Roadside multi-vehicle tracking under different occlusion based on improved Siamese network and trajectory correlation

- Type the Subtitle of the Paper -

Shuai Wang [1)] **Yafei Wang** [2)]

1)Shanghai Jiaotong University, School of Mechanical Engineering, Shanghai, China
E-mail: wangshuai2020@sjtu.edu.cn
2) Shanghai Jiaotong University, School of Mechanical Engineering, Shanghai, China
E-mail: wyfjlu@sjtu.edu.cn

ABSTRACT: Roadside perception requires continuous tracking of vehicles in a fixed perception area. Traffic at city intersections is dense and easy to jam. Traffic lights and traffic jams also significantly lengthen occlusion times, making it harder for roadside sensors to track. In this paper, vehicle tracking model considering dynamic occlusion state transformation and trajectory correlation model based on road occlusion condition are constructed to achieve accurate vehicle tracking in different occlusion state. Firstly, a vehicle dynamic occlusion model was established based on the improved Siam network, and the vehicle dynamic occlusion under the local occlusion condition was analyzed. The tracking accuracy was improved by predicting the vehicle occlusion state. Secondly, a dynamic road occlusion model was established to analyze the possible positions of the occluded vehicles, and the correlation between the new point cloud and the interrupted trajectory in the sensing area was realized based on the point cloud similarity. Finally, the algorithm is verified in the data set DAIR-V2X-I, and it is proved that the algorithm has accurate and continuous tracking effect under the conditions of no occlusion, short-term occlusion and long-term occlusion.
KEY WORDS: Multi-vehicle tracking, Siamese network, Vehicle occlusion

1. INTRODUCTION

In recent years, many countries have accelerated the construction of intelligent transportation system(ITS). Accurate positioning and tracking of vehicles is the basis for further planning and control of ITS. However, with the popularity of intelligent roadside devices, the application scenarios of intelligent roadside devices have brought new challenges. Traditional perception methods applied to vehicle sensors and scenes cannot be well applied to roadside scenarios.[(1-5)]

The range of roadside perception is fixed, which brings advantages to perception, but also brings new problems. For the sensor installed in the vehicle, its sensing range is dynamic. When the vehicle enters its sensing range, the algorithm needs to detect and track the vehicle in real time. When the vehicle is blocked or the distance is too far for accurate detection, the vehicle ID can be lost.[(6-8)] In other words, for vehicle sensor, it only needs to continuously track the vehicles around self-vehicle that can be accurately captured.[(9)] When the vehicle target is lost due to occlusion and other reasons, that is, the sensing range of the vehicle sensor is temporarily reduced, and the algorithm still only needs to accurately detect and track the target within the dynamic sensing range.[(10)] When the lost target enters the perception range again, the perception algorithm can give it a new tracking ID, and its historical track outside the dynamic perception range will not interfere with the perception process of the vehicle sensor.[(11-15)]

This problem is particularly prominent in urban intersections, which are the first choice for the layout of intelligent roadside infrastructure.[(16-18)] The traffic flow within the intersection is dense, and it is easy to see vehicle congestion and long-term occlusion. Therefore, for the roadside sensors located in the range of urban intersections, developing a new perception algorithm to achieve continuous tracking of multi-target vehicles after a long-term occlusion has become a new problem that needs to be resolved urgently.[(19-22)]

In this paper, a vehicle dynamic occlusion model is proposed to analyze the changing trend of vehicle occlusion state, so as to track the local occlusion vehicle more conveniently. Furthermore, a road area occupancy model considering occlusion relationship is established to solve the vehicle track interruption problem caused by complete occlusion.

The main content of this paper is as follows. The second section introduces the vehicle dynamic occlusion state analysis model based on the improved Siamese network, which can predict the vehicle occlusion state and improve the tracking accuracy under local occlusion. The third section introduces the correlation model of road occlusion and track, analyzes the road dynamic occlusion area and the possible position of the occlusion vehicle, and finally realizes the correlation between the interrupt history trajectory and the new point cloud in the perception area through the similarity calculation. In fourth section, DAIR-V2X-I data set is used to verify the tracking effect of the algorithm under different occlusion conditions. The fifth section is the summary of the thesis.

2. VEHICLE TRACKING MODEL CONSIDERING DYNAMIC OCCLUSION STATE TRANSFORMATION

In this part, we will mainly introduce two points, one is the establishment of vehicle dynamic occlusion model; The second is the multi-vehicle tracking algorithm based on the classical siamese model.

