World Electric Vehicle Journal

MDPI

Article

Enhanced Object Detection in Autonomous Vehicles through LiDAR—Camera Sensor Fusion

Zhongmou Dai [1,2], Zhiwei Guan [1,3], Qiang Chen [1,*], Yi Xu [4,5] and Fengyi Sun [1]

[1] School of Automobile and Transportation, Tianjin University of Technology and Education, Tianjin 300222, China; 0311201001@tute.edu.cn (Z.D.); guanzhiwei@tsguas.edu.cn (Z.G.); 0721211001@tute.edu.cn (F.S.)
[2] Shandong Transport Vocational College, Weifang 261206, China
[3] School of Automobile and Rail Transportation, Tianjin Sino-German University of Applied Sciences, Tianjin 300350, China
[4] National & Local Joint Engineering Research Center for Intelligent Vehicle Road Collaboration and Safety Technology, Tianjin 300222, China; xuyi@sdut.edu.cn
[5] QINGTE GROUP Co., Ltd., Qingdao 266106, China
* Correspondence: chen@tute.edu.cn

Abstract: To realize accurate environment perception, which is the technological key to enabling autonomous vehicles to interact with their external environments, it is primarily necessary to solve the issues of object detection and tracking in the vehicle-movement process. Multi-sensor fusion has become an essential process in efforts to overcome the shortcomings of individual sensor types and improve the efficiency and reliability of autonomous vehicles. This paper puts forward moving object detection and tracking methods based on LiDAR—camera fusion. Operating based on the calibration of the camera and LiDAR technology, this paper uses YOLO and PointPillars network models to perform object detection based on image and point cloud data. Then, a target box intersection-over-union (IoU) matching strategy, based on center-point distance probability and the improved Dempster–Shafer (D–S) theory, is used to perform class confidence fusion to obtain the final fusion detection result. In the process of moving object tracking, the DeepSORT algorithm is improved to address the issue of identity switching resulting from dynamic objects re-emerging after occlusion. An unscented Kalman filter is utilized to accurately predict the motion state of nonlinear objects, and object motion information is added to the IoU matching module to improve the matching accuracy in the data association process. Through self-collected data verification, the performances of fusion detection and tracking are judged to be significantly better than those of a single sensor. The evaluation indexes of the improved DeepSORT algorithm are 66% for MOTA and 79% for MOTP, which are, respectively, 10% and 5% higher than those of the original DeepSORT algorithm. The improved DeepSORT algorithm effectively solves the problem of tracking instability caused by the occlusion of moving objects.

Keywords: autonomous vehicles; object detection; object tracking; LiDAR—camera fusion; improved DeepSORT

Citation: Dai, Z.; Guan, Z.; Chen, Q.; Xu, Y.; Sun, F. Enhanced Object Detection in Autonomous Vehicles through LiDAR—Camera Sensor Fusion. *World Electr. Veh. J.* **2024**, *15*, 297. https://doi.org/10.3390/wevj15070297

Academic Editor: Joeri Van Mierlo

Received: 24 March 2024
Revised: 25 June 2024
Accepted: 1 July 2024
Published: 3 July 2024

1. Introduction

Many accidents are caused by drivers' failure to pay attention to moving objects at critical moments or by adverse conditions that impede visibility. Today, autonomous vehicles (AVs) [1,2] can assist or even complete driving operations independently, which is of great significance in efforts to liberate the human body and greatly reduce the accident rate. The key technology of autonomous vehicles, allowing them to interact with their external environments, is the environment perception module [3]. Determining the accuracy of perceived information has become one of the recent research hotspots in the field of

autonomous vehicles [4,5]. To realize accurate perception, it is first necessary to solve the issue of object detection and tracking in the vehicle-movement process.

Invariably, the overall performance of an AV is greatly enhanced by the operation of multiple sensors of different types and modalities at varying ranges and bandwidths, with the data from each being incorporated to produce a fused output [6,7]. Multi-sensor fusion has become an essential process in all AVs, allowing them to overcome the shortcomings of individual sensor types and improving the efficiency and reliability of AVs overall. Among the available multi-sensor fusion techniques, 3D light detection and ranging (LiDAR) can acquire high-precision depth information, such as distance and angle, from the surroundings, boasting a large detection area and exhibiting strong anti-interference ability [8]. However, this technology also suffers from low data resolution, a lack of textural information, and a high data collection cost [9]. Compared to LiDAR, cameras can provide visual information, such as high-resolution color and texture, about the extracted features. Additionally, cameras are inexpensive [10]. However, camera technology cannot provide reliable and accurate results for objects placed at farther distances [11]. As such, the integration of data from LiDAR devices and cameras for use in object detection and tracking [8,12] has become a hot research topic in recent years.

However, LiDAR devices and cameras are located at different spatial positions within multi-sensor systems. A common solution to this is to perform relative coordinate transformation between LiDAR devices and cameras via extrinsic calibration [13]. Existing calibration methods for LiDAR devices and cameras can be broadly classified into three categories: convolution neural network (CNN)-based methods, targetless methods, and target-based methods. CNN-based methods, which aim to extract adaptive co-observed features to regress extrinsic parameters, are currently giving rise to accurate calibration between 3D LiDAR devices and cameras through real-time online data [14,15]. The targetless method leverages the motion estimated by individual sensors or utilizes features related to environmental perception to calibrate the sensors. Gong et al. [16] estimated calibration parameters between 3D LiDAR devices and cameras based on the point-to-plane geometric constraints of three-plane orthogonal trihedrons, which are ubiquitous in natural scenes. Using both 3D LiDAR devices and cameras, target-based methods rely on the simultaneous observation of artificial calibration targets placed in front of sensor systems. Then, extrinsic calibration is calculated by solving the calibration matrix conversion equation [17] or using a method such as supervised learning [18] etc.

At present, there are three primary approaches used to combine data from LiDAR devices and cameras, namely high-level fusion (HLF), low-level fusion (LLF), and mid-level fusion (MLF) [19]. In the HLF approach, each sensor conducts object detection or uses a tracking algorithm independently and subsequently performs fusion [20]. In the LLF approach, data are fused at the lowest level of raw data. Therefore, all information is retained and can potentially be used to improve obstacle detection accuracy. Wu et al. [21] projected 3D LiDAR data into the image space and used a region proposal network (RPN) to generate convolutional features. The features of 3D LiDAR technology were fused with regional features obtained from the camera images, and this information was input into the faster R-CNN network for object detection. Arikumar et al. [22] proposed an object detection mechanism (OD-C3DL) that fuses the data received from the camera and the 3D LiDAR. These data are fed into the PCA process, which then extracts and removes the floor points. Then, OD-C3DL creates the contours of the object region, uses CNN to perform feature extraction, and employs object classification to conduct the desired accurate object identification. Chen et al. [23] proposed MV3D, utilizing LiDAR bird's-eye-view (BEV) features to create 3D object proposals and project them onto multi-view images for RoI feature extraction. HLF approaches are often adopted due to their lower relative complexity than the LLF and MLF approaches. However, HLF provides inadequate information as classifications with a lower confidence value are discarded if there are several overlapping obstacles.

Kim et al. [24] proposed an advanced weighted-mean You Only Look Once (YOLO) algorithm, hoping to fuse RGB camera and LiDAR point cloud data to improve the real-time performance of object detection techniques. Wang et al. [25] calibrated the LiDAR device and camera to convert the point cloud into a depth map and thicken it. The lightweight network Mobilenet v2 and the high-precision network YOLOv3 were combined to detect RGB images and dense depth maps. Finally, the detection results of point cloud depth maps and RGB images were fused with dynamic weights via a decision-level fusion model. PointFusion [26] was presented to leverage the image data and raw point cloud data independently for use in 3D object detection. Dempster–Shafer (D–S) theory [27] was employed to obtain the final vehicle detection result after projecting the point cloud onto an RGB image via a dense depth map and using a YOLOv3 algorithm to detect vehicle targets separately. This decision-level fusion method has strong real-time and adaptive capabilities, as well as a strong anti-interference ability. If a single detection device fails, it will not affect the operation of other detection devices, and the setup can still provide a final decision.

Moving object tracking is used to achieve the cross-frame recognition of target objects, preventing the loss of target objects due to detection failures. Bewley et al. [28] proposed the classic SORT algorithm, combining Kalman filter update prediction and the Hungarian algorithm association matching function to achieve dynamic multi-objective online tracking. Although this algorithm exhibits robust accuracy and high speed, it faces the problem of frequent changes in identity (ID). Wojke et al. [29] introduced deep learning to improve the DeepSORT algorithm, which is based on SORT. This greatly reduced the amount of ID switching compared to that seen using the SORT algorithm. The phenomenon of dynamic target trajectory intersection and occlusion is inevitable in multi-target tracking. The intersection of dynamic multi-target motion trajectories can lead to target occlusion, and other objects may also occlude the target during movement. Therefore, occlusion problems can easily lead to losses in target tracking. Wang et al. [30] proposed a novel 3D MOT framework based on camera—LiDAR fusion. The embedded depth correlation mechanism in the framework tracks an object in a 2D domain when the object is far away and can only be detected by the camera and updates the 2D trajectory with 3D information obtained when the object appears in the LiDAR field of view, thus achieving a smooth fusion of 2D and 3D trajectories. Wang et al. [31] proposed a novel camera—LIDAR fusion 3D MOT framework based on combined appearance motion optimization (CAMO-MOT), in which the occlusion head was designed to identify the object occlusion state and select optimal appearance features to reduce the effect of occlusion. Chen et al. [32] introduced the radial basis forward neural network to overcome the performance deterioration of autonomous vehicle path-tracking controllers. Zhao et al. [33] introduced the solution algorithms and application guidance associated with using infrastructure-based LiDAR sensors to accurately detect and track pedestrians and vehicles at intersections. Hosseinzadeh et al. [34] introduced the notion of danger awareness of human–robot interaction and built a predictive human model to anticipate future actions.

This paper adopts a decision-level fusion approach to fuse the object detection results of LiDAR—camera, utilizing fusion detection results to perform dynamic target-tracking tasks. The principal contributions of this paper are as follows:

1. The design of a LiDAR—camera fusion strategy for object detection is presented. First, the 3D point cloud object detection box is projected onto the image via joint calibration results. Then, a target box IoU matching strategy based on center-point distance probability is adopted to match and fuse the 2D point cloud projection box with the camera detection target box. Subsequently, the D–S theory is utilized for class confidence fusion to obtain the final fusion detection result.
2. In response to the problem of ID transformation, which occurs when the target is occluded, the DeepSORT algorithm is improved via the addition of an unscented Kalman filter to accurately predict the nonlinear target motion state. The IoU matching

module incorporates target motion information to improve the matching accuracy in the data association process.

The remainder of this paper is organized as follows. Section 2 introduces the moving object detection and tracking methods based on LiDAR—camera fusion. In Section 3, the experimental preparation and data used are explained. In Section 4, the experimental processes are presented and analyzed, and the performances of the methods are compared. In Section 5, discussions are presented, and conclusions are drawn.

