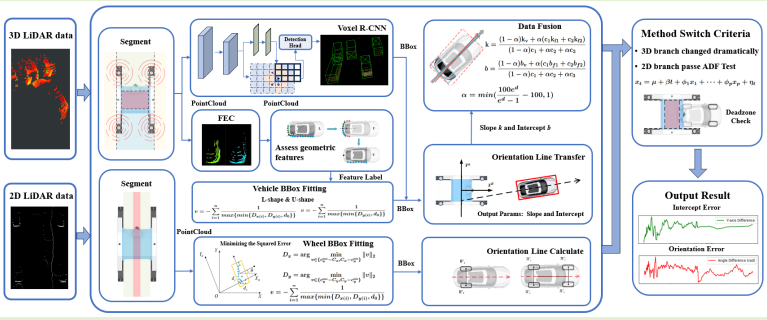
A Hierarchical Target Vehicle Pose Detection Framework in Ro-Ro Terminal Environment

Runjiao Bao, Yongkang Xu, Junfeng Xue, Haoyu Yuan, Lin Zhang, Shoukun Wang

***Abstract*— In light of the urgent need to automate and upgrade roll-on/roll-off (Ro-Ro) transportation methods in ports, many ports have started using transfer Automated Guided Vehicle (AGV) to replace traditional manual operations. Automatic docking is a critical aspect of transportation, which relies heavily on accurately identifying and localizing target vehicles. However, frequent issues such as point cloud feature loss and truncation of point clouds—often due to perception dead zones during the docking process—can significantly hinder existing vehicle recognition methods. This paper introduces a hierarchical vehicle pose detection approach. During the early and mid-stages of docking, we employ two branches for identifying target vehicles to reduce the effects of feature loss and the frequent changes in shape: Voxel R-CNN and an improved search-based optimal bounding box fitting algorithm, which are then combined through result fusion. Additionally, to tackle feature loss caused by point cloud truncation in the late docking phase, we have developed a method to detect wheel poses and calculate the target vehicle pose inversely. Building on this, we established a set of composite evaluation metrics for method switching, ensuring the stability and robustness of the results. Our hierarchical vehicle pose detection method has been successfully implemented in transfer AGVs and applied to port roll-on/roll-off logistics transportation. In datasets collected during actual docking processes, the recognition performance of this framework has surpassed that of the most commonly used 3D object detection methods.**



***Index Terms*— Deep learning, Vehicle detection, Bounding box fitting, Data fusion**

I. Introduction

With the continuous development of the automotive industry, the volume of vehicle imports and exports has steadily increased, and the roll-on/roll-off (Ro-Ro) transportation method at ports urgently requires automation for transformation and upgrading [1], [2]. Currently, vehicle transportation at ports still relies on manual operations, which face numerous issues such as low transfer efficiency, high labor costs, poor working conditions, severe environmental pollution, and difficulties in information management [3]. With the advancement of unmanned systems and intelligent technologies, AGVs are increasingly replacing traditional manual operations [4], [5]. AGVs perform task allocation and path planning through a macro-scheduling system. Upon reaching the rear of the target vehicle, they automatically dock with and lift the vehicle, then transport it to the designated location before lowering it. In this process, automatic docking is undoubtedly a crucial factor directly impacting operational safety, any deviation during the docking process can result in significant economic losses [6]. Accurate identification and localization of the target vehicle are the prerequisites and keys to automatic docking.

Researchers have applied various sensors to accomplish tasks related to target vehicles [7], [8]. Among them, LiDAR sensors are widely used for their suitability for harsh weather conditions, centimeter-level accuracy, and long-range detection capabilities [9]. Methods for vehicle identification based on LiDAR point clouds can be broadly divided into traditional and deep learning methods.

Traditional methods involve segmenting the point cloud and using shape fitting techniques to determine the vehicle's bounding box (BBox). These methods can be divided into feature-based methods and global pose estimation. Feature-based methods rely on edge features to infer the object's pose. Vehicle point clouds are typically classified into L, I, or C shapes [10], with the L shape often used as a reference feature. Kim et al. [11] proposed an iterative endpoint fitting method, where the extreme angle points of a cluster are designated as the endpoints of the baseline. Zhao et al. [12] proposed an edge-based L-shape fitting method, where points are first evaluated to determine whether they are L-shape corner points

This work was supported by the National Natural Science Foundation of China under Grant 62473044, the BIT Research and Innovation Promoting Project under Grant 2024YCXZ007 and 2024YCXY037.
Runjiao Bao, Yongkang Xu, Junfeng Xue, Haoyu Yuan, Lin Zhang and Shoukun Wang are with the School of Automation, Beijing Institute of Technology, Beijing 100081, China (e-mail: 3120230765@bit.edu.cn; yongkang.xu@bit.edu.cn; 3120210867@bit.edu.cn; 3120230769@bit.edu.cn; bit.zhanglin@bit.edu.cn; bitwsk@bit.edu.cn).

Fig. 1. Component structure of the AGV.

or side points of the vehicle. Subsequently, the RANSAC algorithm is used to fit the L-shape feature. Zhang et al. [13] introduced a search-based vehicle pose estimation algorithm that traverses all possible orientations of a rectangle, using criteria such as minimum distance and minimum variance to adapt to different environmental requirements. Compared to feature-based methods, global pose estimation algorithms reduce the reliance on geometric features of point clouds, allowing them to be applied in more complex environments. However, they typically involve high computational costs and complex parameter adjustments. In [14], the authors proposed a heuristic direction-correction BBox fitting method based on convex hulls and line generation. Ding et al. [15] introduced an efficient pose estimation model based on convex hulls, using the minimum occlusion area as the criterion for determining the optimal orientation. He et al. [16] proposed a heuristic rule-based pose estimation method that filters directions based on edge lengths and edge visibility. Global pose estimation methods are susceptible to noise and outliers in the point clouds. Small features outside the vehicle body can significantly affect the vehicle's convex hull modeling, leading to inaccurate results.

Moreover, deep learning-based methods for vehicle detection using LiDAR point clouds have also made significant strides in recent years. Current deep learning-based object detection methods can be broadly classified into three categories: voxel-based, point-based, and pillar-based. A series of methods represented by CenterPoint [17] and PointRCNN [18] use points to represent, detect, and track 3D objects. These methods retain the original precise positional information to the maximum extent. However, point clouds are far more complex than voxels or pillars, so point-based methods are generally less efficient due to the high cost of nearest neighbor search. Second [19] voxelizes the point cloud into a regular grid, making it more suitable for convolutional neural networks (CNNs) before switching to a bird's-eye view (BEV) for object detection. While this approach improves detection speed, it sacrifices some performance. PV-RCNN [20] combines voxel-based CNNs with PointNet [21], fully utilizing the high-quality proposals from voxel CNNs and the flexible receptive field of the PointNet network. Voxel R-CNN [22] proposes that high-precision detection can be achieved using only voxels. This method designs an ROI-Pooling structure and adopts a two-stage approach to combine voxel features of different dimensions for detection, achieving higher detection accuracy than point-based methods while maintaining fast detection. Additionally, pillar-based detectors, represented by PointPillar [23], and PillarNet [24], optimize voxel-based 3D convolutions to use only 2D convolutions, further reducing computational resource consumption. However, their accuracy is relatively lower compared to the other two categories.

However, the detection algorithms proposed by the aforementioned primarily focus on vehicle target recognition under conventional conditions. Automatic docking in Ro-Ro terminals presents particular scenarios. During automatic docking, the AGV is positioned nearly directly behind the vehicle, resulting in an incomplete representation of the vehicle's side geometric features. Additionally, as the AGV approaches the target vehicle, it can obstruct the vehicle's point cloud, leading to the loss of certain features. These factors can significantly affect detection performance, yet the automatic docking process demands exceptionally high recognition accuracy. Ensuring recognition precision in these adverse conditions is a challenging problem.