2.1. Vehicle dynamic occlusion model

Firstly, we divided the vehicle occlusion state into five levels: 0%-20%, 21%-40%, 41%-60%, 61-80% and 81-100%. Among them, 0%-20% indicates that the completeness of vehicle point cloud is relatively high, and the vehicle does not seriously block the field of view of the sensor. However, 81-100% indicated that the completeness of vehicle point cloud was low, and the vehicle completely blocked the field of vision of the sensor and could not accurately perceive the surrounding environment.

We can judge the occlusion state of vehicles through the vehicle point cloud data collected by sensors. Specifically, we can divide vehicle point clouds into several small grids, and then calculate the degree of completeness of point clouds in each small grid, that is, the proportion of point clouds in the small grid to the total number of point clouds. If the integrity degree of point cloud in a small grid is lower than the threshold value, the grid is considered to have vehicle occlusion; otherwise, it is considered that there is no vehicle occlusion in the grid. Finally, we can calculate the proportion of vehicle occlusion in all grids to determine the occlusion state of vehicles.

The realization of this process mainly depends on the classification algorithm in supervised learning. Specifically, the occlusion state of vehicles can be divided into two categories: occluded and unoccluded, and then a classification model can be learned from historical data to predict the occlusion state of future vehicles.

Suppose there are n samples, and each sample contains m features and an occlusion status label y, the samples can be represented as:

$$\{(x1, y1), (x2, y2), \ldots, (x_n, \mathrm{y_n})\} \quad (1)$$

Where, xi is an m-dimensional vector, representing the characteristics of the ith sample, and yi is a binary label, representing the occlusioned state of the ith sample, which can be 0 or 1, respectively representing no occlusioned and occlusioned.

The classification model can be expressed as a function f that maps the input feature vector to a binary classification label:

$$y = f(x) \quad (2)$$

Where y is 0 or 1, indicating no or covered.

The training objective of the model is to minimize the prediction error rate, and the cross entropy loss function can be used:

$$L(w) =-1/n * \Sigma[yi * \log(f(x_i; w)) + (1 - yi) * \log(1 - f(x_i; w))] \quad (3)$$

Where, w is the parameter of the model, $f(x_i; w)$ is the predicted value of the model for the ith sample, and yi is the real label of this sample.

The stochastic gradient lower method can be used to solve the optimal parameter w, so as to obtain a classification model with high accuracy.

Specifically, the steps of stochastic gradient descent are as follows: initialize the model parameter w; Randomly select a sample x and corresponding label y from the training set, and then calculate the gradient of the sample to the model parameters:

$$\nabla \mathrm{L}(\mathrm{w}; \mathrm{x}, \mathrm{y}) = [\partial \mathrm{L}(\mathrm{w}; \mathrm{x}, \mathrm{y})/\partial \mathrm{w1}, \partial \mathrm{L}(\mathrm{w}; \mathrm{x}, \mathrm{y})/\partial \mathrm{w2}, \ldots, \partial \mathrm{L}(\mathrm{w}; \mathrm{x}, \mathrm{y})/\partial \mathrm{wm}] \quad (4)$$

Where, L(w;x,y) is the loss function, representing the difference between the predicted value of the model on sample x and the real label y. Then model parameters should be updated according to the gradient:

$$w = w - \alpha^* \nabla L(w; x, y) \quad (5)$$

Where, α is the learning rate, which controls the step size of each update.

2.2. Multi-vehicle tracking method based on Siamese network

Siamese network is a two-branch structure that is used to learn similarities between features. Its basic structure can be expressed as: input: point cloud cluster pair (I1, I2); output: similarity s. Two-point cloud clusters I1 and I2 are input into two convolutional neural network (CNN) networks with the same weight and structure, namely CNN1 and CNN2, to obtain two feature vectors f1 and f2.(4-6)

$$\begin{aligned} \mathrm{f1} &= \mathrm{CNN}\ 1(\mathrm{I1}) \\ \mathrm{f2} &= \mathrm{CNN}\ 2(\mathrm{I2}) \end{aligned} \tag{6}$$

In Siamese network, CNN is used to extract image features, and PointCNN is used to extract vehicle features in point cloud. Next, we use the LSTM network to process the vehicle's historical occlusion state information and predict the occlusion state at future times. The details of this module have been described in the previous section and will not be repeated in detail here. The input of this module includes a series of historical occlusion states h_t, such as occlusion ratio, occlusion type, etc. The output of this module is mainly the occlusion state p_t predicted by the vehicle in the future time.(8-9)