2. Methods

2.1. Overall Framework

Figure 1 shows the overall framework being used: moving object detection is combined with a tracking method based on LiDAR—camera fusion, with the latter technique primarily comprising a LiDAR—camera detection module, fusion strategy module, and tracking module. The point cloud data obtained via LiDAR scanning is used to obtain the 3D point cloud object detection results through the PointPillars network. The image data captured by the camera is used to obtain 2D object detection results through the YOLOv5 network. Based on the joint calibration of external parameters with the camera and LiDAR device, the 3D object detection box of the point cloud is projected onto the image to obtain the 2D point cloud projection box. Then, target box intersection-over-union matching is performed via a strategy of fusing the target box with the camera detection target box. Subsequently, the confidence fusion strategy is adopted from the D–S theory to complete the fusion of category confidence and output the fusion result. Finally, the fusion detection results are fed into the improved DeepSORT target-tracking algorithm to track dynamic targets for vehicles and pedestrians.

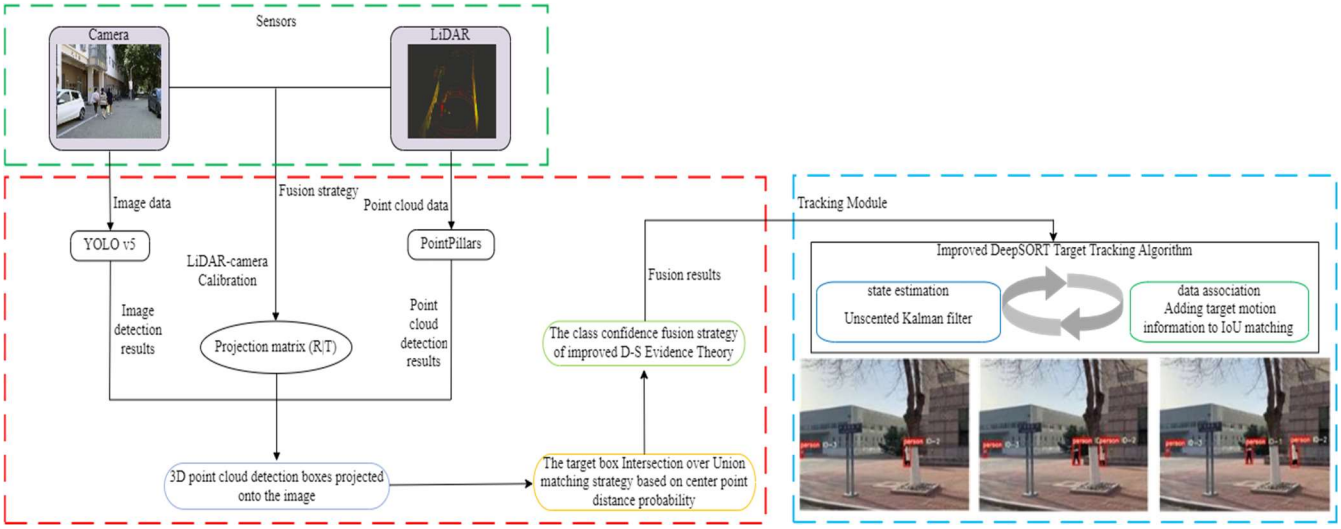


Figure 1. Overview of moving object detection and tracking methods based on LiDAR–camera fusion.

2.2. Fusion Detection

Before fusing camera and LiDAR data, camera and LiDAR data were synchronized over time, and spatial coordinate systems were unified, as shown in Figure 2.

The spatial conversion model for the joint calibration of the camera and LiDAR is shown in Formula (1).

$$c\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & u_o & 0 \\ 0 & f_y & v_o & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix} \begin{bmatrix} X_L \\ Y_L \\ Z_L \\ 1 \end{bmatrix} = K \times M \times \begin{bmatrix} X_L \\ Y_L \\ Z_L \\ 1 \end{bmatrix}, \quad (1)$$

where c is the scaling factor; u and v denote the pixel coordinates of the image; K denotes the internal orientation parameter matrix of the camera, which comprises f_x, f_y, u_0, and v_0; M denotes the external orientation parameter matrix, which comprises rotation matrix R and translation matrix T; and X_L, Y_L, and Z_L denote point cloud coordinates.

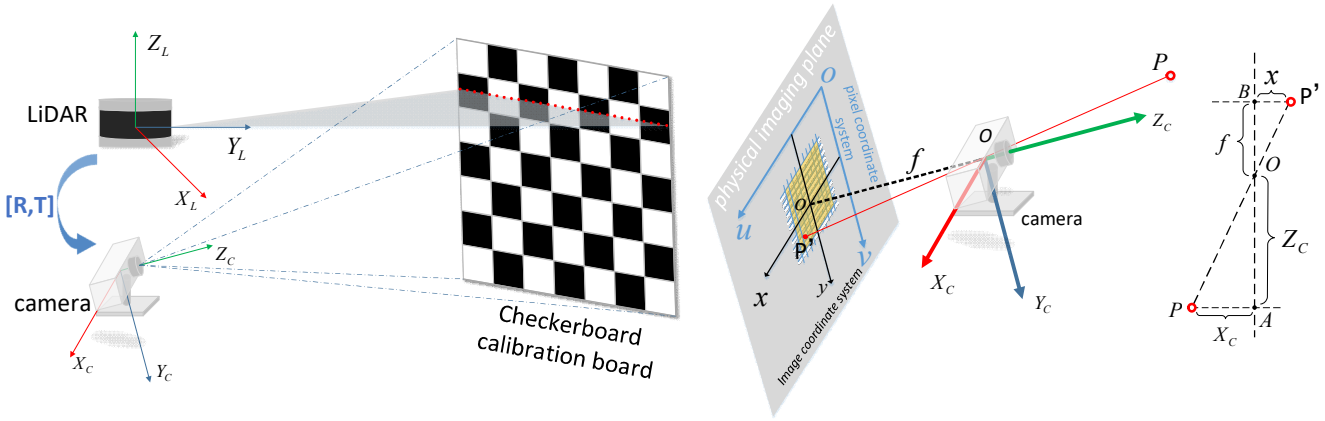
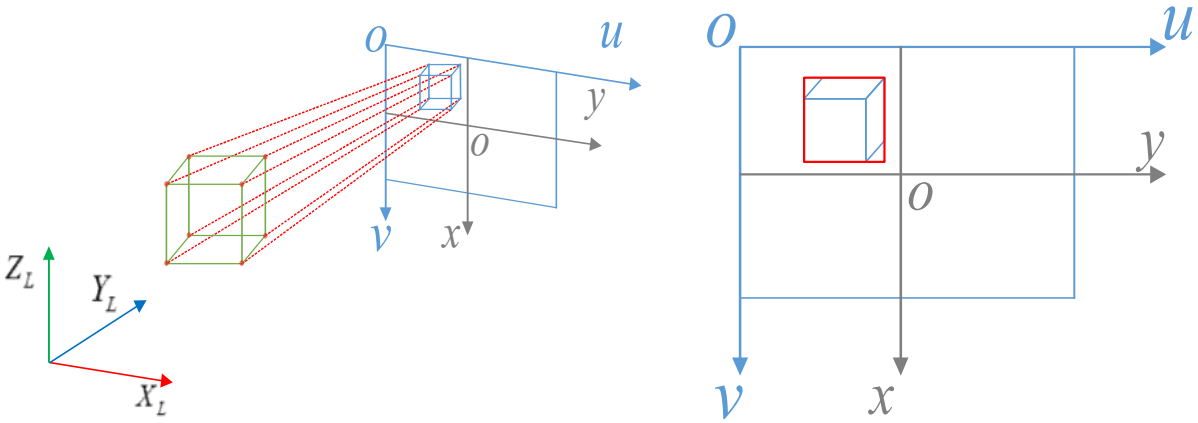


Figure 2. The principle of the LiDAR—camera joint calibration. $O_L - X_L Y_L Z_L$ is the LiDAR coordinate system; $O_C - X_C Y_C Z_C$ is the camera coordinate system; xoy is the image coordinate system; and uov is the image pixel coordinate system.

The detection results of 3D point cloud object detection represent the pose of the detected 3D point cloud stereo detection box through five parameters: length, width, height, the coordinates of the target center in 3D space, and global yaw angle. The 8 vertices of the 3D point cloud stereo detection box can be calculated. Based on the constructed LiDAR coordinate system in relation to the pixel coordinate system conversion model, the 3D point cloud stereo detection box can be presented in a two-dimensional plane via projection onto the image pixel plane, as shown in Figure 3a.



(**a**) (**b**)

Figure 3. (**a**) Projection of 3D point cloud stereo detection boxes in images; (**b**) construction of the minimum bounding rectangle box.

PointPillars detection point cloud data will output the information of the 3D point cloud stereoscopic target detection box. Based on these data, the 3D coordinate information of each vertex of the stereoscopic detection box can be calculated in the LiDAR coordinate system. The 8 vertices of the 3D point cloud stereoscopic detection box can be projected onto the image plane by Formula (1). In Figure 3a, the image formed by mapping the 3D point cloud stereo detection box onto the 2D image pixel plane is presented as a polygon. The method of establishing a minimum bounding rectangle box for polygon images is adopted. This box is transformed into a 2D rectangular detection box, with each side

being parallel to the two coordinate axes of the image pixel coordinate system, as shown in Figure 3b.

2.2.1. Target Box IoU Matching Strategy

After the transformation from a point cloud 3D detection box into a point cloud 2D projection box in a pixel coordinate system, overlapping point cloud 2D projection boxes and image target detection boxes will appear within the image field of view. In this paper, the target box intersection-over-union matching strategy is adopted to achieve the fusion of point cloud projection boxes and image target detection boxes.

The images captured by the camera can present the full view of vehicle and pedestrian targets, but the use of LiDAR remains limited by targets' physical properties. For example, when scanning pedestrian targets at close distances, there are few point clouds in the lower bodies of pedestrians. As another example, when scanning a vehicle, only a portion of the point cloud at the front, rear, or one side of the vehicle body can be obtained. Therefore, there may be situations where the size of the two-dimensional projection box of the final point cloud and the size of the image target detection box are not identical. Directly comparing the two can produce certain errors. Therefore, this paper introduces the method of calculating the probability of center-point distance, finding the detection box with the closest distance between center points to form a matching pair, and then further completing IoU matching fusion.

YOLO algorithm will output object detection box information when detecting pedestrians and vehicles in the image. Assuming that n detection boxes are detected in the image, $C = \{C_1, C_2, \cdots, C_i, \cdots C_n\}$ is the object detection box present in the image field of view. $C_i = (cx, cy, w, h)$ denotes the detection box information, where cx represents the x coordinates of the detection box center point; cy represents the y coordinates of the detection box center point; w represents the width of the detection box; and h represents the height of the detection box. PointPillars detection point cloud data will output 3D point cloud stereo detection box information. Each 3D point cloud stereo detection box is represented by $(x, y, z, w_1, l, h_1, \theta)$, where (x, y, z) represents the central position coordinates of the 3D detection box in the LiDAR coordinate system; (w_1, l, h_1) represents the width, length, and height of the 3D detection box, and θ represents the orientation angle of the 3D detection box around the Z axis. According to the above data, the 3D coordinate information of the stereo detection box in the LiDAR coordinate system can be obtained. The point cloud projection box information in the image can be obtained by Formula (1). Assuming that m projection boxes are obtained, $L = \{L_1, L_2, \cdots, L_J, \cdots, L_M\}$ is the point cloud 2D projection box. $L_j = (lx, ly, w_2, h_2)$ denotes the point cloud projection box information, where the lx is the x coordinate of the center point; ly is the y coordinate of the center point; w_2 is the width and h_2 is the height of the point cloud projection box. The serial numbers of the image object detection box and the 2D point cloud projection box are i and j, respectively. The image object detection box and the 2D point cloud projection box will overlap.

