects first. Based on these features, the similarity between the objects to be matched is calculated. However, it is worth mentioning that in certain scenarios, such as occlusion or significant changes in motion state, the obtained information may not always be entirely accurate.

So, we first record the number of unmatched frames N for each trajectory i. If a trajectory does not have an object to match with it for t frames, the value of N is recorded as t. When a trajectory finds an object to match it in the current frame, N is set to 0. The intersection and concurrency ratio, individual features, contextual features, and Mars distance can be used as the main basis for calculating the similarity matrix for object matching as follows:

1. When N = 0, the similarity matrix is first calculated based on the contextual features. Then, the similarity matrix obtained from the Mahalanobis distance, as well as the intersection and merger ratio, is used as constraints to exclude matching errors due to occlusion, etc. This stage completes the majority of the matches.
2. When N > 0, the contextual features of neighboring objects become more distinguishable, and we obtain the similarity matrix for matching based on these contextual features.
3. When N $\geq$ 3, we need to match the object in two frames that are far apart. Due to the object's movement, the contextual features of the object tend to change significantly. Therefore, in such cases, we match the object based on its individual features.

Once we have obtained the similarity matrices for each stage, we subtract 1 from the corresponding matrices to obtain the respective cost matrices. At this point, the goal-association problem can be converted into a binary assignment problem and further solved using the Hungarian algorithm.

The general flow of this paper is given in Figure 2.

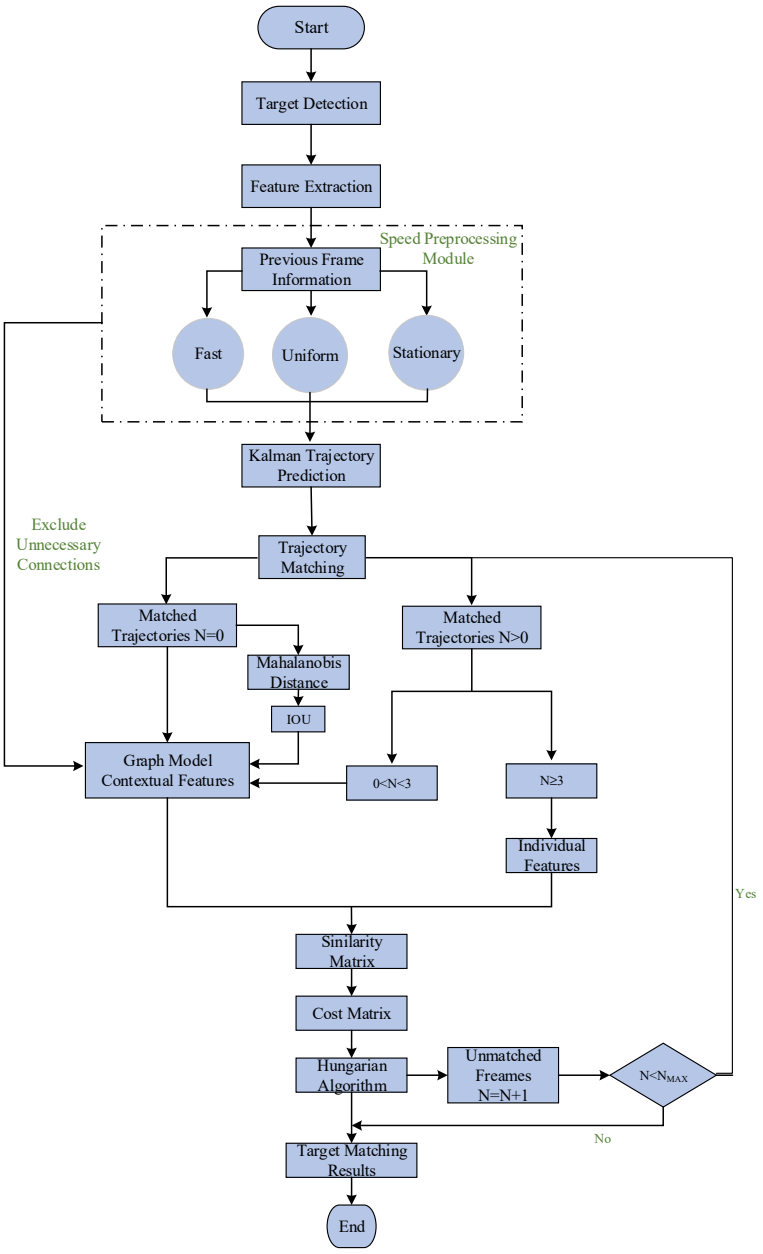


Figure 2. Overall flow chart.