Next, the predicted occlusion state p_t is fused with feature vectors f1 and f2. Here, we take the attention mechanism as an example to achieve feature fusion. The input values are the feature vectors f1, f2 and the predicted occlusion state p_t. The outputs are fusion feature vectors g1 and g2. Then the weighted sum of the feature vectors is performed using attention weights:

$$g1 = f1 * \alpha1 + p\_t * (1 - \alpha1), g2 = f2 * \alpha2 + p\_t * (1 - \alpha2) \tag{7}$$

Among them, α1 and α2 are attention weights. Where, $\alpha1$ and $\alpha2$ are attention weights, which are calculated by Softmax function.

$$\text{Softmax}\ (a) = \mathrm{e}^{\mathrm{a}} / \Sigma \mathrm{e}^{(a\_i)} \tag{8}$$

Input the attention scores a1 and a2 into the Softmax function to calculate $\alpha1$ and $\alpha2$.

$$\begin{aligned} \alpha1 &= \mathrm{e}^{\mathrm{a1}} / (\mathrm{e}^{\mathrm{a1}} + \mathrm{e}^{\mathrm{a2}}) \\ \alpha2 &= \mathrm{e}^{\mathrm{a2}} / (\mathrm{e}^{\mathrm{a1}} + \mathrm{e}^{\mathrm{a2}}) \end{aligned} \tag{9}$$

Next, we need to calculate the similarity s between the fused feature vectors g1 and g2. We use cosine similarity to calculate the similarity between the two fused feature vectors.

$$\mathrm{s} = (\mathrm{g1} \cdot \mathrm{g2}) / (\parallel g1 \parallel \parallel g2 \parallel) \tag{10}$$

Among them ‖·‖ represents the magnitude of the vector.

In order to train the model, we need to define a loss function to measure the difference between the model's prediction and the real value. We use the comparative loss method to define the loss value:

$$L = 0.5^{*} y^{*}\ \mathrm{s}\text{^}2 + 0.5^{*}(1 - y)^{*}\max(0, m - s)\text{^}2 \tag{11}$$

Where s represents similarity, label y represents vehicle status (y=1 represents the same vehicle, y=0 represents different vehicles), and m represents distance boundary.

Then we use Adam (Adaptive Moment Estimation) optimizer to optimize the loss function through the gradient descent method to update the network parameters.

First, we need to define some required hyperparameters, learning_rate=0.001, beta_1=0.9, beta_2=0.999 and epsilon=1e-8. We then initialize the first and second moment variables, iterate over each batch of data and update the parameters.

Finally, we calculate the similarity of point cloud clusters of target vehicles on continuous frames to achieve vehicle tracking. The specific steps are as follows:

a) Read the current frame point cloud, extract vehicle position I_t, and record the historical occlusion state h_t;

b) LSTM is used to predict the occlusion state p_t at future time;

c) Calculate the similarity s between the candidate vehicle point cloud cluster I_c of the adjacent frame and the current vehicle point cloud cluster. If s > T (similarity threshold), it is considered that the target vehicle is found, and I_t is updated to I_c and h_t is updated to jump out of the loop;

d) If the target vehicle is not found, update h_t and consider increasing the threshold T.

Through the above steps, we can realize a vehicle tracking system based on Siamese network, which can use the predicted occlusion state of the vehicle in the future time to realize vehicle tracking.

3. VEHICLE TRACKING MODEL CONSIDERING DYNAMIC OCCLUSION STATE TRANSFORMATION

In this part, we will introduce the method of analyzing the possible location of the occluded vehicle by using the road occlusion state and realizing the correlation with the historical interrupt trajectory.

First we need to initialize the shaded area of the road, which can be represented as polygons or other geometric shapes. In this paper, we represent it as a polygon, represented by the vertex set V, where V = {v_1, v_2,... v_n}, v_i = (x_i, y_i) for each vertex.