The center-point distance probability calculation schematic diagram is shown in Figure 4, where C_i represents the image object detection box detected by the camera; c_i represents the center point of region C_i; L_j represents the 2D point cloud projection box; l_j represents the center point of region L_j; c denotes the minimum diagonal length of the smallest outer bounding box formed by regions C_i and L_j; and d (c_i, l_j) denotes the Euclidean distance between c_i and l_j. The probability formula used to determine the distance from the center point is shown in Formula (2).

$$P_{ij} = 1 - \frac{d(c_i, l_j)^2}{c^2}, \tag{2}$$

The larger the P_{ij}, the closer the distance between the center points of the two region boxes. If P_{ij} has thresholds greater than δ, it is necessary to employ IOU matching fusion.

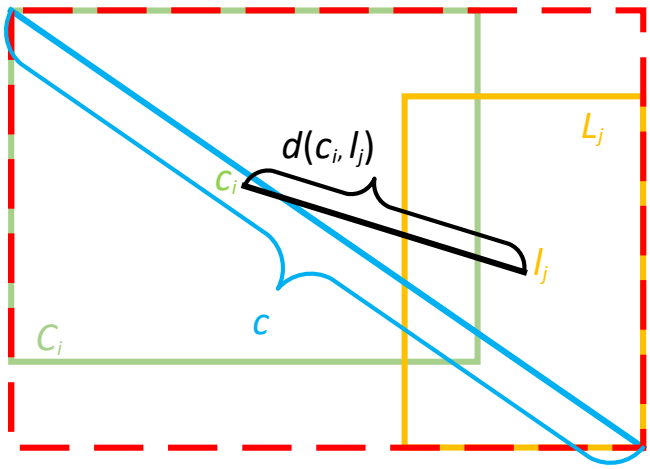


Figure 4. Schematic diagram of the center-point distance probability calculation.

In Figure 5, the area of the 2D projection box of the point cloud is represented by S_{LiDAR}, the area of the image target detection box is represented by S_{Camera}, and the area where SLC represents the two overlaps. The calculation formula for the intersection-over-union ratio of the two is shown in Formula (3).

$$IoU_{LC} = \frac{S_{LC}}{S_{LiDAR} + S_{Camera} - S_{LC}}, \tag{3}$$

where the thresholds are set to α and β ($\beta > \alpha$). When $IoU_{LC} < \alpha$, the objects are independent, and no matching fusion is performed. When $\alpha < IoU_{LC} < \beta$, it can be determined that the two are the same target, and the intersection area of the two can be used as the fused target detection box. When $IoU_{LC} > \beta$, it is determined that the two are the same target, and the minimum bounding box shared by the two is established as the fused target detection box.

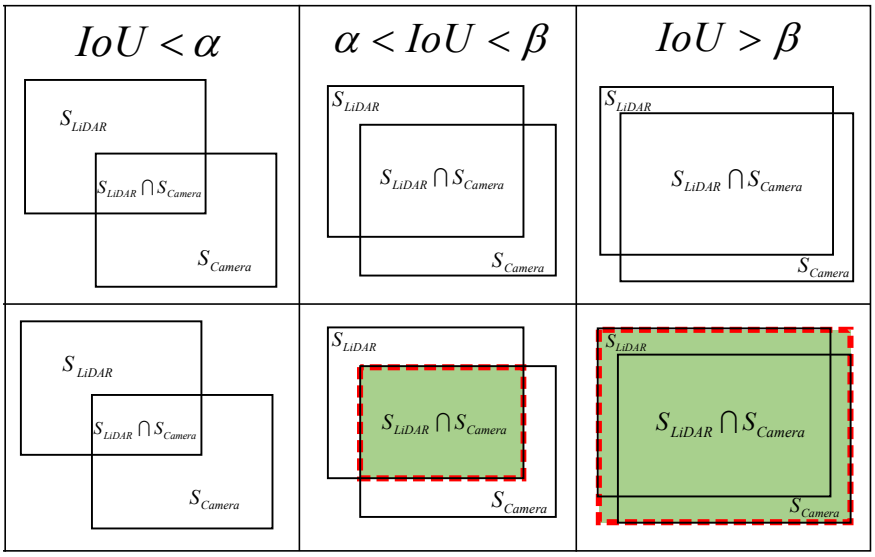


Figure 5. Schematic diagram of the target box intersection-over-union ratio matching fusion strategy.

2.2.2. D–S Theory for Class Confidence Fusion

D–S theory is very applicable in decision-level fusion schemes using multiple sensors, combining evidence data provided by multiple independent information sources via the Dempster synthesis rule. Due to the significant impact of conflict factor K on the inference results, the Dempster synthesis rule has certain limitations. An improved D–S evidence

theory is applied, and the concept of evidence credibility is introduced, meaning that the degree of conflict utilization depends on the credibility of the evidence.

Assuming that there are evidence sets $E = \{E_1, E_2, \cdots, E_n\}$ in the same recognition framework $\Theta = \{\theta_1, \theta_2, \cdots, \theta_N\}$. For $\forall A \subset 2^{\Theta}, (2^{\Theta} = \{\phi, \{\theta_1\}, \{\theta_2\}, \cdots \{\theta_n\}, \{\theta_1, \theta_2\}, \cdots, \Theta\})$, $m : 2^{\Theta} \longrightarrow [0, 1]$ denote for basic probability allocation function. Furthermore, $m(\phi) = 0$; $0 \leq m(A) \leq 1, A \subset \Theta; \sum m(A) = 1, A \subset 2^{\Theta}$. Set $m = \{m_1, m_2, \cdots, m_n\}$ is the corresponding basic probability allocation functions of evidence sets, then the conflict factor k_{ij} between the two evidence sets i and j can be expressed as Formula (4).

$$k_{ij} = \sum_{\substack{A_i \cap A_j = \varphi \\ A_i \in E_i, A_j \in E_j}} m_i(A_i) m_j(A_j), \tag{4}$$

where A_i and A_j are focal elements of m_i and m_j, respectively.

Formula $\varepsilon = e^{-\overline{k}}$ is used to determine the credibility of evidence, where $\overline{k}$ represents the mean of the total conflict factors for each pair of evidence sets in n evidence sets, as shown in Formula 5. When the value of $\overline{k}$ is high, this indicates that there is a significant conflict between the evidence. As a is ε decreasing function of $\overline{k}$, a decrease in the value of a indicates lower evidence credibility.

$$\overline{k} = \frac{1}{n(n-1)/2} \sum_{i<j} k_{ij}, \quad i, j \leq n, \tag{5}$$

where n represents the number of sensors.

The improved Dempster synthesis rule is shown in Formula (6).

$$\begin{cases} m(A) = \sum\limits_{\substack{A_i \in E_i \\ \cap_{i=1}^{n} A_i = A}} \prod_{1 \leq i \leq n} m_i(A_i) + k \cdot \varepsilon \cdot q(A), \quad A \neq \varphi, X \\ m(\Theta) = \sum\limits_{\substack{A_i \in E_i \\ \cap_{i=1}^{n} A_i = \Theta}} \prod_{1 \leq i \leq n} m_i(A_i) + k \cdot \varepsilon \cdot q(\Theta) + k(1-\varepsilon) \\ m(\varphi) = 0 \end{cases}, \tag{6}$$

where $q(A) = \frac{1}{n} \sum\limits_{i=1}^{n} m_i(A)$ reflects the average support of evidence for A.

The improved DS theory makes a more accurate estimate of objects by combining the probability of objects detected by the point cloud projection box and the image object detection box. The final position of the detection target box is determined by calculating the center-point distance and IoU between the point cloud projection box and the image object detection box, and the final probability of the detection object is determined by DS.

2.3. Improved DeepSORT for Object Tracking

The DeepSORT algorithm combines cascading matching with IoU matching using deep data association measurement methods and adds new trajectory confirmation during the Kalman filter update trajectory process, lessening the problem of frequent ID replacement caused by dynamic target occlusion. However, in practical applications, the tracking performance of the DeepSORT algorithm is unstable. The target ID can change, especially when the target is completely occluded or when two dynamic targets intersect and overlap. Therefore, the paper proposes utilizing an improved DeepSORT algorithm with an unscented Kalman filter and combining target motion information with IoU matching, as shown in Figure 6.

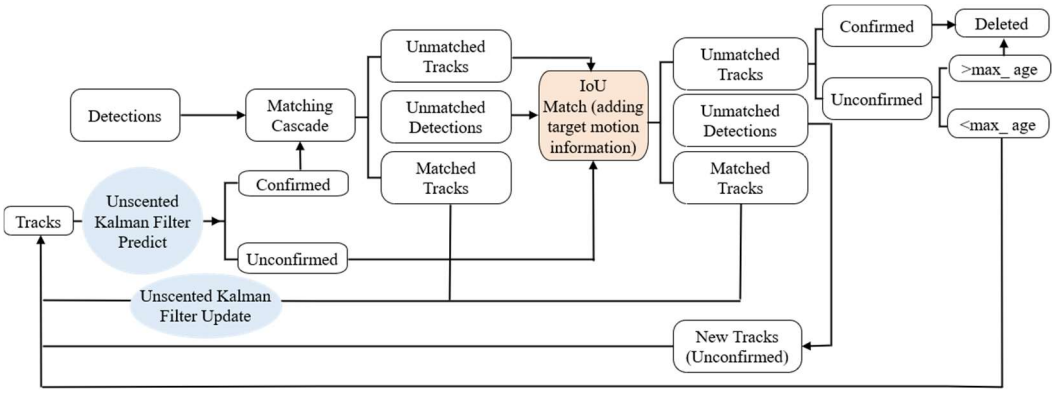


Figure 6. Schematic diagram of the improved DeepSORT algorithm.

2.3.1. Unscented Kalman Filter State Estimation

Unscented Kalman filtering is an unscented transform (UT) based on Kalman filtering and is used to handle the nonlinear transfer problem of probability density mean and covariance. The core idea of unscented transformation is to describe the Gaussian distribution of random variables through a series of sampling points and use nonlinear transformation to obtain a set of nonlinear function value points. The transformed state estimation and variance estimation can be calculated using the point set. The unscented Kalman filtering retains the nonlinear high-order term and uses the normal distribution, which is similar to the distribution expectation and variance of the nonlinear function to approximate, avoiding the calculation of complex Jacobian matrix, and the trajectory prediction effect is better. To improve the accuracy of dynamic target trajectory prediction, the Kalman filter of the DeepSORT algorithm can be replaced by the unscented Kalman filtering.

If a nonlinear function is $y = f(x)$, x will be an L-dimensional random variable, and the mean and variance of x will be $\overline{x}$ and P_x. A *sigma* sampling point set is constructed, where $\chi = \{\chi_0, \chi_1, \cdots, \chi_{2L}\}$.

$$\begin{cases} \chi_0 = \overline{x} \\ \chi_i = \overline{x} + \sqrt{(L+\lambda)P_x}, \ \ (i = 1, \cdots, L) \\ \chi_i = \overline{x} - \sqrt{(L+\lambda)P_x}, \ \ (i = L+1, \cdots, 2L) \\ \lambda = \alpha^2(L+\kappa) - L \end{cases}, \tag{7}$$

where λ is a scale factor; α determines the degree of hashing of this sampling point and is generally taken within $\left[1e^{-4}, \ 1\right]$; and κ is a scale parameter.