5. Analysis of Experimental Results

5.1. Assessment Criteria

This experiment was conducted on the Windows 11 operating system. To enhance the analysis of the performance of the approach proposed in this paper, we selected the Transcenter, Transtrack, Fairmot, SUSHI, and Bytetrack algorithms for comparison.

$(\mathrm{MOTA}\uparrow)$: Multi-object tracking accuracy, which combines the number of false positives $(\mathrm{FP}\downarrow)$, the number of false negatives $(\mathrm{FN}\downarrow)$, and the number of identity switches $(\mathrm{IDS}\downarrow)$. The calculation formula is as follows:

$$\mathrm{MOTA} = 1 - (\mathrm{FN} + \mathrm{FP} + \mathrm{IDS})/\mathrm{GT} \tag{11}$$

where GT represents the number of real objects.

$\mathrm{MT}(\uparrow)$: the proportion of tracked trajectories to the total number of object trajectories (greater than eighty percent means that most objects are tracked accurately).

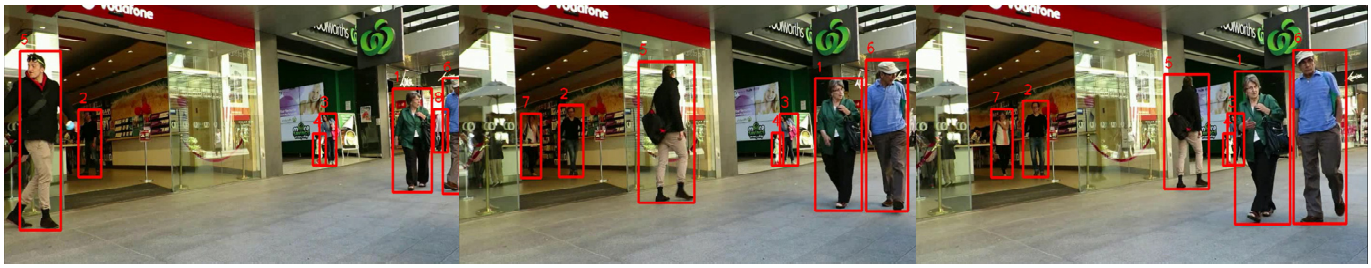
ML(↓): the proportion of untracked tracks to the total object tracks (less than twenty percent means that most object tracking tracks are missing).

IDS(↓): the number of times the object ID information was switched.

IDF1(↑): analyzes the global tracking result of an object in the whole video, which is no longer limited to a single frame. It is a reflection of the accuracy of ID recognition during tracking [31].

5.2. Simulation Analysis

Figure 3 shows the tracking results intercepted by the approach in this paper on MOT17-09; object 4 is a small child, but it is not missed because the object is too small, and it still does not switch IDs after it has been occluded by object 3 several times.



(a) (b) (c)

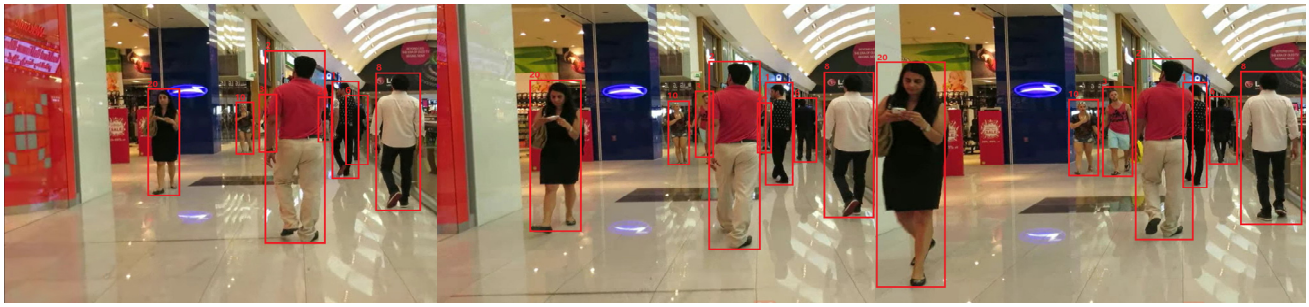
Figure 3. Tracking results on MOT17-09 (from (**a**–**c**): frame 24, frame 75, frame 101, respectively).