Then we need to analyze the dynamic occlusion region caused by the vehicle projection relationship. Firstly, the sensing range R is set, indicating the maximum detection distance of the sensor. Set the sensor position on the side to P_s=(x_s, y_s). For each pair of tracking vehicles (i, j), the projection relationship between them is analyzed. Calculate the vector D_i = p_i-p_s from sensor to vehicle i, where P_i is the position of vehicle i. The vector D_j

= p_j-p_s from the sensor to vehicle j is calculated, where P_j is the position of vehicle j.[18-22]

Then we need to calculate the projection point P_proj and judge whether the projection point P_proj is located near vehicle j. If so, it is considered that vehicle i is blocked by vehicle j.

$$P\_proj = P\_i + (D\_i \cdot Dj/\| Dj\|^{\wedge}2) * {}^{*}j \tag{12}$$

Next we need to set up a dynamic update model for the occlusion area. First, the occlusion region is represented as A geometric shape (such as polygons A and B). On the two-dimensional plane, polygons A and B can be represented as a set of vertices as A={P1_A, P2_A,... , Pn_A} and B={P1_B, P2_B,... , Pm_B}. Then the minimum distance dist_min between the projection point P_proj and the boundary of the occlusion region is calculated. Using the formula for distance from point to boundary:

$$\text{dist\_min} = \min(\text{ distance(P\_proj, Edge\_i)}) \tag{13}$$

Determine whether the occlusion area needs to be updated based on dist_min and preset threshold T_dist. If dist_min is less than T_dist, the occlusion region needs to be merged or extended. When updating the occlusion region, the boundary of the occlusion region can be adjusted using geometric operations. We achieve this effect by calculating the union of occluded regions A and B:

$$\text{Union } (A, B) = A \cup B \tag{14}$$

Then we need to correlate the historical trajectory with the emerging point cloud cluster. Firstly, the distance matrix D between the new point cloud information C and the historical trajectory H is calculated in the occlusion region. Each element d_ij represents the distance between the new point cloud information c_i and the historical track h_j:

$$\text{D} = [\text{d\_ij}] \tag{15}$$

$$d\_ij = \|c\_i - hj\| \tag{16}$$

The distance matrix D is matched by the Hungarian algorithm. Through this algorithm, the best matching relation M between the new point cloud information and the historical trajectory can be found. Then, according to the preset association threshold T_assoc, the matching results are filtered. If d_ij is less than T_assoc, the matching result is accepted. Otherwise, the matching result is rejected.

Then we extract vehicle feature F, including shape information F_shape and dynamic information F_move. When associating new point cloud information with historical trajectory information, the feature similarity matrix S is calculated. Each element s_ij represents the characteristic similarity between the new point cloud information c_i and the historical trajectory h_j:

$$S = [s\_ij] \tag{17}$$

$$\begin{aligned} \text{s\_ij} = \text{w\_color} * \text{sim}(\text{F\_move}(\text{c\_i}), \text{F\_move}(\text{h\_j})) + \\ \text{w\_shape} * \text{sim}(\text{F\_shape}(\text{c\_i}), \text{F\_shape}(\text{h\_j})) \end{aligned} \tag{18}$$

Where, w_move and w_shape are the weights, and sim is the feature cosine similarity measure. Then, the feature similarity matrix S is combined with the distance matrix D to calculate the comprehensive correlation matrix R:

$$R = [r\_ij] \tag{19}$$

$$r\_ij = \lambda * d\_ij + (1 - \lambda) * (1 - s\_ij) \tag{20}$$

Where, λ is the weight between distance and feature similarity, and the value range is [0, 1].

Then we need to define a threshold that can be adjusted dynamically. Firstly adjust_threshold(T, x) is defined as a threshold adjustment function, where T is the original threshold and x is the influencing factors (such as vehicle spacing, speed, etc.). Then exponential function is used to represent the relationship between vehicle spacing d and threshold T_assoc:

$$\begin{aligned} \text{T\_assoc\_adj} = \text{adjust\_threshold}(\text{T\_assoc}, \text{d}) = \\ \text{T\_assoc} * \text{e}^{(-\text{k}*\text{d})} \end{aligned} \tag{21}$$

Where, k is a constant and represents the rate at which the threshold changes with vehicle spacing. When associating new point cloud information with historical track information, the dynamically adjusted threshold T_assoc_adj is used for screening. If r_ij is less than T_assoc_adj, the matching result is accepted. Otherwise, the matching result is rejected.

In summary, by building a road occlusion model, the interrupt trajectory can be accurately correlated with the newly emerged point cloud cluster from the occlusion area, providing an effective solution for road side perception of common occlusion problems at the far end.

4. EXPERIMENTS

In this study, we used the public data set DAIR-V2X-I released by Tsinghua University to verify our multi-vehicle tracking algorithm. The DAIR-V2X-I dataset contains a large number of road information in real traffic scenarios, including vehicle location, speed, shape, color and other attributes. The data set has complex occlusion conditions and is suitable for verifying multi-vehicle tracking algorithms considering different degree occlusion problems.