To address the issues outlined above, we propose a hierarchical vehicle pose detection method that employs two branches: the Voxel R-CNN network and clustering fitting for 3D object detection. Subsequently, we apply a weight-based fusion algorithm to combine different detection results, enhancing the robustness of the detection performance. When preset conditions are met at the end of the docking process, we switch to 2D LiDAR to detect the wheels and inversely calculate the vehicle's pose. To our knowledge, we are the first to design a target vehicle pose detection method specifically for precise vehicle docking, which has been applied in practical production settings. Our contributions are summarized as follows:

- An improved search-based optimal BBox fitting algorithm is proposed. This algorithm enhances the robustness of detection in complex working conditions by first assessing geometric features to select different optimal evaluation metrics. It effectively integrates with deep learning methods to address scenarios involving feature loss.
- To address the situation where the docking AGV obscures the target vehicle, a method for wheel recognition and inverse calculation of vehicle pose using 2D LiDAR is proposed at the end of the docking process. This method effectively fills a gap that conventional object detection methods left in this context.
- A composite evaluation metric is designed to improve detection stability. This metric facilitates switching between two detection methods based on different sensors through a multi-condition decision mechanism, ensuring the smoothness and robustness of the predictive results.

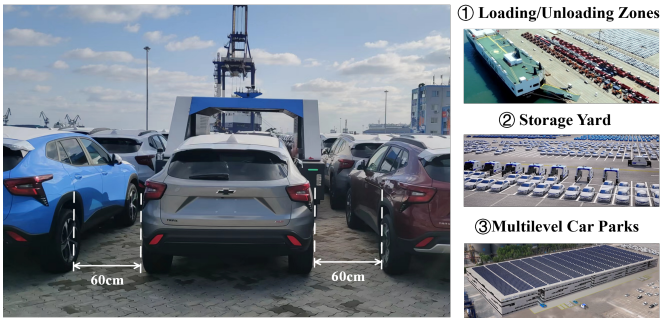


Fig. 2. Actual working environment.

TABLE I
LIST OF IMPORTANT VARIABLES AND DEFINITIONS

Notations	Definitions
k, b	The slope and intercept of the orientation line
$\Delta k, \Delta b$	The deviation of the estimated slope and intercept from the ground truth
(x_i, y_i)	Coordinates of the i-th wheel
d_{th}	The clustering distance threshold
v_1, v_2, v_3	BBox fitting evaluation metric for three methods
τ	Augmented Dickey-Fuller statistic

II. System Platform and Working Requirement

A. AGV Configuration

To meet the demands of navigating and handling vehicles in the complex environment of a port, we have designed a heavy-duty electric vehicle transfer AGV, as shown in Fig. 1. The core of the AGV's control unit is a CPU based on the x86 architecture, equipped with an RTX4060 GPU, and developed using Ubuntu 20.04 and ROS Noetic as the software platform. The AGV features independent drive and steering capabilities for its four wheels. Additionally, it is designed with an integrated lifting mechanism that can automatically adjust the clamp's position based on the axle distance of the target vehicle, accommodating various axle configurations for effective transfer.

To enhance the AGV's ability to capture comprehensive point cloud data of its surroundings and internal area, we abandon the traditional method of mounting 3D LiDAR on the AGV's top. Instead, four 3D LiDAR sensors are positioned at the four corners of the AGV. Each of the four lifting mechanisms has a 2D LiDAR sensor for precise wheel detection. The AGV is also equipped with an RTK system for high-precision localization.

B. Working Environment and Process

Ro-Ro terminals are typically divided into loading and unloading zones, storage yards, and multilevel car parks. The loading and unloading zone is used to park vehicles about to be imported or exported, while the storage yards and multilevel car parks store vehicles that are yet to be shipped or have just arrived. Currently, vehicle transport at Ro-Ro terminals relies on manual driving, associated with high costs, low efficiency, and environmental pollution. Therefore, the transport AGV needs to be able to automatically handle vehicle transfers across multiple regions and locations within the terminal. The working environment of the port is shown in Fig. 2.

In the overall workflow, the AGV cloud scheduling system coordinates the scheduling of multiple transfer robots based on the vehicle transfer tasks defined by the port management system, followed by global path planning across various work areas. The vehicle operation system performs local path planning based on the routes issued by the cloud system, guiding the AGV to the designated pickup area behind the target vehicle. The AGV automatically identifies the target vehicle ahead and initiates the docking process, moving forward along the axis of the target vehicle. When the front 2D LiDAR of the AGV reaches the center position of the target vehicle's front wheels, the AGV halts and uses its lifting mechanism to lift the target vehicle. It then moves to the designated location according to the assigned task and lowers the target vehicle.

C. Detection Requirement

In the workflow, the vehicle docking process directly impacts the safety of transferring vehicles. Any collision between the AGV and the vehicle can lead to significant financial losses. Accurately locating the target vehicle is a prerequisite for precise docking, making it a critical step in the overall process. Traditional 3D vehicle detection often uses 3D BBox to describe the vehicle's position and orientation, but in this task, 3D BBoxes contain redundant information. We simplify the BBox to represent the vehicle's axial orientation line, making it easier for the AGV to follow directly.

First, the BBox is compressed into 2D by removing the Z-axis information. For a standard BBox in the AGV coordinate system, defined as $\{x_i, y_i, w_i, h_i, \theta_i\}$, where x_i and y_i represent the center coordinates, w_i and h_i denote the width and length of the bounding box, respectively, and θ_i is the orientation angle. The slope and intercept of the orientation line can be determined by the following equations:

$$\begin{cases} k = \tan(\theta_i) \\ b = y_i - x_i \cdot \tan(\theta_i). \end{cases} \tag{1}$$

Using the values of k and b, the deviation between the AGV and the target vehicle can be effectively assessed. Besides, minimizing k and b as control objectives facilitates the design of an optimized controller for precise docking.

III. Proposed Detection Framework

Fig. 3 illustrates the overall structure of our proposed hierarchical detection method. First, 3D point clouds are obtained using 3D LiDAR. Two branches, Voxel R-CNN and Fast Euclidean Clustering (FEC) [25], are used for 3D object detection and point cloud clustering, respectively. The Voxel R-CNN branch directly detects the vehicle's pose, while the FEC branch determines the vehicle's pose through BBox fitting. The detection results from both branches are then fused using a weight-based fusion algorithm to determine the target vehicle's orientation line.

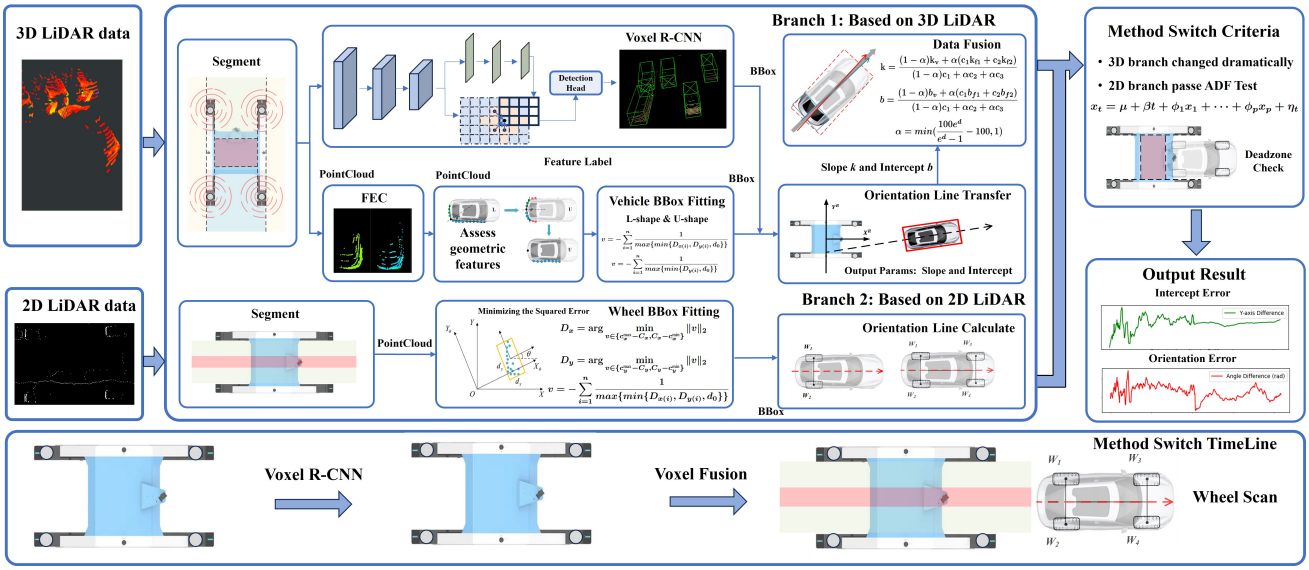


Fig. 3. General process of our proposed framework.