Combining the unscented transform with the Kalman filter, the state equation and observation equation of the system are assumed to be:

$$\begin{cases} x_t = F(x_{t-1}) + Q_t \\ y_t = H(x_t) + R_t \end{cases}, \tag{8}$$

where x_t and x_{t-1} are state vectors at time t and $t-1$; y_t is the observed quantity at time t; and Q_t and R_t are state transition noise and observation noise following a normal distribution.

By introducing the set of sampling points into the nonlinear equation of state, the following equation can be obtained:

$$x^i_{t|t-1} = F\left(x^i_{t|t-1}\right), \ \ i = 0, \cdots 2L, \tag{9}$$

It is possible to calculate the mean and covariance of state prediction at time t according to the weights corresponding to each sampling point.

$$\begin{cases} \tilde{x}^{i}_{t|t-1} = \sum_{i=0}^{2L} \omega_i^m x^i_{t|t-1} \\ \tilde{P}_{t|t-1} = \sum_{i=0}^{2L} \omega_i^c (x^i_{t|t-1} - \tilde{x}^{i}_{t|t-1})(x^i_{t|t-1} - \tilde{x}^{i}_{t|t-1})^T + Q_t \end{cases}, \tag{10}$$

By introducing the *sigma* sampling point set into the nonlinear observation equation, the following equation can be obtained:

$$y^i_{t|t-1} = H(x^i_{t|t-1}), \ \ i = 0, \cdots 2L, \tag{11}$$

The predicted mean, observation vector variance, and covariance of the observation vector at time t are calculated according to the weights corresponding to each *sigma* sampling point:

$$\begin{cases} \tilde{y}^{i}_{t|t-1} = \sum_{i=0}^{2L} \omega_i^m y^i_{t|t-1} \\ P_{yy} = \sum_{i=0}^{2L} \omega_i^c (y^i_{t|t-1} - \tilde{y}^{i}_{t|t-1})(y^i_{t|t-1} - \tilde{y}^{i}_{t|t-1})^T + R_t \\ P_{xy} = \sum_{i=0}^{2L} \omega_i^c (x^i_{t|t-1} - \tilde{y}^{i}_{t|t-1})(y^i_{t|t-1} - \tilde{y}^{i}_{t|t-1})^T \end{cases}, \tag{12}$$

With the Kalman filter gain $K_t = P_{xy} {P_{yy}}^{-1}$, the state estimate and the estimated variance at time t can be calculated:

$$\begin{cases} x^i_t = \tilde{x}^{i}_{t|t-1} + K_t(y_{t|t-1} - \tilde{y}^{i}_{t|t-1}) \\ P_{t|t} = P_{t|t-1} - K_t P_{yy} {K_t}^T \end{cases}, \tag{13}$$

The pseudo-code for the unscented Kalman filter is shown in Table 1.

Table 1. The pseudo-code for the unscented Kalman filter.

Pseudo-Code: Unscented Kalman Filtering
Initialization:
Select the number and location of sigma points
Assign weights to each sigma point
Initialize the state vector x and covariance matrix P
For each time step t:
Prediction Step:
For each sigma point X_sigma[i]:
X_sigma[i] = nonlinear_function(X_sigma[i]) // Apply nonlinear dynamic model
Compute the predicted mean and covariance:
X_pred = sum(W[i] * X_sigma[i]) // Weighted sum of sigma points
P_pred = sum(W[i] * (X_sigma[i] − X_pred)′ * (X_sigma[i] − X_pred)) //Weighted covariance
Update Step:
For each sigma point X_sigma[i]:
Z_sigma[i] = measurement_model(X_sigma[i]) // Apply nonlinear measurement model
Compute the measurement mean and covariance:
Z_pred = sum(W[i] * Z_sigma[i]) // Weighted sum of transformed sigma points
R = sum(W[i] * (Z_sigma[i] − Z_pred)′ * (Z_sigma[i] − Z_pred)) //Weighted covariance of measurements
Compute the Kalman gain K:
K = P_pred * H′ * inv(H * P_pred * H′ + R) // Kalman gain matrix
Update the state vector and covariance matrix:
X = X_pred + K * (Z − Z_pred) // State update
P = (I − K * H) * P_pred // Covariance update
Iteration:
Use the updated X and P for the next iteration

2.3.2. Improving Data Association in IoU Matching Modules

The IoU matching module of DeepSORT calculates the intersection-over-union ratio between the trajectory box and the detection box and determines whether it is associated with setting a threshold. In complex traffic scenes, tracking failures between dynamic targets caused by occlusion, intersection, and other factors are common. Simple IoU match-

ing cannot meet the requirements of accurate tracking matching. This section improves IoU matching via the addition of target motion information. Simultaneously, similarity is calculated by determining the Euclidean distance between trajectory and detection. This is then combined with improved IoU matching to improve the accuracy of matching.

It is necessary to define trajectory set $T = \{t_1, \cdots, t_i, \cdots, t_n\}$ and detection set $D = \{d_1, \cdots, d_j, \cdots, d_m\}$, where t_i and d_j are state vectors, each containing position, velocity, and direction information. It is also necessary to calculate the corresponding distance by introducing position, velocity, and direction factors into IoU matching. The position distance can be calculated using IoU, and the expression is shown in Formula (14).

$$d_c(i,j) = 1 - IoU(c_i, c_j), \tag{14}$$

where $d_c(i,j)$ represents the position distance; c_i, c_j represent the positions of the detection box and the trajectory box, respectively.

The distance in the direction of velocity can be calculated by the relative velocity between the center point of the detection box and the trajectory box, as shown in Formula (15).

$$d_v(i,j) = \frac{|v_i - v_j|}{\sqrt{(w^2 + h^2)}}, \tag{15}$$

where $d_v(i,j)$ represents the velocity distance; v_i and v_j represent the relative velocity of the detection box and the trajectory box, respectively; and w and h are the width and height of the bounding boxes on both sides.

The distance in the direction of motion is calculated using the angle difference between the center point of the detection box and the trajectory box in the direction under consideration, as shown in Formula (16).

$$d_\theta(i,j) = 1 - cos(\theta_i, \theta_j), \tag{16}$$

where $d_\theta(i,j)$ represents the velocity distance and θ_i and θ_j represent the motion direction of the detection box and the trajectory box, respectively.

The improved IOU matching can be obtained by weighting the sum of Formulas (14)–(16), as shown in Formula (17).

$$IoU(t_i, d_j) = w_c d_c(i,j) + w_v d_v(i,j) + w_\theta d_\theta(i,j), \tag{17}$$

where w_c, w_v, and w_θ represent the weights of position, velocity, and directional distance, respectively.

The Euclidean distance can be calculated based on the state vectors of detection and trajectory, as shown in Formula (18). Normalizing the value obtained yields the similarity score for detection and trajectory, as shown in Formula (19).

$$d(t_i, d_j) = \sqrt{\sum_{i=1}^{n} \sum_{j=1}^{m} (t_i - d_j)^2}, \tag{18}$$

$$s(t_i, d_j) = \frac{1}{1 + d(t_i, d_j)}, \tag{19}$$

By combining improved IoU matching and similarity score, the final correlation score is obtained via weighted summation, as shown in Formula (20).

$$s(t_i, d_j) = w_s s(t_i, d_j) + w_{IoU} IoU(t_i, d_j), \tag{20}$$

Then, it is necessary to define the threshold function as shown in Formula (21), compare $s(t_i, d_j)$ with the set threshold, and retain the detection results and tracking objects above the threshold.

$$matching(i,j) = \begin{cases} 1, & s(t_i, d_j) \geq threshold \\ 0, & overwise \end{cases}, \quad (21)$$

3. Experimental Preparation and Data Introduction

As shown in Figure 7, the RoboSense LiDAR device and USB monocular camera are assembled using the existing AV chassis equipment available in the laboratory. The parameters related to the camera and LiDAR are shown in Tables 2 and 3.

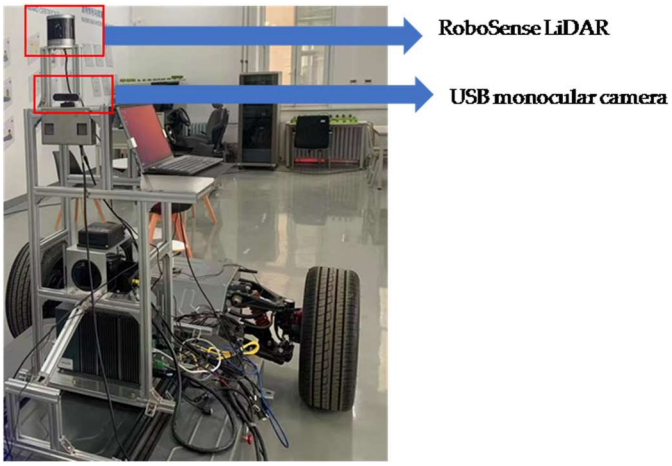


Figure 7. Experimental equipment pictures.

Table 2. Parameters of the camera.

Item	Parameter
Model	Q20
Maximum resolution	1920 ×1800
Pixel	4M pixel
Frame rate	30 FPS

Table 3. Parameters of the LiDAR.

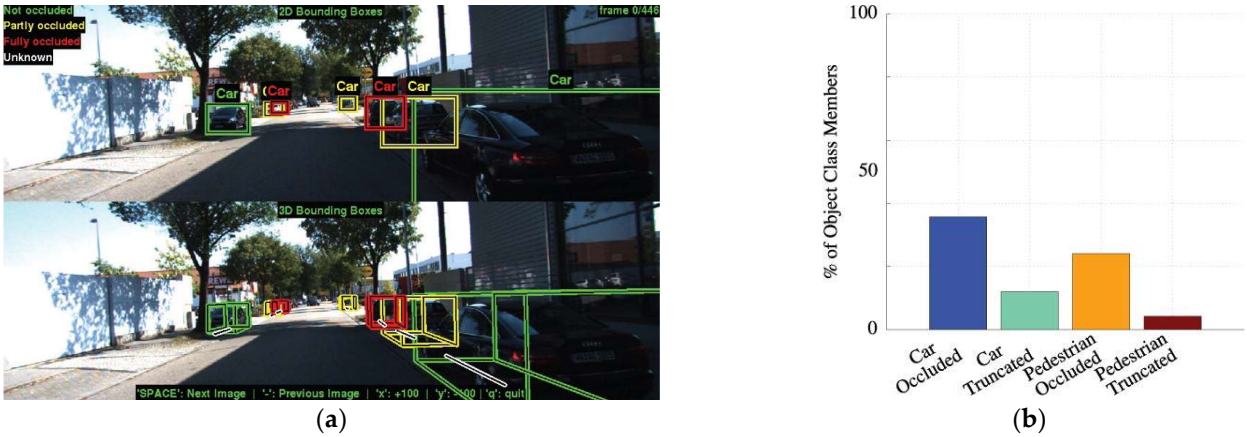
Item	Parameter
Number of lines	16
Frame rate	10 Hz
Laser wavelength	905 nm
Range capability	150 m
Accuracy	±2.0 cm
HFOV	360°
VFOV	30°
Horizontal resolution	0.4°
Vertical resolution	2.0°

The experiments were conducted on a notebook computer equipped with an Intel i5 12400 processor and NVIDIA GeForce RTX 3060 graphics. In doing so, widely used deep learning frameworks and image processing libraries commonly applied in object detection and tracking tasks are employed. The experimental hardware and parameter settings are summarized in Table 4.