In the experiment, we first preprocess the data set, extract the relevant features and calculate the occlusion area. Then, the extracted features and occlusion area information are input into the multi-vehicle tracking algorithm designed by us. In the algorithm, we consider the dynamic changes of the occlusion region, and correlate the historical track information of the interrupted perception region with the information of the new point cloud in the occlusion region according to the possibility. We also introduce dynamic threshold adjustment and matching strategy of vehicle feature similarity to improve the accuracy and robustness of the algorithm.

In order to evaluate the performance of our algorithm in dealing with different degrees of occlusion, we use the accuracy and precision to evaluate the algorithm in the experiment to show the advantages of our algorithm in dealing with occlusion problems. In the following part, we will present the quantitative results of the experiment in detail to prove the effectiveness of our multi-vehicle tracking algorithm in dealing with complex occlusion conditions. At the same time, we will discuss the limitations of the algorithm and possible improvements in the future. The tracking effect of the proposed method in the complex traffic environment at intersections is shown in the figures below.

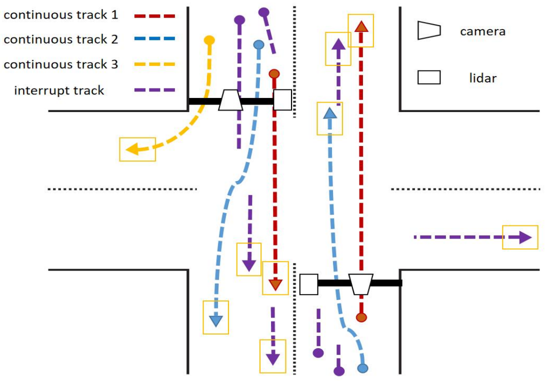

Fig. 1. The complex traffic scenario at urban intersection.

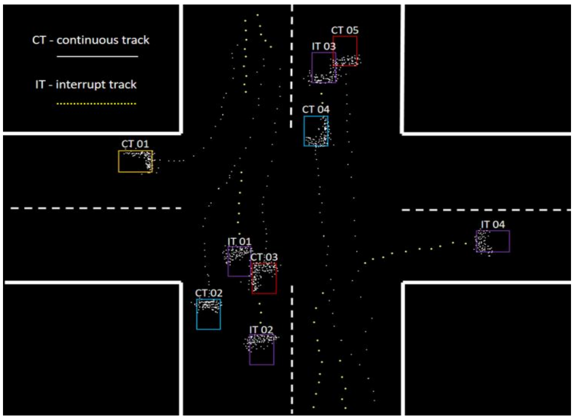

Fig. 2. The tracking effect of algorithm on complex traffic scenario.

In order to better demonstrate the effect of the proposed method in different tracking scenarios, occlusion cases are divided into five categories: no occlusion (0s); partial occlusion (0S); very short occlusion (1s); short-term occlusion (5s-10s); long-time occlusion (over 10s).

Table 1 Algorithm tracking effect under different occlusion cases.

Occlusion time	Evaluation indicators	
	MOTA	MOTP
0	94.86%	93.55%
0 (part)	87.15%	91.72%
≤ 5s (whole)	85.75%	84.92%
5s-10s (whole)	80.14%	82.95%
≥ 10s (whole)	82.45%	86.07%

It is not difficult to see from the above results that the scheme proposed in this paper shows good tracking effect under different occlusion conditions, no matter no occlusion scene, partial occlusion scene or different length of full occlusion scene. Among them, the occlusion form within the interval of 5s-10s is usually caused by vehicle creep. Since the traffic flow density is usually high in this scene, the occlusion tracking algorithm in this scene has higher requirements. It can be seen from the results that even in this scene, the method proposed in this paper still shows a great tracking effect.

5. CONCLUSION

In this study, we propose an improved multi-vehicle tracking algorithm based on Siamese network to optimize the complex occlusion problem. Firstly, we designed an improved Siamese network combined with vehicle occlusion state prediction to improve vehicle tracking accuracy by learning the state characteristics of vehicles in continuous moments. Then, according to the position of the roadside sensor, we analyze the possible occlusion area in the road, and consider the dynamic change of the occlusion area. When associating historical track information with new point cloud information, we introduced dynamic threshold adjustment and vehicle feature similarity matching to improve the accuracy and robustness of the algorithm. Experimental verification on DAIR-V2X-I, a public data set released by Tsinghua University, shows that our algorithm shows good tracking performance under different degrees of occlusion.