When preset conditions are met, the system switches to 2D LiDAR, using wheel detection to reverse-calculate the vehicle's orientation line. A multi-condition switching mechanism between the two detection methods based on different sensors ensures the smoothness and robustness of the prediction. The specific details are described in the following sections. To better explain our proposed method, we have listed all the important variables in Table I for easier understanding by the readers.

In our study, the following key assumptions were made to ensure the effectiveness and consistency of the docking: first, it is assumed that the experimental environment remains free from extreme weather events, and that all onboard sensors and communication modules of the AGV function reliably throughout the process. Second, the docking area is expected to have a maximum slope of no more than 8°, with the ground surface remaining stable and free from significant structural changes such as settlement or damage. Third, the target vehicle is assumed to be located within approximately 10 meters of the docking path entry point. The vehicle remains stationary during the entire docking operation, with its position and orientation known and fixed, and the surrounding environment is free from dynamic obstacles such as moving vehicles or pedestrians. Finally, it is assumed that a continuous, unobstructed visual detection corridor is maintained between the AGV's initial position and the target vehicle to support reliable perception and localization.

A. Point Cloud Preprocessing

Before processing the point cloud, it is necessary to transform the LiDAR coordinate system into the AGV coordinate system. To achieve a complete point cloud view around the AGV and to prevent the formation of perception dead zones, the 3D point clouds from each direction are stitched together. Subsequently, the Cloth Simulation Filter [26] is applied to the 3D point clouds, filtering without restricting the Z-axis range. This approach maximizes the retention of 3D point clouds

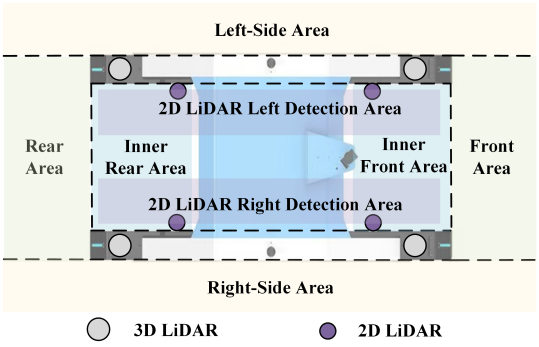


Fig. 4. Segmentation of perception range.

information while eliminating interference from road surface undulations and debris on the ground. For the 2D point clouds, the data from the two 2D LiDARs on the left and right sides are stitched together to detect the wheels on both sides. To avoid interference between the left and right, we only use one side's 2D LiDAR within its detection area and disregard point clouds in regions where wheels cannot possibly appear between the two detection areas.

Since the docking process primarily focuses on the point clouds in the front area and inner area of the AGV, we perform region segmentation on the obtained point clouds of both dimensions. The overall segmentation diagram is shown in Fig. 4.

For the 3D point clouds, we apply voxel filtering to all region point clouds. A voxel grid is created, with the centroid representing all points within each voxel. Voxel filtering offers the fastest filtering speed with slightly lower precision, making it well-suited for rough filtering. We then extract the distance of the nearest point in the side area to the vehicle. Based on this distance, we determine whether to reduce speed; if the distance is below a specified threshold, it indicates the presence of obstacles around the AGV, necessitating a pause in the docking operation for safety reasons. Specifically, only

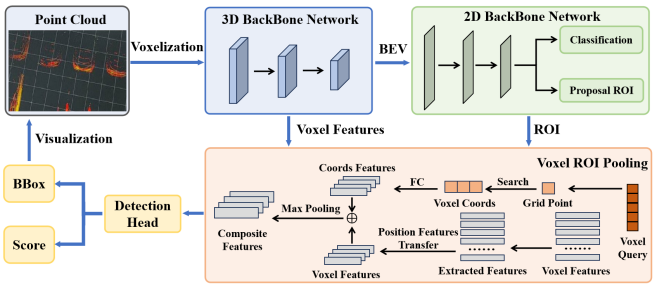


Fig. 5. Overall network structure of Voxel R-CNN.

the 3D point clouds from the inner area and front area are used as the basis for vehicle pose localization.

B. Voxel R-CNN

Voxel R-CNN is a voxel-based two-stage 3D object detection network. Among the three major paradigms for 3D object detection—voxel-based, point-based, and pillar-based—Voxel-RCNN strikes a balance between accuracy and speed, demonstrating excellent applicability in most environments. Its main structure consists of four components: the 3D backbone network, the 2D backbone network, Voxel RoI pooling, and the Detect Head, as illustrated in Fig. 5. The 3D backbone network converts the voxelized input point clouds into features at different scales, reducing the dimensionality of the last layer's 3D features to BEV features. The 2D backbone network performs feature extraction through convolutional layers and a feature fusion subnetwork, producing 3D region proposals along with their corresponding sample classes. Voxel RoI pooling extracts features within the 3D proposal regions, while the Detect Head outputs BBox and confidence information based on the received feature vectors.

Voxel RoI pooling is the key component of this network. After obtaining 3D region proposals, the area is divided into G×G×G sub-voxels. Since 3D feature volumes are very sparse, the center of the sub-voxel is used as the grid center, and nearby voxel features are fused into the grid points. When searching for neighboring voxels, Manhattan distance is used. Voxels with distances below the threshold are marked as neighbors. For the sequence of neighboring voxels $[v_i^1, v_i^2, \ldots, v_i^K]$, the PointNet module is used for feature fusion, with the following formula:

$$\eta_i = \max_{k=1,2,\ldots,K} \Psi([v_i^k - g_i; \phi_i^k]), \tag{2}$$

where $v_i^k - g_i$ represents the relative coordinate between the voxel and the grid point, ϕ_i^k represents the voxel feature of v_i^k, and Ψ represents processing using MLP, with max representing max pooling. The final fused feature vector η_i is obtained. In practice, the voxel feature and relative coordinates are split into two branches. Before searching for the voxel sequence, voxel features are processed through a fully connected layer, followed by a position feature transformation for the corresponding voxel. Finally, the voxel features and relative coordinates are merged. This operation reduces the computational complexity of searching for feature vectors in neighboring voxel sequences, improving algorithm efficiency.

After training, Voxel R-CNN can predict vehicle poses within the regions of interest provided by the point cloud segmentation, specifically targeting incomplete and sparse vehicle point clouds. When the AGV is far from the target, clustering methods may perform poorly, but Voxel R-CNN can estimate the target vehicle's pose effectively.

C. Point Cloud Clustering and BBox Fitting

Apart from the Voxel R-CNN branch, another branch applies radius filtering to the regions of interest provided by point cloud segmentation to remove isolated noise points. Then, Fast Euclidean clustering (FEC) clusters the detected point clouds.

FEC and Euclidean clustering (EC) both calculate the proximity between unclustered points and clustered point sets using the Euclidean distance, which can be expressed as:

$$min(|P_i - P_i'|^2) \geq d_{th}, \tag{3}$$

where P_i and P_i' are two points belonging to different clusters, and d_{th} is the preset clustering distance threshold. Notably, in EC, random points are selected as the basis for a cluster during the clustering process, and the outermost loop iteration unit is the cluster, leading to a significant amount of redundant calculations for the same points. The main improvement in FEC is that the outermost loop iteration unit is the point itself, ensuring that each point is only calculated once.

In this branch, two different point cloud clustering sets are obtained using two different FEC distance thresholds. After FEC, it is necessary to fit the BBox of the extracted target vehicle's point clouds to determine its pose. Since the docking process does not require considering the Z-axis information, the extracted 3D point clouds are first compressed along the Z-axis to reduce it to 2D. Edge extraction is then performed to eliminate interference from the internal point clouds. Under the premise of ensuring optimality, finding the best-fitting rectangle using an optimization-based approach would significantly reduce the algorithm's real-time performance. Therefore, a search-based algorithm is employed to approximate the best rectangle. The basic workflow of this approach is to iterate through all possible orientations of the rectangle, identifying a rectangle that contains all scanned points for each orientation and computes the error based on an evaluation function. After the iteration, the optimal orientation corresponding to the minimum error and its associated fitted rectangle are selected.