Table 4. Experimental hardware and parameter settings.

Item	Parameter
Operating system	Ubuntu 18.04
CPU	Intel(R) Core i5-12400
Memory	16 GB
GPU	NVIDIA GeForce RTX 3060
Graphics memory	12 GB
CUDA version	Cuda 11.1 + CuDNN 8.6.0
Development language	Python 3.8
Deep learning framework version	PyTorch 1.12

As shown in Figure 8. the KITTI [35] dataset is used as the basis for validating and analyzing the performance of the proposed method. Front-view images of the KITTI 3D dataset provide a resolution of 1280 × 384 pixels. The KITTI dataset has 7518 images available for testing and 7481 images for training. The object tracking benchmark includes 29 test sequences and 21 training sequences. In addition to the camera parameters and the RGB images, the KITTI dataset provides additional data that can be further used in 3D-image-based detection tasks. The KITTI dataset provides frames that precede the LiDAR signal in time. The detection of vehicles in KITTI images is conducted with the help of the calibration files, label files, and LiDAR data that are provided by the KITTI dataset. Three-dimensional bounding boxes are generated at the coordinates of the vehicles using label files provided by the KITTI dataset.



(**a**) (**b**)

Figure 8. KITTI dataset [32]. (**a**) The projection of 3D point clouds into the image plane; (**b**) the total number of objects for the two predominant classes, "car" and "pedestrian".

4. Experimental Results and Analysis

4.1. Experimental Analysis of Moving Object Detection

4.1.1. Evaluation Indicators

The experimental evaluation metrics used to quantitatively compare performances include FPS (frame/s, representing the detection speed of the algorithm), precision (reflects the ability of the model to correctly predict the precision of positive samples), FP rate (the ratio of the number of false detections to the total number of detections, reflecting the ability of the model to correctly predict the purity of positive samples), and miss rate (the ratio of the number of missed detections to the number of true obstacles, corresponding to the model's ability to correctly predict the purity of negative samples). Using these metrics, a comprehensive and accurate analysis of the proposed method is performed to obtain both accuracy and efficiency.

$$Precision = \frac{TP}{TP + FP} \times 100\%, \tag{22}$$

$$FP\ rate = \frac{FP}{TP + FN} \times 100\%, \tag{23}$$

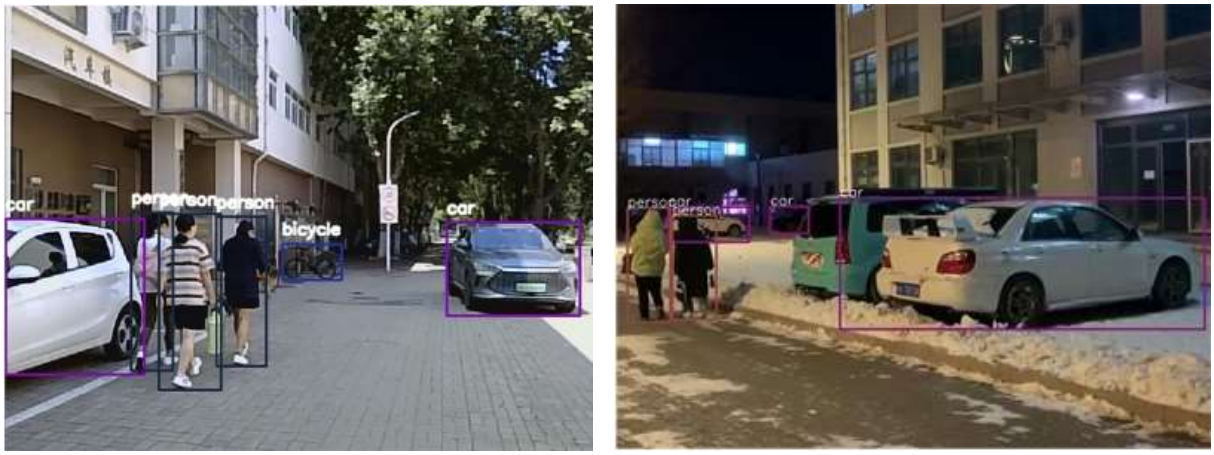
$$miss\ rate = \frac{FN}{TP + FN} \times 100\%, \tag{24}$$

where TP represents the positive samples that are correctly detected; FP represents the positive samples that are incorrectly objected; and FN represents the negative samples that are incorrectly objected.

4.1.2. Experiment Using Cameras

Using experimental equipment, the campus road-scene videos were recorded, and laser point cloud data were obtained. A real road-scene dataset was created by extracting frames from the video data stream, and the KITTI dataset was added to train the YOLOv5 network. The training set consisted of 2575 images (including 575 annotated self-collected data).

The method takes the collected video data stream (about 700 frames of images) as test data, including different scenes during the day and night. Subsequently, a YOLOv5 image recognition network is used to recognize the dynamic targets of vehicles and pedestrians in the self-collected video data. The recognition results are shown in Figure 9.



(**a**) (**b**)

Figure 9. The camera detection results in different road scenes. (**a**) During daytime; (**b**) during night.

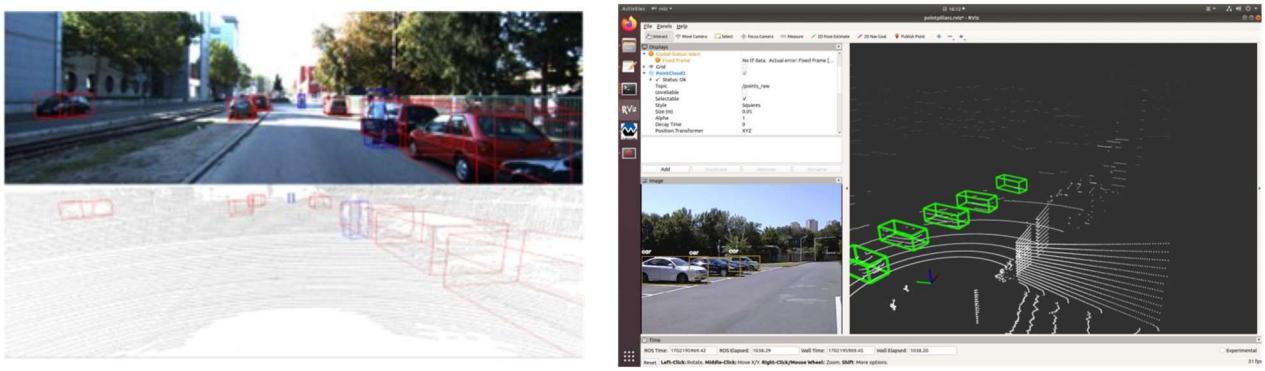
Using the recognition results above, the YOLOv5 image recognition network can recognize vehicle and pedestrian targets in different scenes, but it is more prone to false positives and missed detections in nighttime scenes. According to the analysis of experimental results, the detection results for cars and pedestrians in different scenes were statistically analyzed, and the statistical results are shown in Table 5. The precision of car detection during daytime was 92.73%, which was 3.19% higher than that at night. The precision of pedestrian detection during daytime was 91.24%, which was 4.96% higher than that at night. The FP rates of cars and pedestrians during daytime are 2.23% and 6.45% lower than those at night, respectively.

Table 5. Object detection statistical results for cars and pedestrians in different scenes.

Scene	Category	Precision (%)	FP Rate (%)	Miss Rate (%)
Day	car	92.73	0.83	1.04
	pedestrian	91.24	1.48	2.97
Night	car	89.54	3.06	3.18
	pedestrian	86.38	7.83	4.26

4.1.3. Experiment Using LiDAR

Each set of data in the KITTI dataset contains corresponding LiDAR point cloud data and RGB image data and provides internal parameters of the camera and external parameters between sensors, facilitating projection efforts between coordinate systems. Figure 10a shows the visualization results of PointPillars detecting information in the KITTI dataset. The 3D detection box in the figure is the output result of the Point Pillars algorithm network, which is projected onto a 2D image through the extrinsic parameters of the LiDAR device and cameras. It is necessary to verify the detection performance of the PointPillars algorithm in actual scenes through self-collected data, as shown in Figure 10b, which shows the visualization results obtained using RVIZ in the ROS system. The left side shows the RGB image of the actual scene captured by the camera, and the right side shows the point cloud image obtained with a 3D detection box.



(**a**) (**b**)

Figure 10. (**a**) Detection results of the PointPillars algorithm on a set of data from the KITTI dataset; (**b**) visualization of 3D point cloud target detection results using RVIZ under ROS.

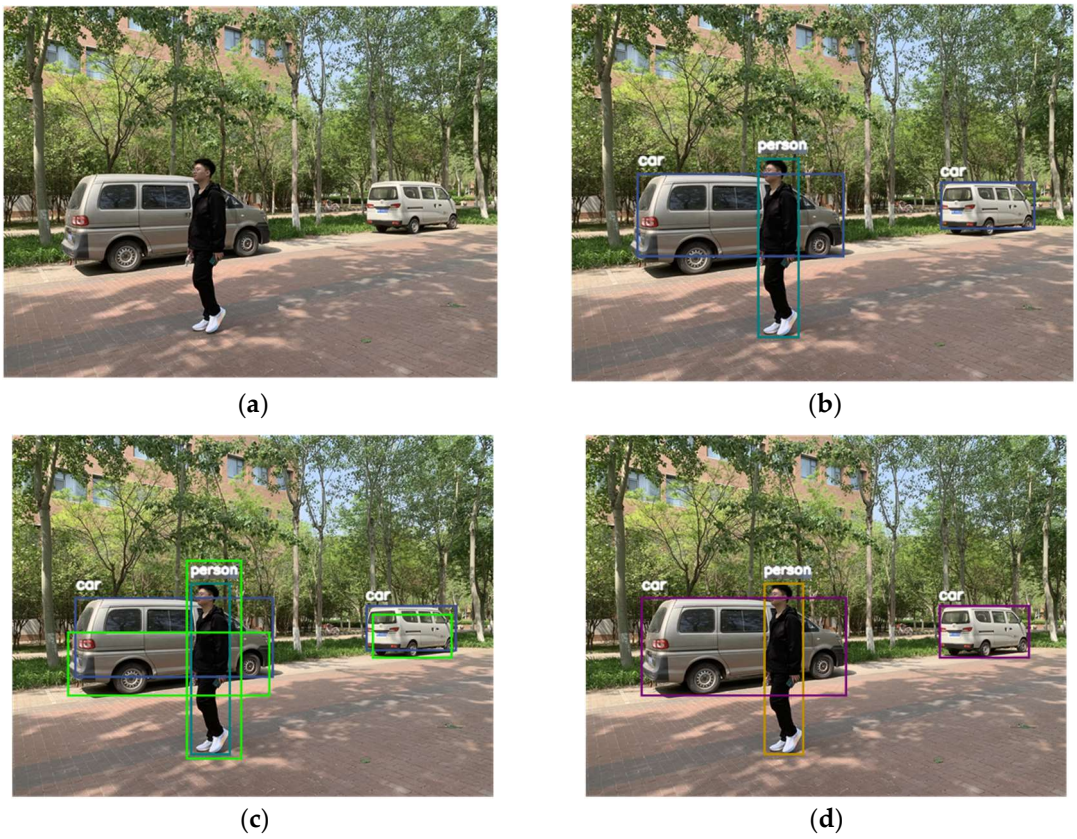
As shown in Figure 10b, in actual scenes, targets that maintain a certain distance from the LiDAR can be successfully detected, and the scale prediction is also relatively consistent with the true values. However, due to the vertical field angle of the LiDAR, pedestrian targets that are closer to the LiDAR can only be scanned by the LiDAR, with fewer point clouds obtained on the lower bodies of pedestrians, resulting in missed detections.