Table 1 compares the tracking results of the six approaches—Transcenter, Transtrack, Fairmot, SUSHI, and Bytetrack—in the above scenarios. From the data in the table, it can be seen that the multi-object tracking accuracy of the approach in this paper is almost equal to that of the SUSHI (Unifying Short and Long-Term Tracking) algorithm, which is better than the other algorithms. The IDF1 of this approach can reach 84, while the IDF1 of the rest of the algorithms is 62.4 for Transcenter, 63.5 for Transtrack, 72.3 for Fairmot, 83 for SUSHI, and 77.2 for Bytetrack. The number of times the object ID information is switched in this paper is 2303, which is less effective than SUSHI, but still better than the rest of the algorithms. However, compared with the rest of the algorithms, it still has a big advantage. The speed of the SUSHI algorithm is difficult to meet the real-time demand, but the FPS of this paper's approach can be up to 55, although it is slightly lower than Transtrack's 59.2, and the rest of the evaluation indexes are far more than that of Transtrack. To sum up, this paper's approach can realize better tracking of multiple objects.

Table 1. Comparison of metrics for different algorithms.

Algorithm	MOTA (↑)	IDF1 (↑)	IDS (↓)	FPS (Frame/s)
Transcenter	73.2	62.4	4614	1
Transtrack	75.2	63.5	3603	59.2
Fairmot	73.7	72.3	3303	25.9
SUSHI	81	83	1149	21
Bytetrack	78.9	77.2	2359	29.8
Proposed	80	84	2303	55

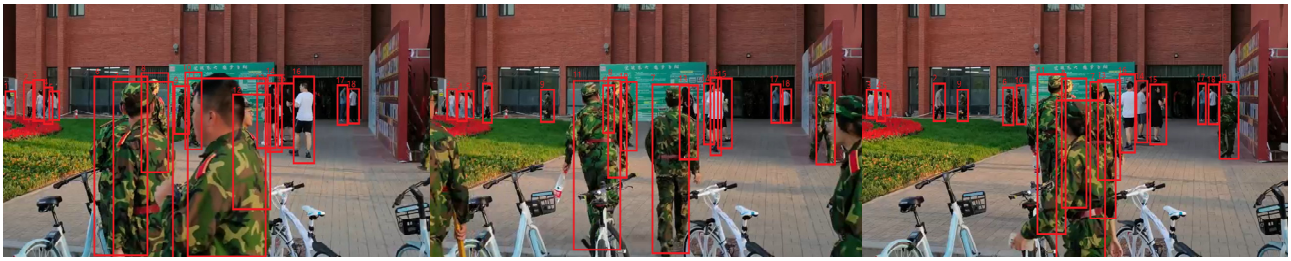
As can be seen from the tracking results of MOT17-11 in Figure 4, object 3 can still be tracked well when a large area is occluded, and object 24 as well as object 6 still have the correct ID after being occluded.



(a) (b) (c)

Figure 4. Tracking results on MOT17-11 (from (**a**–**c**): frames 113, 146, and 170, respectively).

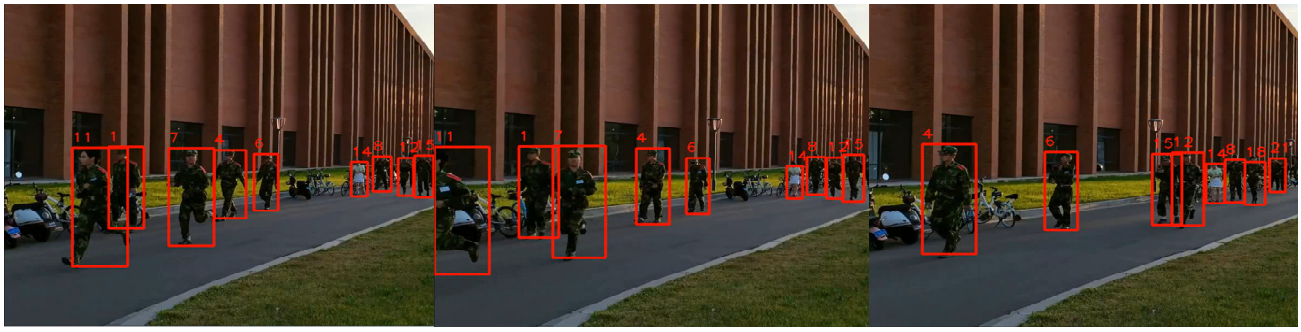
Figure 5 shows the video of an object of similar appearance taken with a cell phone, in which most of the people are wearing uniform clothes and the backs are obviously mostly occluded. The approach in this paper obtains a better representation of the appearance of similar objects because of the use of contextual features, which makes the tracking results better. In the figure, objects 17 and 18 are standing in place, talking, and almost stationary. This paper takes into account the appearance information and motion information and uses the preprocessing module to reduce the computation of the graph model.