To reduce the search range and improve the algorithm's real-time performance, the algorithm's calculations are based on the orientation angle provided by Voxel R-CNN, then searching within a specific positive and negative range. In the scenarios addressed in this paper, the geometric features of vehicle point clouds are primarily U-shaped and L-shaped. At a distance, they may appear as either an L-shape or a U-shape, but as the distance decreases, they will predominantly maintain a U-shape. Different handling and evaluation metrics must be set for point clouds with distinct features to ensure the algorithm's robustness.

First, the geometric features of the point clouds need to be assessed. The closest point to the AGV's local coordinate

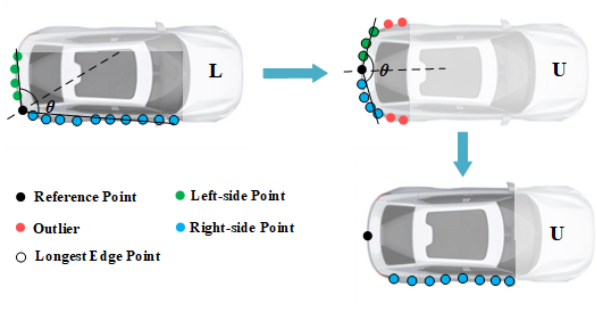


Fig. 6. Typical point clouds geometric features in the docking process.

system is obtained as a reference point. The PROSAC [27] algorithm is then applied to all vehicle edge point clouds. PROSAC can eliminate outliers from the point clouds and provide a point set that forms the longest edge. If the generated point set includes the reference point, a line connecting the reference point and the origin of the AGV's local coordinate system will divide the vehicle edge point clouds into left and right sections. Subsequently, linear fitting is performed on each side, and the angles of the two lines are calculated. To minimize the influence of outliers and provide an overall trend of the edges, PROSAC is used to extract inliers of the line, followed by PCA fitting for line fitting. For L-shaped point clouds, the angles between the two lines will be close to 90 degrees, while the angle for U-shaped point clouds will be significantly greater than 90 degrees. Moreover, suppose the point set generated by PROSAC does not include the reference point. In that case, this indicates that the current situation is characterized by the side length of the U-shaped point clouds being larger than that of the rear edge. Fig. 6 illustrates all possible geometric feature cases of the point clouds.

Standard evaluation metrics for L-shaped features include minimizing the rectangle area, maximizing the proximity of points to the edges, and minimizing the squared error from points to the edges. In the scenarios addressed in this paper, there is no requirement to minimize the rectangle area, and there is a high demand for real-time performance, necessitating a reduction in the computational load of the algorithm. Therefore, the proximity of points to the edges is selected as the evaluation metric v_1. Its formulation is as follows:

$$\begin{cases} D_x = \arg \min\limits_{v \in \{c_x^{\max} - C_x, C_x - c_x^{\min}\}} \|v\|_2 \\ D_y = \arg \min\limits_{v \in \{c_y^{\max} - C_y, C_y - c_y^{\min}\}} \|v\|_2 \\ v_1 = -\sum\limits_{i=1}^{n} \dfrac{1}{\max\{\min\{D_{x(i)}, D_{y(i)}\}, d_0\}}, \end{cases} \tag{4}$$

where $c_x^{\min}, c_x^{\max}, c_y^{\min}, c_y^{\max}$ represent the extreme values of the point set in the BBox's local coordinate system along the X and Y axes, $c_{x(i)}, c_{y(i)}$ denote the local coordinates of points within the set, and C_x, C_y represent the collections formed by these local coordinates. D_x and D_y are the sets formed by the shortest distances. Additionally, a minimum distance threshold d_0 is designed when calculating the minimum proximity to avoid anomalous weights caused by points on or near the edges.

By maximizing v_1, points are encouraged to be distributed closer to the bounding box edges while maintaining robustness against edge artifacts.

Algorithm 1 Clustering and Fitting with Feature Analysis

Input: Point cloud data P, clustering threshold d_{th}, shape discrimination threshold ϵ, prior orientation θ from Voxel R-CNN, search range η

Output: Bounding box b

```
Build a kd-tree structure for P to accelerate queries
P using FEC with threshold d_th, getting clusters P'
Select the cluster P_most containing the most points
Find the reference point P_ref as the point nearest to the AGV's local coordinate system
Apply PROSAC on P_most to extract a point set P_edge forming the longest edge
if P_ref ∈ P_edge then
    Divide P_most into left and right subsets based on the line from P_ref to the origin
    Perform PCA-based line fitting on P_left and P_right to obtain L_left and L_right
    Calculate the angle α between L_left and L_right
    if α ≥ ε then
        for μ = θ − η to θ + η step δ do
            Project P_most onto rotated axes
            Evaluate score v by proximity criterion (Eq. (4))
            Insert q into Q with key (μ)
        end for
    end if
else
    Remove points within 25 cm longitudinal range of P_ref to obtain P_cut
    for μ = θ − η to θ + η step δ do
        Project P_cut onto rotated axes
        Evaluate score v by proximity criterion (Eq. (5))
        Insert q into Q with key (μ)
    end for
end if
Select key (μ_max) from Q with maximum value
Fit the final bounding box b with orientation μ_max
```

For U-shaped features, the points within a 25 cm longitudinal range of the reference point are removed, retaining only the left and right edge points of the U-shaped point clouds. Subsequently, only the Y-axis distances are considered as the evaluation metric v_2, expressed as follows:

$$v_2 = -\sum_{i=1}^{n} \frac{1}{\max\{D_{y(i)}, d_0\}}. \tag{5}$$

Additionally, within this branch, we use two different FEC distance thresholds to obtain two different point cloud clustering sets for BBox fitting. The results obtained with the smaller threshold are more focused on the vehicle's dense point clouds, leading to relatively conservative estimates. In contrast, the larger threshold considers more dispersed points, resulting in relatively aggressive estimates.

In Algorithm 1, we present the full procedure of the clustering and bounding box fitting module. Specifically, the

steps of kd-tree construction, Fast Euclidean clustering, PCA-based line fitting, and PROSAC edge extraction are based on well-established techniques in the literature. In contrast, our contributions lie in the adaptive segmentation strategy according to the presence of the reference point in the edge set, the design of dual evaluation criteria tailored to different edge configurations, and the dynamic search process around the initial orientation estimate provided by Voxel R-CNN. These innovations enhance the robustness and accuracy of the bounding box fitting process under various point cloud conditions.

D. Result Fusion

The Voxel R-CNN branch can provide an effective detection when the target vehicle's point clouds are sparse at a distance. Clustering methods often have low accuracy in these scenarios. Conversely, when the target vehicle's point clouds are closer and the clustering method exhibits greater stability, the Voxel R-CNN branch helps to reduce the search range for BBox angles, improving the algorithm's real-time performance. Based on the complementary characteristics of the two branches, we need to design a reasonable result fusion approach to effectively leverage the strengths of each branch.

Through these two branches, multiple different BBox can be obtained, which can be transformed into multiple orientation lines. To minimize randomness as much as possible, inspired by Weighted Box Fusion (WBF), we propose a weight-based fusion method for the combined orientation lines, as follows:

$$\begin{cases} k = \dfrac{(1-\alpha)k_v + \alpha(c_1 k_{f1} + c_2 k_{f2})}{(1-\alpha)c_1 + \alpha c_2 + \alpha c_3} \\ b = \dfrac{(1-\alpha)b_v + \alpha(c_1 b_{f1} + c_2 b_{f2})}{(1-\alpha)c_1 + \alpha c_2 + \alpha c_3} \\ \alpha = \min\left(\frac{100e^d}{e^d-1} - 100, 1\right), \end{cases} \tag{6}$$

where c_1 and c_2 are the weight coefficients corresponding to the FEC branch with the smaller threshold and the FEC branch result with the larger threshold result, respectively. k_v and b_v represent the slope and intercept of the orientation line from the Voxel R-CNN branch, while k_{f1}, b_{f1}, k_{f2}, and b_{f2} represent the slope and intercept of the orientation lines from the FEC branch's smaller and larger thresholds. d is the X-axis coordinate difference from the origin to the reference point of the target vehicle in the AGV's local coordinate system.

Considering the differing performance of each branch during various stages of docking, we design a distance-based nonlinear weight ratio. At long distances, the estimation leans more toward the results of the Voxel R-CNN branch, while at short distances, it favors the results of the clustering branch. Additionally, within the two different BBoxes obtained from the FEC branch, the BBox corresponding to the smaller FEC distance threshold requires a larger weight due to its stricter detection criteria, i.e., $c_1 \geq c_2$.