4.1.4. LiDAR—Camera Fusion

To verify the fusion algorithm proposed in this paper, the results of target detection were analyzed from two aspects: single sensors and data fusion. The fusion process in different daytime scenes is shown in Figure 11. As shown in the figure, the fused target box can more fully envelop car and pedestrian targets, and the fusion effect of the target box is satisfactory.

The data fusion results based on the improved D–S evidence theory class confidence fusion strategy are shown in Table 6. The values contained in the table represent the determinism of the model for the class of objects contained in the target box in the mathematical form $P_r(class_i|Object)$, whose class probability is calculated by the logistic regression function in the LiDAR and the camera detection algorithm classification model. Additionally, the sum of probabilities for all classes is 1.

The detection fusion results of actual nighttime scenes are shown in Figure 12. As shown in the figure, due to the dim lighting at night, the target box detected by the camera cannot fully envelop the pedestrian and vehicle targets. The 2D projection box of the point cloud can only roughly determine the position of the target. The fusion of the two target boxes has a strong recognition effect and can more completely envelop the vehicle and pedestrian targets. Table 7 shows the data fusion results obtained utilizing the confidence fusion strategy, based on the D–S evidence theory, in nighttime scenes.



(**a**) (**b**)

(**c**) (**d**)

Figure 11. Fusion process in the daytime scene, where (**a**) is the actual scene; (**b**) is the camera detection result; (**c**) is the result obtained by projecting the 3D detection frame of the point cloud onto the image, where the green frame is the 2D projection frame of the point cloud; and (**d**) is the final fusion result.

Table 6. Object detection statistical results based on D–S evidence theory in daytime scenes.

Object	Sensor	Probability of Pedestrian	Probability of Car	Uncertainty
Car	Camera	0.061	0.903	0.036
	LiDAR	0.048	0.925	0.024
	LiDAR—camera fusion	0.012	0.973	0.004
Pedestrian	Camera	0.934	0.042	0.021
	LiDAR	0.837	0.117	0.047
	LiDAR—camera fusion	0.954	0.020	0.005

(**a**) (**b**)

Figure 12. *Cont.*

(**c**) (**d**)

Figure 12. The fusion process in the night scene, where (**a**) is the actual scene; (**b**) is the camera detection result; (**c**) is the result obtained by projecting the 3D detection frame of the point cloud to the image, where the green frame is the 2D projection frame of the point cloud; and (**d**) is the final fusion result.

Table 7. Object detection statistical results based on D–S evidence theory in nighttime scenes.

Object	Sensor	Probability of Pedestrian	Probability of Car	Uncertainty
Car	Camera	0.126	0.832	0.038
	LiDAR	0.062	0.915	0.025
	LiDAR—camera fusion	0.003	0.941	0.005
Pedestrian	Camera	0.893	0.042	0.031
	LiDAR	0.834	0.123	0.051
	LiDAR—camera fusion	0.925	0.023	0.007

From Tables 6 and 7, the reader can see that the probabilities of detecting cars and pedestrians in daytime scenes based on the D–S evidence theory can reach 97.3% and 95.4%, respectively. Moreover, it can reach 94.1% and 92.5% in nighttime scenes, respectively. Furthermore, it is higher than that based on camera use or LiDAR devices, respectively. The LiDAR—camera fusion detection method based on the D–S evidence theory is more accurate and reliable.

4.2. Experimental Analysis of Moving Object Tracking

4.2.1. Evaluation Indicators

Multiple-object tracking accuracy (MOTA), multiple-object tracking precision (MOTP), Identification F-Score (IDF1), and Higher Order Tracking Accuracy (HOTA) values are employed as performance evaluation indicators for the original DeepSORT algorithm and the improved DeepSORT algorithm.

MOTA is a comprehensive indicator used for measuring error tracking, omission tracking, and ID switching output, as shown in Formula (25).

$$\mathrm{MOTA} = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t}, \tag{25}$$

represents the positive samples that are correctly detected.

MOTP is used to measure the degree of matching between tracking results and true values, as shown in Formula (26).

$$\mathrm{MOTP} = \frac{1}{\sum_{i=1}^{N} m_i} \sum_{i=1}^{N} \sum_{j=1}^{m_i} d_{i,j}, \tag{26}$$

where $d_{i,j}$ denotes the error between the estimated position and the true position of object j for time i, and where m_i denotes the number of correctly tracked objects in time i.

IDF1 is the ratio of the number of correct target detections to the average of the sum of the true and calculated detections; here, the IDF1 score is calculated as follows:

$$\text{IDF1} = \frac{IDTP}{IDTP + 0.5IDFP + 0.5IDFN}, \tag{27}$$

where *IDTP* can be viewed as the number of detected targets that are correctly assigned during tracking, *IDFN* as the number of detected targets that are missed during tracking, and IDFP as the number of detected targets that are incorrectly assigned during tracking.

HOTA unifies accurate detection, correlation tracking, and localization in a unified metric, calculated as follows:

$$\text{HOTA} = \int_0^1 HOTA_\alpha d\alpha \approx \frac{1}{19}\sum_{\alpha \in \left\{ \begin{array}{l} 0.05, 0.1, \ldots, \\ 0.9, 0.95 \end{array} \right\}} HOTA_\alpha, \tag{28}$$

$$HOTA_\alpha = \sqrt{\frac{\sum_{C \in \{TP\}} A(c)}{|TP| + |FN| + |FP|}}, \tag{29}$$

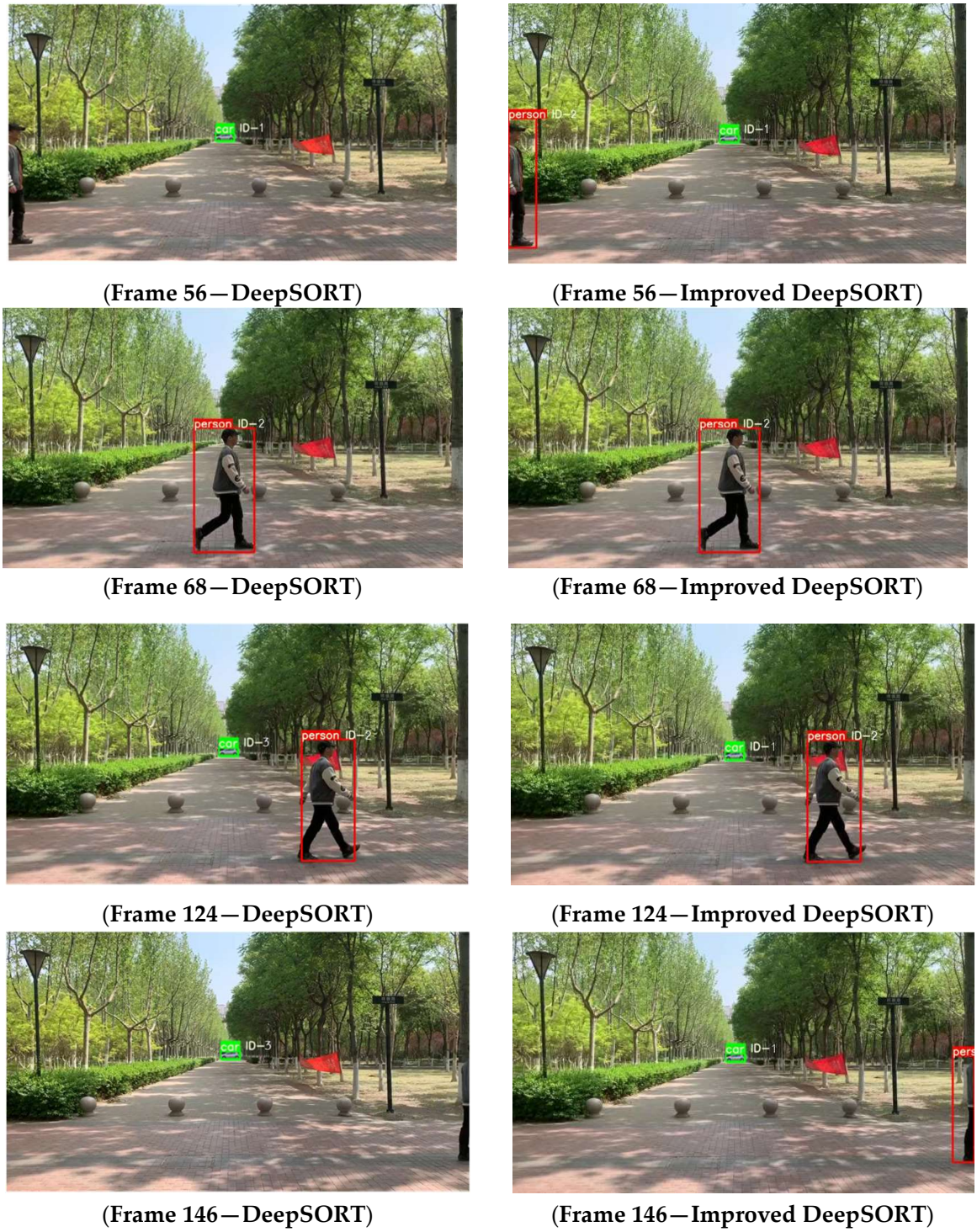
$$A_C = \frac{|TPA(c)|}{|TPA(c)| + |FNA(c)| + |FPA(c)|} \tag{30}$$

where α is the similarity localization threshold, *TPA* denotes true-positive association, FNA denotes false-negative association, and *FPA* denotes false-positive association.

4.2.2. Experiment for Pedestrians

The test results of the original DeepSORT algorithm and the improved DeepSORT algorithm when using self-collected data are shown in Figure 13. In frames 56, 68, 124, and 146, the improved DeepSORT algorithm illustrates superior robustness to the original DeepSORT algorithm. The improved DeepSORT algorithm tracks the target ID-2 at frame 56 before it fully enters the field of view, while the original DeepSORT algorithm does not start tracking the ID-2 target at frame 56. In frame 146, the original DeepSORT algorithm ceases to track the target ID-2, while the improved DeepSORT algorithm tracks the ID-2 target until it completely leaves the field of view. In addition, from frames 68 and 124, the original DeepSORT algorithm's ID-1 target is occluded and re-labeled as ID-3, while the improved DeepSORT algorithm's ID-1 target's ID remains unchanged after being occluded.