(a) (b) (c)

Figure 5. Tracking results for look-alike objects (from (**a**–**c**): frames 30, 60, and 90, respectively).

Figure 6 shows the tracking results of similar-looking objects under backlit conditions. Because of the light, the object's facial features are blurred, and the same dress also brings more challenges for tracking. At this time, it is quite important to post-process the trajectories that are not matched in the current frame for the time being. Some objects are running faster and some are walking slower; the motion information and the use of preprocessing also play a crucial role in the tracking results.



(a) (b) (c)

Figure 6. Tracking results of objects with similar appearance under backlight conditions (from (**a**–**c**): frame 208, frame 219, frame 300, respectively).

Table 2 compares the tracking results before and after the improvement of the algorithm in different scenarios. Scene 1 is shown in Figure 5, and Scene 2 is shown in

Figure 6. From the data in the table, it can be seen that the approach in this paper combines the motion features with the appearance features, and utilizes both pre-processing and post-processing, which can achieve better multi-object tracking.

Table 2. Comparison of the tracking results before and after improvement in different scenarios.

Similarity Type	Scene 1		Scene 2	
	MOTA (↑)	IDS (↓)	MOTA (↑)	IDS (↓)
Similarity of motion features	69.7	96	72.6	214
Similarity of appearance features	72.4	95	74.7	159
Motion + Appearance features similarity	84.2	90	80.5	103
Motion + Appearance + Preprocessing similarity	88.9	72	82.6	99
Motion + Appearance + PostProcessing Similarity	94.3	68	86	92
Proposed Similarity	96.1	6	92	20
Real similarity	99.3	1	92.4	4

6. Summary

This paper integrates contextual features and trajectory prediction using Kalman filtering to predict the motion state of an object in the frame preceding the current frame. Initially, a graph model is employed to extract the contextual features of each object. Subsequently, the Hungarian algorithm is applied to match the object in the current frame with the object in the previous frame, and the unmatched frames undergo further processing. The method presented in this paper exhibits superior capabilities for extracting and distinguishing contextual features. The integration with trajectory prediction demonstrates a significant improvement in overall tracking performance compared to earlier multi-object tracking algorithms.

This paper improves the tracking effect to a certain extent through geometric constraint information and context features and solves the redundancy detection problem in the detection-based tracking model. Although the algorithm improves the tracking speed in crowded scenes and achieves excellent results on the MOT data set, there is still a lot of room for improvement in extremely crowded scenes. Due to the severe occlusion of human bodies in highly congested environments, conventional methods face challenges in detecting the entire human body. The model can be enhanced through head detection and other methods. Simultaneously, processing speed is crucial. In future practical applications, it is imperative to further improve the processing speed of tracking while ensuring high accuracy.

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

Funding: This work was supported by the 'Jie Bang Gua Shuai' Science and Technology Major Project of Liaoning Province in 2022 (No. 2022JH1/10400025) and the Fundamental Research Funds for the Central Universities of China (No. N2216010).

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors wish to express their gratitude to every reviewer of this paper.

Conflicts of Interest: The authors declare no conflict of interest. The company had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Blackman, S.S. Multiple hypothesis tracking for multiple object tracking. *IEEE Aerosp. Electron. Syst. Mag.* **2004**, *19*, 5–18. [CrossRef]