E. Wheel Scanning and Detection

Even with the use of multi-branch fusion to analyze point clouds from the 3D LiDAR, occlusion by the target vehicle can still create perception dead zones within the vehicle. During the docking process, important features of the target vehicle may be obscured, and even when the target vehicle is entirely inside the AGV, the point clouds may become fragmented into two separate segments. This fragmentation significantly affects the accuracy of 3D LiDAR detection methods, which is a critical issue when the target vehicle is entering the AGV, as detection accuracy is essential. Therefore, we have opted to use 2D LiDAR to detect the wheels and accurately localize the vehicle's position.

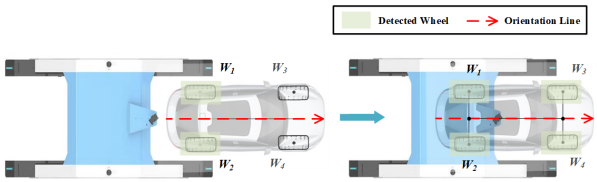


Fig. 7. Schematic of the 2D detection method.

Since the wheels are relatively small and generate few point cloud data points, we use the most accurate metric, the squared point-to-edge error minimization, as the evaluation criterion v_3 when fitting the best BBox. The formula is as follows:

$$\begin{cases} S_1 = \{d_{x(i)} \in D_x \mid D_{x(i)} \leq D_{y(i)}\} \\ S_2 = \{d_{y(i)} \in D_y \mid D_{y(i)} \leq D_{x(i)}\} \\ v_3 = -\text{variance}\{S_1\} - \text{variance}\{S_2\}. \end{cases} \tag{7}$$

As the AGV approaches the target vehicle, the detectable wheel scenario transitions from detecting only the rear wheels to detecting all four wheels. Since the target vehicle's wheels may rotate during transport, after fitting the BBoxes of the wheels, we do not use the orientation of the BBoxes. Instead, we rely solely on the center points of the wheels for detection. The vehicle's position can be inferred based on the center points of the wheels. The schematic is shown in Fig. 7.

Let x_i and y_i denote the coordinates of wheel W_i in the AGV's local coordinate system. When only the rear wheels are detected, the slope k of the orientation line is:

$$k = \frac{x_1 - x_2}{y_2 - y_1}. \tag{8}$$

When all four wheels are detected, the slope k is:

$$k = \frac{(y_3 + y_4) - (y_1 + y_2)}{(x_3 + x_4) - (x_1 + x_2)}. \tag{9}$$

In both cases, the intercept b of the orientation line is:

$$b = \frac{y_1 + y_2}{2} - k \cdot \frac{x_1 + x_2}{2}. \tag{10}$$

F. Switch Condition

The smooth transition between 3D and 2D detection methods directly impacts detection performance in a complex vehicle retrieval operation environment. If the switch occurs too late, the 3D LiDAR point clouds may become occluded; conversely, if it occurs too early, the 2D LiDAR may be too far from the tires, leading to loss of target features and fluctuations in detection results. Therefore, we established a

multi-condition decision mechanism that evaluates the detection results from both branches based on the most recent 100 frames and conducts condition checks. The 3D detection method will switch to the 2D detection method if conditions are met.

A significant jump in the slope or intercept error of the 3D detection method indicates that the AGV is close to the target vehicle, and the 3D LiDAR begins to lose vehicle features, leading to inaccurate detection results. At this point, we should switch to the 2D detection method.

Besides, an Augmented Dickey-Fuller (ADF) test is performed on the 2D detection results. Typically, as the distance decreases, the system error gradually diminishes until it disappears, causing the detection results to fluctuate around the actual value with random error. At this close range, the system's slope and intercept indicators are significantly lower than the previous system error and even lower than the random error, making the time series of detection results more stable. The ADF test can effectively verify the stationarity of the time series. For a p-th order autoregressive (AR) sequence, its general expression can be written as:

$$x_t = \mu + \beta t + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + \eta_t, \tag{11}$$

where x_t is the value of the sequence at time step t, μ is a constant term, β is the coefficient of a linear trend term, $\phi_1, \ldots, \phi_p$ are the autoregressive coefficients, and η_t is a white noise error term. When both μ and β are zero, the sequence is a mean-zero, trendless sequence, corresponding to the case where the system error of the detection values has reduced to a small value. The characteristic equation associated with the AR sequence is:

$$\lambda^p - \phi_1 \lambda^{p-1} - \cdots - \phi_p = 0, \tag{12}$$

where λ represents the characteristic roots. If the sequence is stationary, all roots are within the unit circle in the complex plane. Defining $1 - \phi_1 - \cdots - \phi_p$ as ρ, the null and alternative hypotheses for testing the stationarity of the sequence are formulated as:

$$H_0 : \rho \geq 0 \quad \text{versus} \quad H_1 : \rho < 0, \tag{13}$$

where H_0 indicates that the sequence is non-stationary (contains a unit root), and H_1 indicates that the sequence is stationary. The ADF statistic is constructed as:

$$\tau = \frac{\hat{\rho}}{S(\hat{\rho})}, \tag{14}$$

where $\hat{\rho}$ is the estimated value of ρ and $S(\hat{\rho})$ denotes the standard deviation of $\hat{\rho}$. Subsequently, the critical values for the test statistic are obtained using the Monte Carlo method, and the test results can be derived by comparing the statistics corresponding to the confidence level from the table. The method will be switched if the 2D detection result passes the ADF detection.

Additionally, when the coordinates of the target vehicle's rear wheels leave the segmented detection area inside the AGV or 2D LiDAR detects all four wheels, it is considered that the target vehicle is close enough, at which point the switch to the 2D detection method will also occur.

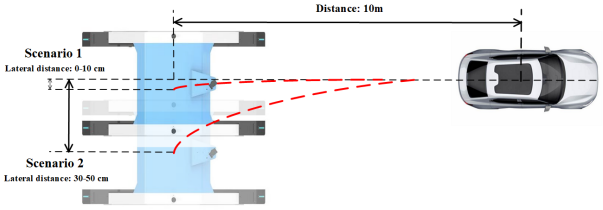


Fig. 8. Experimental scene setup.

IV. Experimental Results and Discussion

A. Experimental Setup

To evaluate the performance metrics of various algorithms in typical docking scenarios, we designed the following experimental setup: at a docking distance of 10 meters, we defined Scene 1 with a lateral distance of 0-10 cm and Scene 2 with a lateral distance of 30-50 cm, representing accurate guidance and guidance with errors, respectively. Both scenes use Vehicle A as the target. To assess the performance of different algorithms on vehicles with varying shapes, we replaced Vehicle A with Vehicle B in Scene 3. Vehicles' parameters are shown as Table II.

TABLE II
Vehicle Parameters

Vehicle Parameter	Vehicle A	Vehicle B
Mass, kg	2045	2415
Width, mm	1890	1940
Length, mm	5030	4915
Wheelbase, mm	3000	2850
Track (F/R), mm	1620	1644

Our experimental scenarios are situated in Yantai Port, the largest international transshipment port for commercial vehicles in China. The vehicle models and scene layouts used in the experiments strictly replicate the actual operational procedures of commercial vehicle transshipment. This ensures that the dataset collected reflects realistic docking conditions and provides a representative environment for evaluating the effectiveness of our proposed method. During the docking process, we employed our proposed method for observation, constructing an MPC (Model Predictive Control) controller based on the obtained slope and intercept for docking control. When switching between 3D and 2D LiDAR stages in our method, the scene is divided into two stages, with the pre-switching stage referred to as S1 and the post-switching stage referred to as S2. S1 corresponds to the typical detection situation, that is, the vehicle is at a medium distance and the point cloud data is basically complete. S2 represents more challenging conditions, where the car body point cloud is truncated to simulate occlusion scenarios. The switching timing between the two steps is shown in Section III-F. We collected point cloud data during docking and verified the algorithm's effectiveness against the actual vehicle positions.