Comparing the tracking results of the two algorithms shown in Figure 14, the target ID-1 is first occluded by a tree at frame 56, subsequently overlapping with target ID-2 at frame 64. At this point, the original DeepSORT algorithm can no longer distinguish between the target numbers ID-1 and ID-2. In frame 72, the target ID-2 is occluded by a tree, and the previous target number ID-1 is mistakenly recognized as ID-2. In frame 80, when the previous ID-2 is occluded once again, the identification number changes to ID-4. The original DeepSORT algorithm performs poorly when tracking targets that are occluded or that overlap. However, the improved DeepSORT algorithm maintains the same ID number for each target throughout the tracking process when tracking results in the same scene, with the same ID utilized from frame 28 to frame 80, effectively addressing the problem of the occlusion or overlap of targets.



(**Frame 56—DeepSORT**) (**Frame 56—Improved DeepSORT**)

(**Frame 68—DeepSORT**) (**Frame 68—Improved DeepSORT**)

(**Frame 124—DeepSORT**) (**Frame 124—Improved DeepSORT**)

(**Frame 146—DeepSORT**) (**Frame 146—Improved DeepSORT**)

Figure 13. The tracking effect of the original DeepSORT algorithm and improved DeepSORT algorithm on self-collected data. Frames 56, 68, 124, and 146 were selected for comparison. The left picture shows the tracking effect of the original DeepSORT algorithm, and the right picture shows the tracking effect of the improved DeepSORT algorithm.

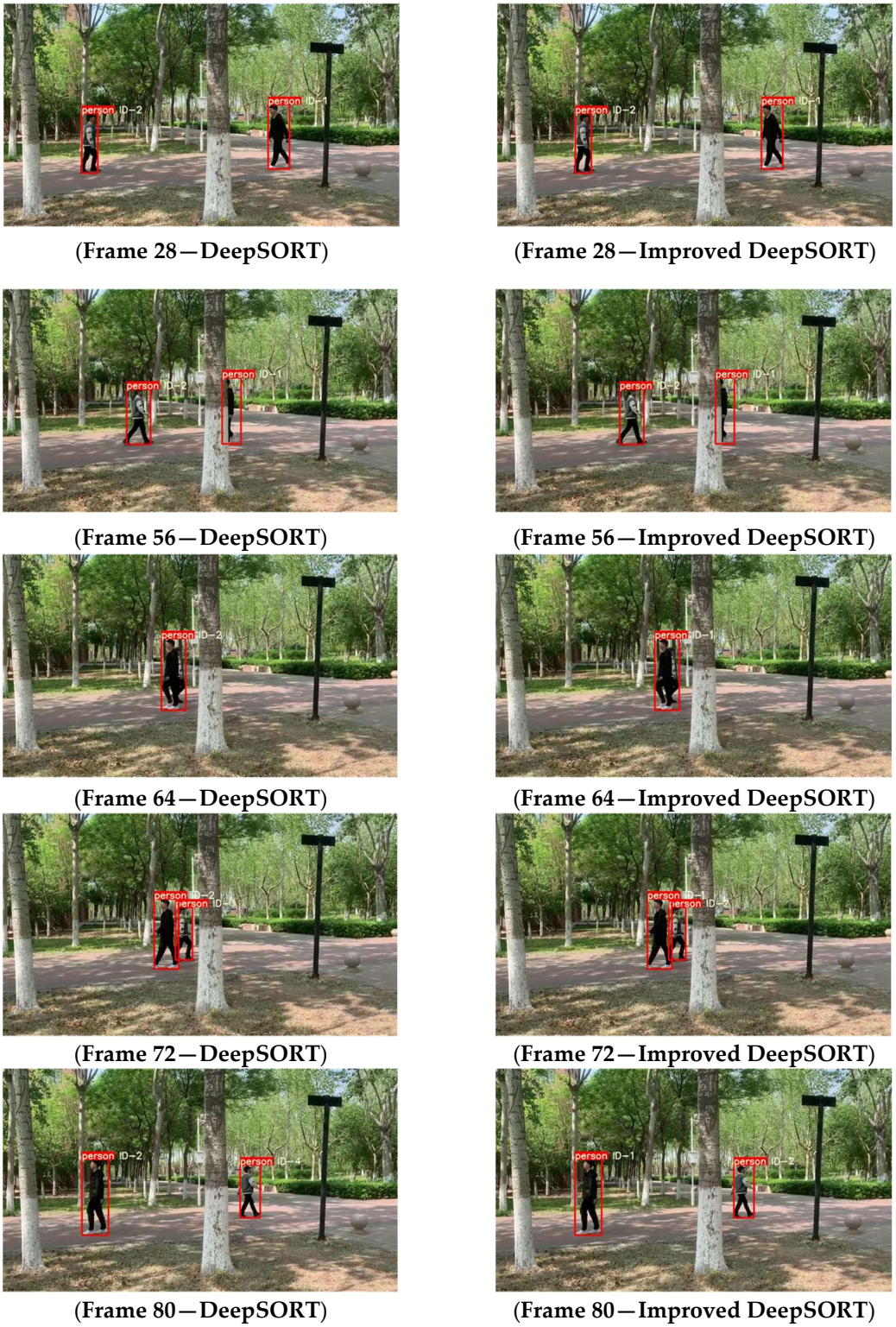
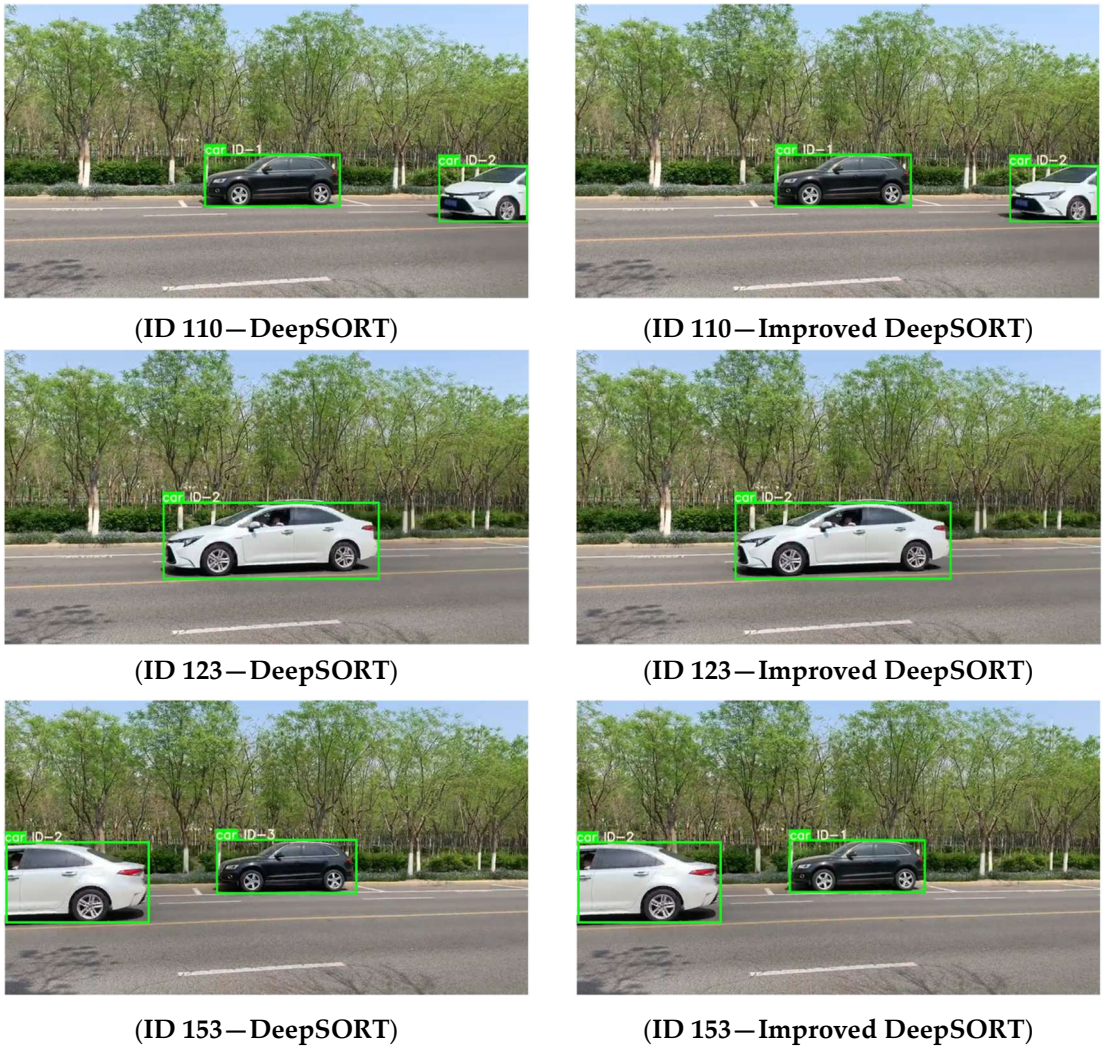


Figure 14. The tracking effect of the original DeepSORT algorithm and improved DeepSORT algorithm on self-collected data. Frames 28, 56, 64, 72 and 80 were selected for comparison. The left picture shows the tracking effect of the original DeepSORT algorithm, and the right picture shows the tracking effect of the improved DeepSORT algorithm.

4.2.3. Experiment for Cars

Figure 15 shows the tracking of cars when performed using two algorithms. In frame 123, the white car completely occludes the black car. When the black car reappears,

the original DeepSORT algorithm's tracking result in frame 153 shows a change in the white car's ID, while the improved DeepSORT algorithm retains the original white car ID unchanged.



(ID 110—DeepSORT) **(ID 110—Improved DeepSORT)**

(ID 123—DeepSORT) **(ID 123—Improved DeepSORT)**

(ID 153—DeepSORT) **(ID 153—Improved DeepSORT)**

Figure 15. The tracking effect of the original DeepSORT algorithm and improved DeepSORT algorithm on self-collected data. Frames 110, 123, and 153 were selected for comparison. The left picture shows the tracking effect of the original DeepSORT algorithm, and the right picture shows the tracking effect of the improved DeepSORT algorithm.

The original DeepSORT algorithm and the improved DeepSORT algorithm were experimentally validated via the use of self-collected data. The performance of the four different algorithms was compared using MOTA, MOTP, HOTA, and IDF1 indicators, and the comparison results are shown in Table 8. The results in the table show that the improved DeepSORT algorithm performs well in terms of MOTA and MOTP metrics.

Table 8. Comparison results of object detection statistics.