Although our multi-vehicle tracking algorithm shows advantages in dealing with complex occlusion problems, there are still several directions worth further research and improvement:

1) Introduction of more prior information: In addition to vehicle feature similarity and dynamic changes of occlusion area, more prior information, such as lane information and intersection layout, can be considered to improve the performance of the tracking algorithm.

2) Online learning and model updating: In real-world scenarios, road conditions and vehicle behavior may change. Therefore, online learning and model updating methods can be studied so that the algorithm can adapt to the changing environment.

3) Fusion of multi-source sensor data: Considering multi-source sensor data, such as lidar, millimeter-wave radar and camera, can further improve the robustness and accuracy of the tracking algorithm.

4) Optimization of computing efficiency: For real-time application scenarios, more efficient algorithm implementation methods and computing platforms can be studied to reduce computing delay and resource consumption.

REFERENCES

(1) Zhao, K. , et al. "3D Vehicle Detection Using Multi-Level Fusion From Point Clouds and Images." *IEEE Transactions on Intelligent Transportation Systems* PP.99(2022).

(2) Zhang, R. , et al. "A Multi-Vehicle Longitudinal Trajectory Collision Avoidance Strategy Using AEBS With Vehicle-Infrastructure Communication." *IEEE Transactions on Vehicular Technology* 2(2022):71.

(3) Qian, K. , P. Chen , and D. Zhao . "GOMT: Multispectral video tracking based on genetic optimization and multi-features integration." *IET Image Processing* (2023).

(4) Cao, Nan, et al. "3D Object Detection and Tracking in Monocular Images Using Deep Learning and Particle Filter." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(5) Li, Qichao, et al. "Multi-Sensor Fusion for Robust 3D Object Detection in Autonomous Driving." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(6) Wang, Zirui, et al. "PointNetGPD: Detecting Grasping Positions for Robotic Manipulation Based on Point Cloud Data." *IEEE Robotics and Automation Letters* 6.2 (2021): 2708-2715.

(7) Yang, Xinchi, and Hongsheng Yin. "Robust Multi-Object Tracking with Graph Convolutional Network." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(8) Chen, Hao, and Fuqiang Lv. "End-to-End Multi-Object Tracking with Transformer and Graph Neural Network." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(9) Li, Xiaodong, et al. "Robust Multi-Object Tracking by Learning Graph Attention Network." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(10) Wei, Yuxin, et al. "A Modular Approach for Multi-Object Tracking with Sensor Fusion in Autonomous Driving." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(11) Chen, Wenjian, et al. "3D Object Detection for Autonomous Vehicles with Dynamic Graph Neural Network." *IEEE Transactions on Vehicular Technology* 70.7 (2021): 6586-6596.

(12) Zhang, Jianfeng, et al. "A Deep Learning Approach for Robust Multi-Object Tracking in Traffic Scenes." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(13) Wang, Zhongdao, et al. "Towards Real-Time Multi-Object Tracking." *Proceedings of the European Conference on Computer Vision* (ECCV). 2020.

(14) Yu, Qiang, et al. "DeepSORT: Deep Learning-Based Multi-Object Tracking with Sensor Fusion." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(15) Luo, Wenjie, et al. "Graph-Transformer: A General Graph Neural Network for Joint Multiple Object Detection and Tracking." Proceedings of the *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

(16) Xu, Yunlong, et al. "Learning a Cost-Effective Annotation Pipeline for Video Object Detection." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

(17) Chen, Yan, et al. "Learning Feature Representation for Aerial Vehicle Tracking with Cross-View Domain Adaptation." *IEEE Transactions on Intelligent Transportation Systems* (2021).

(18) Shin, V. , V. Hamdipoor , and Y. Kim . "Reduced-order multisensory fusion estimation with application to object tracking." *IET signal processing* 4(2022):16.

(19) Yan, Z., et al. "Formation trajectory tracking control of discrete-time multi-AUV in a weak communication environment." *Ocean engineering* 245-Feb.1(2022).

(20) Xiang, Chao, et al. "FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object

Tracking." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020.

(21) Duan, Yunpeng, et al. "MOTR: End-to-End Multiple-Object Tracking with TRansformer." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

(22) Lin, Tsung-Yi, et al. "CenterNet: Keypoint Triplets for Object Detection." *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.