We use RTK to measure the actual pose of the target vehicle. We reuse the parameters k and b as performance evaluation metrics, which were originally introduced in Section 2 to represent the slope and intercept of the target vehicle direction

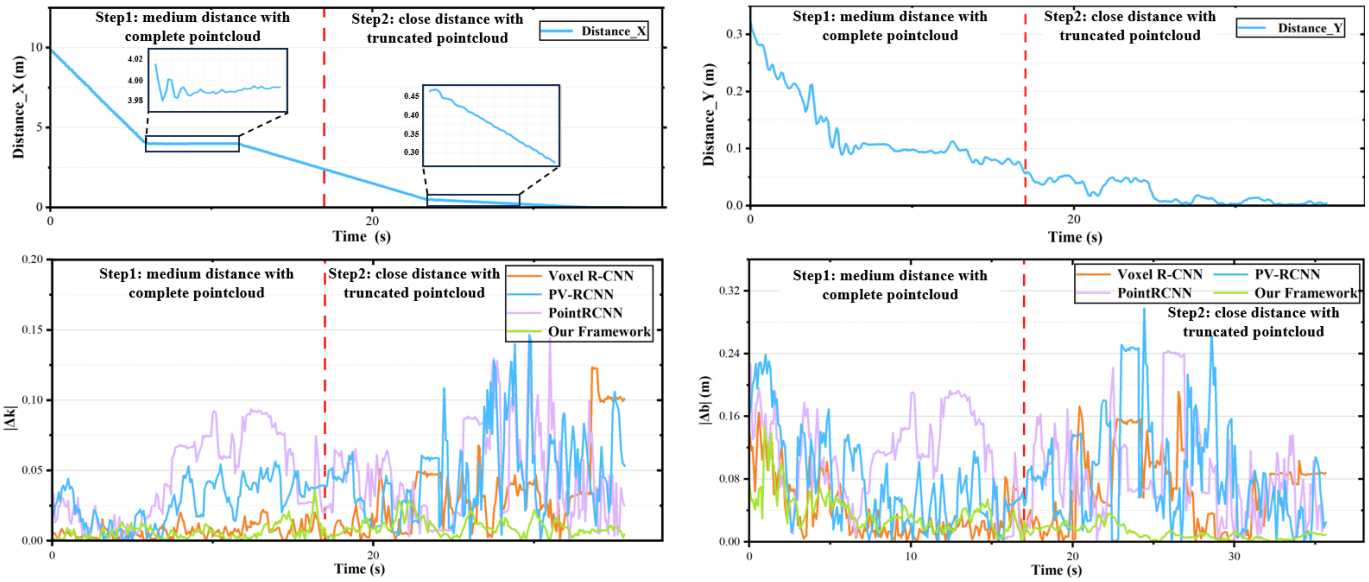


Fig. 9. Comparison Experiment Response.

line relative to the AGV coordinate system, respectively. Furthermore, we define $\Delta k = k_{\text{estimated}} - k_{\text{true}}$ and $\Delta b = b_{\text{estimated}} - b_{\text{true}}$, which represent the deviation of the estimated slope and intercept from the ground truth. These two metrics are used to quantitatively assess the estimation accuracy. Each scene will undergo 50 docking attempts and data collection, with average values calculated for comparison.

Additionally, the deep learning-based 3D point cloud detection models used in the experiment were all trained on the same mixed dataset, consisting of the KITTI dataset [28] and 1000 frames of point cloud data collected during real-world vehicle docking operations at the Yantai Port automobile terminal. The KITTI dataset is a standard benchmark in autonomous driving research, containing 7,481 labeled training samples with sensor data collected from real-world urban and highway scenes, including LiDAR point clouds and annotated 3D bounding boxes. In this study, we focus on the vehicle category from the 3D object detection track to validate the generalization capability of our method in complex outdoor environments. The Yantai Port data accounts for a relatively small proportion of the entire training set and is included primarily to enhance the model's generalization to port-specific scenarios.

We divided the dataset into training, validation, and test sets using an 80-10-10 split. The network was trained end-to-end using the ADAM optimizer for 200 epochs on an NVIDIA L40 GPU, with a learning rate of 0.001. To prevent overfitting, we applied an early stopping mechanism with a patience count of 10, halting training once the validation loss plateaued. Additionally, standard 3D object detection data augmentation techniques were used during training, including random flipping along the X-axis, global scaling with a factor randomly sampled from [0.95, 1.05], and rotation around the Z-axis within $[-\pi/4, \pi/4]$. These strategies collectively enhance the model's robustness and generalization across different environments.

B. Comparison Experiment

In this section, we compare our proposed method with the previously optimal 3D object detection networks. Specifically, we select three widely recognized state-of-the-art methods—Point-RCNN [18], PV-RCNN [20], and Voxel R-CNN [22]—as baselines, as they have demonstrated strong performance on standard 3D detection benchmarks and are commonly used in the literature for evaluating LiDAR-based perception frameworks.

We conducted 50 experiments in three scenarios, averaging the data to evaluate performance metrics. The quantitative comparison results are shown in Table III, which summarizes the end-to-end performance of our proposed method and multiple baseline methods in three different test scenarios, focusing on the overall estimation accuracy under different initial positions and vehicle types.

TABLE III
METRICS OF VARIOUS METHODS

Scenario	Method	Metrics					
		$\lvert\Delta k\rvert$ Mean	$\lvert\Delta k\rvert$ Max	$\lvert\Delta k\rvert$ STD	$\lvert\Delta b\rvert$ Mean	$\lvert\Delta b\rvert$ Max	$\lvert\Delta b\rvert$ STD
Scenario 1	PointRCNN	0.0337	0.1335	0.0277	0.0667	0.2073	0.0551
	PV-RCNN	0.0317	0.1317	0.0222	0.0644	0.2529	0.0539
	Voxel R-CNN	0.0311	0.1238	0.0330	0.0560	0.1902	0.0448
	Our Framework	**0.0085**	**0.0438**	**0.0075**	**0.0215**	**0.1427**	**0.0290**
Scenario 2	PointRCNN	0.0448	0.1441	0.0310	0.0952	0.2437	0.0599
	PV-RCNN	0.0380	0.1464	0.0269	0.0840	0.2974	0.0670
	Voxel R-CNN	0.0217	0.1234	0.0261	0.0532	0.1914	0.0459
	Our Framework	**0.0081**	**0.0361**	**0.0061**	**0.0226**	**0.1455**	**0.0215**
Scenario 3	PointRCNN	0.0329	0.1327	0.0276	0.0676	0.1924	0.0550
	PV-RCNN	0.0303	0.1455	0.0217	0.0652	0.2700	0.0549
	Voxel R-CNN	0.0298	0.1307	0.0314	0.0575	0.1778	0.0409
	Our Framework	**0.0077**	**0.0446**	**0.0077**	**0.0206**	**0.1396**	**0.0246**

Our method outperforms previous approaches across all performance indicators, demonstrating the effectiveness of our proposed hierarchical framework. We visualized one typical result as shown in Fig. 9. Since GPS positioning has certain random errors, the images of X and Y axis distances contain

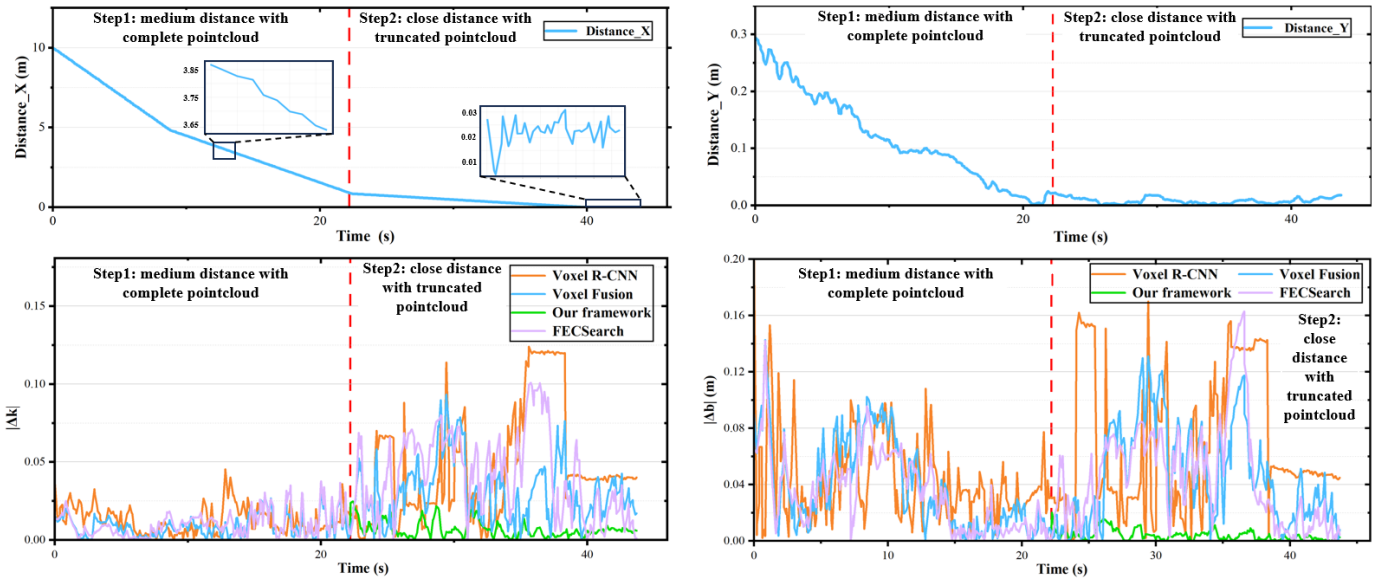


Fig. 10. Ablation Experiment Response.