Methods	MOTA	MOTP	HOTA	IDF1
SORT	0.49	0.62	0.46	0.51
ByteTrack	0.60	0.71	0.52	0.57
DeepSORT	0.56	0.74	0.53	0.59
Improved DeepSORT	0.66	0.79	0.61	0.72

5. Conclusions

The fusion of camera and LiDAR data has become an essential process in efforts to overcome the shortcomings of individual sensor types and improve the efficiency and reliability of autonomous vehicles. This paper presents research on moving object detection and tracking technology based on camera and LiDAR data fusion. First, based on the calibration of cameras and LiDAR devices, YOLOv5 and PointPillars network models are used to perform object detection with image and point cloud data. When using the YOLOv5 image recognition network, the precision of car and pedestrian detection during the daytime is 92.73% and 91.24%, respectively. The FP rate for cars and pedestrians during daytime is 2.23%, which is 6.45% lower than that at night, respectively. A target box IoU matching strategy, based on center-point distance probability, and the improved D–S theory are used for class confidence fusion to obtain the final fusion detection result. The probability of detecting cars and pedestrians in daytime scenes can reach 97.3% and 95.4%, respectively. Moreover, it can reach 94.1% and 92.5% in nighttime scenes, respectively. Additionally, it is higher than that obtained using a camera or LiDAR alone. Through the method developed, more accurate and reliable fusion detection results are obtained compared with the final output of a single sensor. In the process of moving object tracking, an unscented Kalman filter is used to accurately predict the motion state of nonlinear objects, and object motion information is added to the IoU matching module to improve the matching accuracy in the data association process. Through self-collected data verification, fusion detection, and tracking is significantly better than that of a single sensor. The evaluation index values of the improved DeepSORT algorithm are 66% for MOTA, 79% for MOTP, 0.61 for HOTA, and 0.76 for IDF1, which are, respectively, 10%, 5%, 8%, and 13% higher than those of the original DeepSORT algorithm. The improved DeepSORT algorithm effectively solves the problem of tracking instability caused by the occlusion of moving objects.

At present, the field of AV environment perception is still in the stage of continuous exploration. The perception technology based on camera and LiDAR data fusion studied in this paper can achieve good target detection and tracking effects, which can provide relevant reference value when performing studies in the field of autonomous vehicle perception. In subsequent research, information from millimeter-wave radar can be integrated on this basis. The precise velocity information provided by millimeter-wave radar can be used to predict target motion status more accurately in dynamic target-tracking tasks. In addition, the SOTA target detection model with Yolov9 and other SOTA targets will be investigated subsequently to evaluate the computational latency of the proposed camera—LIDAR fusion method to achieve better real-time detection.

Author Contributions: Conceptualization, Q.C. and Y.X.; Data curation, Z.D. and F.S.; Formal analysis, Z.G. and F.S.; Investigation, Z.D.; Methodology, Q.C.; Project administration, Q.C.; Software, F.S.; Visualization, Z.G., Z.D. and Y.X.; Writing—original draft, Q.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Project of the Tianjin Education Committee, grant number 2021KJ018; the University Foundation of the Tianjin University of Technology and Education, grant number KYQD202107; the Tianjin Science and Technology Plan, grant number 23YDTPJC00980; the Tianjin Applied Basic Research Project, grant number 22JCZDJC00390, the Shandong Province Major Science and Technology Innovation Project (grant number: 2023CXGC010111) and the Graduate Education Quality Course Construction Project—Intelligent Automotive Sensor Technology (grant number: 4053223015).

Institutional Review Board Statement: Not applicable.

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

Data Availability Statement: The KITTI dataset can be found at https://www.cvlibs.net/datasets/kitti/ (accessed on 17 November 2023). The other data that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments: The authors would like to thank the Tianjin University of Technology and Education and the Tianjin Education Committee for their support of this work.

Conflicts of Interest: Yi Xu is employee of QINGTE Group Co., Ltd. The paper reflects the views of the scientists, and not the company.

References

1. Badue, C.; Guidolini, R.; Carneiro, R.V.; Azevedo, P.; Cardoso, V.B.; Forechi, A.; Jesus, L.; Berriel, R.; Paixao, T.M.; Mutz, F.; et al. Self-driving cars: A survey. *Expert Syst. Appl.* **2021**, *165*, 113816. [CrossRef]
2. Bishop, R. Intelligent vehicle applications worldwide. *IEEE Intell. Syst. Their Appl.* **2000**, *15*, 78–81. [CrossRef]
3. Lan, Y.; Huang, J.; Chen, X. Environmental perception for information and immune control algorithm of miniature intelligent vehicle. *Int. J. Control Autom.* **2017**, *10*, 221–232. [CrossRef]
4. Mozaffari, S.; Al-Jarrah, O.Y.; Dianati, M.; Jennings, P.; Mouzakitis, A. Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review. *IEEE Trans. Intell. Transp. Syst.* **2020**, *23*, 33–47. [CrossRef]
5. Mehra, A.; Mandal, M.; Narang, P.; Chamola, V. ReViewNet: A Fast and Resource Optimized Network for Enabling Safe Autonomous Driving in Hazy Weather Conditions. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 4256–4266. [CrossRef]
6. Liu, X.; Baiocchi, O. A comparison of the definitions for smart sensors, smart objects and Things in IoT. In Proceedings of the 2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, Canada, 13–15 October 2016.
7. Fayyad, J.; Jaradat, M.A.; Gruyer, D.; Najjaran, H. Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review. *Sensors* **2020**, *20*, 4220. [CrossRef] [PubMed]
8. Chen, X.D.; Zhang, J.C.; Pang, W.S.; Ai, D.H.; Wang, Y.; Cai, H.Y. Key Technology and Application Algorithm of Intelligent Driving Vehicle LiDAR. *Opto-Electron. Eng.* **2019**, *46*, 190182. [CrossRef]
9. Fan, J.; Huang, Y.; Shan, J.; Zhang, S.; Zhu, F. Extrinsic calibration between a camera and a 2D laser rangefinder using a photogrammetric control field. *Sensors* **2019**, *19*, 2030. [CrossRef]
10. Vivet, D.; Debord, A.; Pagès, G. PAVO: A Parallax based Bi-Monocular VO Approach for Autonomous Navigation in Various Environments. In Proceedings of the DISP Conference, St Hugh College, Oxford, UK, 29–30 April 2019.
11. Mishra, S.; Osteen, P.R.; Pandey, G.; Saripalli, S. Experimental Evaluation of 3D-LIDAR Camera Extrinsic Calibration. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 9020–9026. [CrossRef]
12. Kanezaki, A.; Suzuki, T.; Harada, T.; Kuniyoshi, Y. Fast object detection for robots in a cluttered indoor environment using integral 3D feature table. In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 4026–4033. [CrossRef]
13. Jeong, J.; Cho, Y.; Kim, A. The road is enough! Extrinsic calibration of non-overlapping stereo camera and LiDAR using road information. *IEEE Robot. Autom. Lett.* **2019**, *4*, 2831–2838. [CrossRef]
14. Lv, X.; Wang, B.; Ye, D.; Wang, S. LCCNet: LiDAR and Camera Self-Calibration using Cost Volume Network. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Nashville, TN, USA, 19–25 June 2021; pp. 2888–2895. [CrossRef]
15. Wu, X.; Zhang, C.; Liu, Y. Calibrank: Effective Lidar-Camera Extrinsic Calibration by Multi-Modal Learning to Rank. In Proceedings of the IEEE International Conference on Image Processing, Abu Dhabi, United Arab Emirates, 25–28 October 2020; pp. 3189–3193.
16. Gong, X.; Lin, Y.; Liu, J. Extrinsic calibration of a 3D LIDAR and a camera using a trihedron. *Opt. Lasers Eng.* **2013**, *51*, 394–401. [CrossRef]
17. Li, M.L.; Dai, B.; Li, Z.; He, H.B. High-precision Calibration of Placement Parameters between a Ground 3D Laser Scanner and an External Digital Camera. *Opt. Precis. Eng.* **2016**, *24*, 2158–2166. [CrossRef]
18. Cao, M.W.; Qian, Y.Q.; Wang, B.; Wang, X.; Yu, X.Y. Joint Calibration of Panoramic Camera and LiDAR Based on Supervised Learning. *arXiv* **2018**, arXiv:1709.029261.
19. Yoo, J.H.; Kim, Y.; Kim, J.; Choi, J.W. 3D-CVF: Generating Joint Camera and LiDAR Features Using Cross-View Spatial Feature Fusion for 3D Object Detection. *arXiv* **2020**, arXiv:2004.12636v2.
20. Shahian Jahromi, B.; Tulabandhula, T.; Cetin, S. Real-Time Hybrid Multi-Sensor Fusion Framework for Perception in Autonomous Vehicles. *Sensors* **2019**, *19*, 4357. [CrossRef]
21. Wu, Q.; Li, X.; Wang, K.; Bilal, H. Regional feature fusion for on-road detection of objects using camera and 3D-LiDAR in high-speed autonomous vehicles. *Soft Comput.* **2023**, *27*, 18195–18213. [CrossRef]
22. Arikumar, K.S.; Deepak Kumar, A.; Gadekallu, T.R.; Prathiba, S.B.; Tamilarasi, K. Real-Time 3D Object Detection and Clas-sification in Autonomous Driving Environment Using 3D LiDAR and Camera Sensors. *Electronics* **2022**, *11*, 4203. [CrossRef]
23. Chen, X.; Ma, H.; Wan, J.; Li, B.; Xia, T. Multi-view 3d object detection network for autonomous driving. In Proceedings of the CVPR, Honolulu, HI, USA, 21–26 July 2017.

24. Kim, J.; Kim, J.; Cho, J. An advanced object classification strategy using YOLO through camera and LiDAR sensor fusion. In Proceedings of the 2019 13th International Conference on Signal Processing and Communication Systems (ICSPCS), Gold Coast, Australia, 16–18 December 2019.
25. Wang, Y.; Liu, X.; Zhao, Q.; He, H.; Yao, Z. Target Detection for Construction Machinery Based on Deep Learning and Mul-ti-source Data Fusion. *IEEE Sens. J.* **2023**, *23*, 11070–11081. [CrossRef]
26. Xu, D.; Anguelov, D.; Jain, A. PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation. *arXiv* **2018**, arXiv:1711.10871v2.
27. Wang, J.; Liu, F. Temporal evidence combination method for multi-sensor target recognition based on DS theory and IFS. *J. Syst. Eng. Electron.* **2017**, *28*, 1114–1125. [CrossRef]
28. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and real time tracking. In Proceedings of the 2016 International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468.
29. Wojke, N.; Bewley, A.; Paulus, D. Simple online and real time tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3645–3649.
30. Wang, X.; Fu, C.; Li, Z.; Lai, Y.; He, J. DeepFusionMOT: A 3D multi-object tracking framework based on camera-LiDAR fusion with deep association. *IEEE Robot. Autom. Lett.* **2022**, *7*, 8260–8267. [CrossRef]
31. Wang, L.; Zhang, X.; Qin, W.; Li, X.; Gao, J.; Yang, L.; Liu, H. Camo-mot: Combined appearance-motion optimization for 3d multi-object tracking with camera-lidar fusion. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 11981–11996. [CrossRef]
32. Chen, M.; Ren, Y.; Ou, M. Adaptive Robust Path Tracking Control for Autonomous Vehicles Considering Multi-Dimensional System Uncertainty. *World Electr. Veh. J.* **2023**, *14*, 11. [CrossRef]
33. Hosseinzadeh, M.; Sinopoli, B.; Bobick, A.F. Toward Safe and Efficient Human–Robot Interaction via Behavior-Driven Danger Signaling. *IEEE Trans. Control. Syst. Technol.* **2024**, *32*, 1. [CrossRef]
34. Zhao, J.; Xu, H.; Liu, H.; Wu, J.; Zheng, Y.; Wu, D. Detection and tracking of pedestrians and vehicles using roadside LiDAR sensors. *Transportation Research Part C: Emerg. Technol.* **2019**, *100*, 68–87. [CrossRef]
35. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]