2. Dendorfer, P.; Osep, A.; Milan, A.; Schindler, K.; Cremers, D.; Reid, I.; Leal-Taixé, L. Motchallenge: A benchmak for single-camera multiple object tracking. *Int. J. Comput. Vis.* **2021**, *129*, 845–881. [CrossRef]
3. Zheng, L.; Tang, M.; Chen, Y.; Zhu, G.; Wang, J.; Lu, H. Improving multiple object tracking with single object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Beijing, China, 19–25 June 2021; pp. 2453–2462.
4. Wang, Q.; Zheng, Y.; Pan, P.; Xu, Y. Multiple object tracking with correlation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Beijing, China, 19–25 June 2021; pp. 3876–3886.
5. Liu, Z.; Shang, Y.; Li, T.; Chen, G.; Wang, Y.; Hu, Q.; Zhu, P. Robust Multi-Drone Multi-object Tracking to Resolve object Occlusion: A Benchmark. *IEEE Trans. Multimed.* **2023**, *25*, 1462–1476. [CrossRef]
6. Yang, D. Research on multi-object tracking technology based on machine vision. *Appl. Nanosci.* **2023**, *13*, 2945–2955. [CrossRef]
7. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
8. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef]
9. Sun, P.; Cao, J.; Jiang, Y. Transtrack: Multiple object tracking with transformer. *arXiv* **2020**, arXiv:2012.15460.
10. 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]
11. Wang, Z.; Zheng, L.; Liu, Y.; Li, Y.; Wang, S. Towards real-time multi-object tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 3069–3087.
12. Aharon, N.; Orfaig, R.; Bobrovsky, B.Z. BoT-SORT: Robust associations multi-pedestrian tracking. *arXiv* **2022**, arXiv:2206.14651.
13. Azhar, M.I.H.; Zaman, F.H.K.; Tahir, N.M.; Hashim, H. People tracking system using DeepSORT. In Proceedings of the 2020 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 21–22 August 2020; pp. 137–141.
14. Wang, Y.H. SMILEtrack: SiMIlarity LEarning for Multiple Object Tracking. *arXiv* **2022**, arXiv:2211.08824.
15. Voigtlaender, P.; Krause, M.; Osep, A.; Luiten, J.; Sekar, B.B.G.; Geiger, A.; Leibe, B. Mots: Multi-object tracking and segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 7942–7951.
16. Dendorfer, P.; Rezatofighi, H.; Milan, A.; Shi, J.; Cremers, D.; Reid, I.; Leal-Taixé, L. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv* **2020**, arXiv:2003.09003.
17. Xu, T.; Feng, Z.H.; Wu, X.J.; Kittler, J. Learning Adaptive Discriminative Correlation Filters via Temporal Consistency Preserving Spatial Feature Selection for Robust Visual Object Tracking. *IEEE Trans. Image Process.* **2019**, *28*, 5596–5609. [CrossRef] [PubMed]
18. Wang, Y.; Kitani, K.; Weng, X. Joint object detection and multi-object tracking with graph neural networks. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 13708–13715.
19. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual Object Tracking Using Adaptiv--**e Correlation Filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.
20. Liu, F.; Gong, C.; Huang, X.; Zhou, T.; Yang, J.; Tao, D. Robust Visual Tracking Revisited: From Correlation Filter to Template Matching. *IEEE Trans. Image Process.* **2018**, *27*, 2777–2790. [CrossRef] [PubMed]
21. Wen, L.; Du, D.; Cai, Z.; Lei, Z.; Chang, M.; Qi, H.; Lyu, S. UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Comput. Vis. Image Underst.* **2020**, *193*, 102907. [CrossRef]
22. Zeng, F.; Dong, B.; Wang, T.; Chen, C.; Zhang, X.; Wei, Y. Motr: End-to-end multiple-object tracking with transformer. *arXiv* **2021**, arXiv:2105.03247.
23. Zhang, W.; Zhou, H.; Sun, S.; Wang, Z.; Shi, J.; Loy, C.C. Robust multi-modality multi-object tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 2365–2374.
24. Luiten, J.; Osep, A.; Dendorfer, P.; Torr, P.; Geiger, A.; Leal-Taixé, L.; Leibe, B. Hota: A higher order metric for evaluating multi-object tracking. *Int. J. Comput. Vis.* **2021**, *129*, 548–578. [CrossRef]
25. Sulikowski, P.; Zdziebko, T. Deep Learning-Enhanced Framework for Performance Evaluation of a Recommending Interface with Varied Recommendation Position and Intensity Based on Eye-Tracking Equipment Data Processing. *Electronics* **2020**, *9*, 266. [CrossRef]
26. Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Weng, F.; Yuan, Z.; Wang, X. Bytetrack: Multi-object tracking by associating every detection box. In Proceedings of the European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022; pp. 1–21.
27. Jiang, X.; Ma, T.; Jin, J.; Jiang, Y. Sensor Management with Dynamic Clustering for Bearings-Only Multi-object Tracking via Swarm Intelligence Optimization. *Electronics* **2023**, *12*, 3397. [CrossRef]
28. Cheng, Y.; Zhang, S.; Wang, X.; Wang, H. Self-Tuning Process Noise in Variational Bayesian Adaptive Kalman Filter for object Tracking. *Electronics* **2023**, *12*, 3887. [CrossRef]
29. Zheng, X.; Liu, Y.; Pan, S.; Zhang, M.; Jin, D.; Yu, P.S. Graph neural networks for graphs with heterophily: A survey. *arXiv* **2022**, arXiv:2202.07082.

30. Chu, P.; Fan, H.; Tan, C.C.; Ling, H. Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 7–11 January 2019; pp. 161–170.
31. Chen, Z.; Xiong, X.; Meng, F.; Xiao, X.; Liu, J. Scaling-Invariant Max-Filtering Enhancement Transformers for Efficient Visual Tracking. *Electronics* **2023**, *12*, 3905. [CrossRef]