certain noise. A large error typically occurs at the initial stage of the docking process when the AGV is still far from the target vehicle. At this distance, the deviation in the recognition results manifests as larger Δk and Δb. Our method exhibits a significantly enhanced ability to handle feature loss compared to previous methods. Our approach accurately identifies the target vehicle even at greater distances, where the point clouds are sparser. During this phase, the edge point clouds of the target vehicle in Scenes 1 and 2 exhibit different geometric features, yet our algorithm shows no significant performance degradation, confirming its compatibility with both L-shape and U-shape configurations. Furthermore, at the very end of the docking process, the target vehicle enters a blind zone, resulting in segmentation into two parts, which greatly impacts the performance of both the 3D object detection network and point cloud clustering algorithms. Our method's 2D LiDAR detection phase demonstrates good stability in this context.

These results indicate that our proposed method effectively leverages the strengths of each module, showcasing strong robustness even under challenging conditions of missing point cloud features while ensuring the precision and stability of detection results.

C. Ablation Experiment

We conducted ablation experiments in the scenarios above to verify the effectiveness of each component of our method. We provided various variants to evaluate the contribution of each branch and stage, with the results shown in Table IV. The 3D detection step of our method is referred to as Voxel Fusion, where the branch based on clustering and bounding box fitting is referred to as FECSearch. A typical experimental result is visualized in Fig. 10. Due to the random errors of GPS, the XY distances in Fig. 10 also have some noise. Ablation experiment is mainly used for module-level ablation analysis, so it is only expanded for scenes 1 and 2. The two scenes use the same vehicle model, which is convenient for analyzing the contribution of each module under the premise of controlling variables.

TABLE IV
METRICS OF COMPONENTS OF THE PROPOSED FRAMEWORK

Scenario	Method	Metrics					
		$\|\Delta k\|$ Mean	$\|\Delta k\|$ Max	$\|\Delta k\|$ STD	$\|\Delta b\|$ Mean	$\|\Delta b\|$ Max	$\|\Delta b\|$ STD
Scenario1-S1	Voxel R-CNN	0.0132	0.0453	0.0091	0.0449	0.1692	0.0313
	FECSearch	0.0121	0.0439	0.0094	0.0642	0.1969	0.0448
	Voxel-Fusion	**0.0086**	**0.0298**	**0.0063**	**0.0428**	**0.1427**	**0.0328**
	Our Framework	**0.0086**	**0.0298**	**0.0063**	**0.0428**	**0.1427**	**0.0328**
Scenario1-S2	Voxel R-CNN	0.0450	0.1511	0.0379	0.0647	0.1902	0.0513
	FECSearch	0.0496	0.1388	0.0370	0.0690	0.1715	0.0604
	Voxel-Fusion	0.0291	0.1132	0.0197	0.0467	0.1311	0.0335
	Our Framework	**0.0082**	**0.0439**	**0.0082**	**0.0048**	**0.0248**	**0.0047**
Scenario2-S1	Voxel R-CNN	0.0079	**0.0249**	0.0057	0.0459	0.1714	0.0527
	FECSearch	0.0120	0.0468	0.0089	0.0897	0.1889	0.0510
	Voxel-Fusion	**0.0067**	0.0361	**0.0056**	**0.0357**	**0.1455**	**0.0499**
	Our Framework	**0.0067**	0.0361	**0.0056**	**0.0357**	**0.1455**	**0.0499**
Scenario2-S2	Voxel R-CNN	0.0339	0.1233	0.0297	0.0664	0.1913	0.0449
	FECSearch	0.0358	0.1268	0.0315	0.0510	0.2280	0.0526
	Voxel-Fusion	0.0283	0.0941	0.0193	0.0414	0.1476	0.0344
	Our Framework	**0.0094**	**0.0279**	**0.0116**	**0.0135**	**0.0387**	**0.0146**

In the S1 phase of Scenario 1, integrating Voxel R-CNN with the improved search-based bounding box fitting algorithm demonstrates enhanced robustness in handling feature loss at long distances. It is worth noting that in this phase, Voxel Fusion is equal to our full framework, and both achieve the best overall results.

In the more challenging S2 phase of Scenario 1, where the vehicle body point clouds are truncated—resulting in degraded performance of both Voxel R-CNN and the BBox fitting algorithm—our method still achieves accurate vehicle pose estimation. This robustness is attributed to the wheel detection module based on 2D LiDAR, as evidenced by the significantly lower $|\Delta k|$ and $|\Delta b|$ values reported in Table IV.

A similar trend is observed in Scenario 2, where our full framework consistently outperforms all individual components across the evaluation metrics. These results validate the effectiveness of combining voxel-level fusion with robust bounding box fitting and wheel-based pose estimation in handling

incomplete or noisy input data.

D. Docking Effect Experiment

We designed docking experiments to validate the proposed method's effectiveness in practical docking scenarios. We formulated a multi-objective MPC controller that minimizes K and B as the objective function, conducting 50 docking tests for each scenario. During testing, if the AGV's inner distance to the vehicle is within 10 cm, it is considered a deviation in the docking process. If the distance is within 5 cm, there is a risk of scraping the target vehicle, prompting the AGV to stop, which is recorded as a docking failure. The results of the docking experiments are shown in Table V.

TABLE V
SAFE DOCKING RATE AND SUCCESSFUL DOCKING RATE ACROSS DIFFERENT SCENARIOS AND METHODS

Scenario	Method	Safe Docking Rate	Successful Docking Rate
Scenario1	Point-RCNN	70%	88%
	FECSearch	78%	86%
	Voxel-RCNN	80%	96%
	Our Framework	**100%**	**100%**
Scenario2	Point-RCNN	66%	84%
	FECSearch	72%	86%
	Voxel-RCNN	76%	92%
	Our Framework	**100%**	**100%**

The results indicate that our proposed method demonstrates an excellent success rate in practical docking, effectively addressing various special conditions and complex situations in automated docking. During the final stage of the docking process, existing 3D detection-based methods are more susceptible to perception instability due to point cloud occlusion. This often results in sudden shifts in the estimated vehicle position, leading to large steering adjustments by the AGV that may compromise docking accuracy. In contrast, our method switches to a wheel-based detection strategy using 2D LiDAR in this stage, which offers more stable lateral features under occlusion and ensures smoother control. This design choice significantly improves docking reliability and leads to a consistently higher success rate.

E. Time Performance Experiment

To verify the proposed method's time performance in practical use, we designed a time performance comparison experiment based on the experimental platform described earlier. The processing time for each algorithm was recorded across 50 docking attempts, and the average processing time for each attempt was calculated. The experimental results are shown in Table VI.

As observed from the table, the basic search-based BBox fitting algorithm has a high time complexity, making it unsuitable for real-time detection in practical applications. Voxel R-CNN has the shortest processing time among the deep learning-based 3D object detection methods. Our proposed FECSearch branch, while taking slightly more time than Voxel R-CNN, outperforms the basic search-based fitting algorithm significantly. When both branches run simultaneously, the overall processing time is much lower than the sum of the two branches since one mainly works on the CPU and the other on the GPU. Furthermore, our method of reverse-calculating the vehicle pose based on wheel detection has the shortest processing time, ensuring the real-time performance required for close-range docking detection.

TABLE VI
COMPARISON OF MOTHODS BY TIME

Category	Method Name	Time (s)
DL-based Method	PV RCNN	0.0681
	PointRCNN	0.0457
	Voxel R-CNN	0.0341
Traditional Method	Search	0.1868
	FECSearch	0.0463
	Scan	0.0246
Fusion Method	VoxelFusion	0.0521

V. CONCLUSION

This article presents a target vehicle pose detection framework for automatic docking in Ro-Ro ports. To address the issues of feature loss at long distances during docking and the frequent changes in point cloud shape during the process, we designed a 3D point cloud recognition method that integrates Voxel-RCNN with an improved search-based optimal BBox fitting algorithm. To mitigate the loss of features due to point cloud truncation when the target vehicle enters the vehicle perception dead zone at the end of docking, we developed a method for detecting wheel poses to compute the target vehicle's pose inversely. Furthermore, to avoid abrupt changes in detection results during method transitions, we established a composite decision metric to ensure smooth switching. Extensive experiments on real datasets collected during the docking process demonstrate that the proposed framework exhibits superior detection performance compared to the most widely used 3D object detection methods.

REFERENCES

[1] A. Bhargava, M. Suhaib, and A. S. Singholi, "A review of recent advances, techniques, and control algorithms for automated guided vehicle systems," *J. Braz. Soc. Mech. Sci. Eng.*, vol. 46, no. 7, p. 419, 2024.

[2] C.-C. Lin, C.-T. Tsai, Y.-L. Liu, T.-T. Chang, and Y.-S. Chang, "Security and privacy in 5g-iiot smart factories: Novel approaches, trends, and challenges," *Mobile Netw. Appl.*, vol. 28, no. 3, pp. 1043–1058, 2023.

[3] Ç. Iris and J. S. L. Lam, "A review of energy efficiency in ports: Operational strategies, technologies and energy management systems," *Renew. Sustain. Energy. Rev.*, vol. 112, pp. 170–182, 2019.

[4] Y. Yang, S. He, and S. Sun, "Research on the cooperative scheduling of armgs and agvs in a sea–rail automated container terminal under the rail-in-port model," *J. Mar. Sci. Eng.*, vol. 11, no. 3, p. 557, 2023.

[5] P. Z. Sun, J. You, S. Qiu, E. Q. Wu, P. Xiong, A. Song, H. Zhang, and T. Lu, "Agv-based vehicle transportation in automated container terminals: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 1, pp. 341–356, 2022.

[6] M. M. Ur Rahman, A. Yasmeen, and J. Gross, "Phy layer authentication via drifting oscillators," in *Proc. IEEE Glob. Commun. Conf.*, 2014, pp. 716–721.

[7] Y. Xu, R. Bao, L. Zhang, J. Wang, and W. Shoukun, "Embodied intelligence in ro/ro logistic terminal: autonomous intelligent transportation robot architecture," *Sci. China Inform. Sci.*, vol. 68, pp. 150 210–, 2025.

[8] Y. Wang, B. Yang, R. Hu, M. Liang, and R. Urtasun, "Plumenet: Efficient 3d object detection from stereo images," in *Proc. IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, 2021, pp. 3383–3390.

[9] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proc. IEEE Int. Conf. Comput. Vision*, 2019, pp. 9296–9306.
[10] D. Wittmann, F. Chucholowski, and M. Lienkamp, "Improving lidar data evaluation for object detection and tracking using a priori knowledge and sensorfusion," in *ICINCO. Proc. Int. Conf. Informatics Control, Autom. Rob.*, vol. 1, 2014, pp. 794–801.
[11] D. Kim, K. Jo, M. Lee, and M. Sunwoo, "L-shape model switching-based precise motion tracking of moving vehicles using laser scanners," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 2, pp. 598–612, 2017.
[12] C. Zhao, C. Fu, J. M. Dolan, and J. Wang, "L-shape fitting-based vehicle pose estimation and tracking using 3d-lidar," *IEEE Trans. Intell. Veh.*, vol. 6, no. 4, pp. 787–798, 2021.
[13] X. Zhang, W. Xu, C. Dong, and J. M. Dolan, "Efficient l-shape fitting for vehicle detection using laser scanners," in *IEEE Intell. Veh. Symp. Proc.*, 2017, pp. 54–59.
[14] B. Naujoks and H.-J. Wuensche, "An orientation corrected bounding box fit based on the convex hull under real time constraints," in *IEEE Intell. Veh. Symp. Proc.*, 2018, pp. 1–6.
[15] N. Ding, R. Ming, and B. Wang, "Efficient convex-hull-based vehicle pose estimation method for 3d lidar," *Transp. Res. Rec.*, vol. 2678, no. 12, pp. 1172–1182, 2024.
[16] Y. He, W. Zhang, and M. Yang, "Pose estimation of moving vehicles based on heuristic rules for autonomous driving," in *IEEE Int. Conf. Robot. Biom.*, 2022, pp. 729–734.
[17] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2021, pp. 11 784–11 793.
[18] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 770–779.
[19] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/10/3337
[20] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "Pv-rcnn: Point-voxel feature set abstraction for 3d object detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2020, pp. 10 529–10 538.
[21] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 652–660.
[22] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel r-cnn: Towards high performance voxel-based 3d object detection," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 2, 2021, pp. 1201–1209.
[23] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 12 697–12 705.
[24] G. Shi, R. Li, and C. Ma, "Pillarnet: Real-time and high-performance pillar-based 3d object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 35–52.
[25] Y. Cao, Y. Wang, Y. Xue, H. Zhang, and Y. Lao, "Fec: Fast euclidean clustering for point cloud segmentation," *Drones*, vol. 6, no. 11, p. 325, 2022.
[26] W. Zhang, J. Qi, P. Wan, H. Wang, D. Xie, X. Wang, and G. Yan, "An easy-to-use airborne lidar data filtering method based on cloth simulation," *Remote Sens.*, vol. 8, no. 6, p. 501, 2016.
[27] O. Chum and J. Matas, "Matching with prosac-progressive sample consensus," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, vol. 1, 2005, pp. 220–226.
[28] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Rob. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

Runjiao Bao received the B.Eng. degree from Jilin University, China, in 2023. He is currently pursuing a M.S. degree in Control Science and Engineering as a member of the State Key Laboratory of Intelligent Control and Decision of Complex Systems, School of Automation, Beijing Institute of Technology, China. His research interests include trajectory planning and control.

Yongkang Xu received the M.Sc. degree in mechatronic engineering from Central South University, Changsha, China, in 2021. He is pursuing a Ph.D. degree in control science and engineering as a member of the State Key Laboratory of Intelligent Control and Decision of Complex Systems, School of Automation, Beijing Institute of Technology, China. His interests mainly include environmental awareness, mobile robotics, motion control, neural networks, and autonomous navigation.

Junfeng Xue received the B.S. degree of Automation (Robotic Engineering) from Beijing University of Technology in China in 2021. He is currently a M.S. student at School of Automation, Beijing Institute of Technology, China. His research interests include robot control, wheeled motion control, and vibration isolation control of robot. He has been awarded the Best Conference Paper Finalist of World Robot Conference on Advanced Robotics and Automation in 2022.

Haoyu Yuan received his B.S. degree in mechanical and electronic engineering from Beijing Jiaotong University, Beijing, China, in 2023. He is currently pursuing his M.S. degree in control science and engineering at the Beijing Institute of Technology, Beijing, China. His current research interest is focused on the perception and control of unmanned systems.

Lin Zhang received the M.S. degree in engineering from Yanshan University, Qinhuangdao, China, in 2023. He is pursuing a Ph.D degree in control engineering as a member of the State Key Laboratory of Intelligent Control and Decision of Complex Systems, School of Automation, Beijing Institute of Technology, China. His research interests are the motion and control of special unmanned systems. Email: bit.zhanglin@bit.edu.cn

Shoukun Wang received the B.S., M.S., and a Ph.D. degree in the department of automation, from Beijing Institute of Technology, Beijing, China, in 1999,2002,2004 respectively. He has been teaching at the School of Automation, Beijing Institute of Technology, since 2004. His research interests include sensors, measurement, and electrohy draulic control. He has participated in over 30 scientific research projects since 2001, which mainly belong to measurement and servo control.