Enhanced tree trunk detection for the autonomous field mower via LiDAR-camera fusion in complex environments

Jie Ji[1*], Jianhang Yang[1], Mengling Wang[1], Bohan Zhang[1], Yang Liu[2]
(1. *College of Engineering and Technology, Southwest University, Chongqing 400715, China*;
2. *China Automotive Engineering Research Institute Co., Ltd, Chongqing 401122, China*)

Abstract: The increasingly widespread application of autonomous field mowers in agriculture has significantly heightened the demand for precise and reliable tree trunk detection technologies, particularly in complex and challenging operational environments. To overcome the inherent limitations of single-sensor systems, such as the sparse point cloud resolution in Light Detection and Ranging (LiDAR), photometric sensitivity in camera-based methods, and persistent occlusion interference, this study proposes a multi-sensor fusion framework that integrates data from multi-line LiDAR and a monocular camera for robust tree trunk detection. First, a spatio-temporal calibration framework was developed to ensure accurate alignment of multi-source data. Subsequently, the PointPillars network was utilized for efficient extraction of 3D point cloud features, while an improved You Only Look Once Version 8 Nano (YOLOv8n) model was integrated to enable precise 2D image feature extraction. Additionally, the Complete Intersection over Union (CIoU) fusion strategy was adopted to enable effective cross-modal bounding box matching. Experimental results demonstrate that the proposed fusion approach achieves average positioning errors of 0.0619 m in the horizontal direction and 0.0583 m in the vertical direction, along with a tree trunk detection accuracy of 93.68%. This method effectively resolves the false detection issues typically encountered with traditional point cloud clustering algorithms in complex environments, while also mitigating performance degradation in vision-based detection under complex texture conditions. The proposed framework presents an innovative approach to environment-aware perception for autonomous mowing operations.

Keywords: autonomous field mower, tree trunk detection, multi-sensor fusion, PointPillars network, YOLOv8n
DOI: 10.25165/j.ijabe.20261901.10196

Citation: Ji J, Yang J H, Wang M L, Zhang B H, Liu Y. Enhanced tree trunk detection for the autonomous field mower via LiDAR-camera fusion in complex environments. Int J Agric & Biol Eng, 2026; 19(1): 213–225.

1 Introduction

With the accelerating pace of agricultural modernization, intelligent agricultural machinery is playing an increasingly crucial role in enhancing production efficiency and reducing labor costs. Particularly in modern garden management, intelligent machinery has been extensively employed across multiple domains, including precision spraying[1], intelligent fertilization[2], weed mowing[3], branch pruning[4], pest and disease detection[5], and the monitoring of tree growth conditions[6]. This widespread adoption has significantly improved both operational efficiency and management standards. Among these applications, tree trunk detection—serving as a critical foundational technology for intelligent agricultural machinery—directly influences the level of automation in garden management and the precision of equipment operations. Therefore, the development of efficient, accurate, and reliable trunk detection technology is essential for advancing intelligent garden management systems.

Light Detection and Ranging (LiDAR) and cameras are two commonly employed sensor types in intelligent agricultural machinery, each offering unique advantages in environmental perception. Numerous studies have been carried out by researchers to explore the feasibility of trunk detection using these two sensing modalities. LiDAR is capable of providing high-precision three-dimensional spatial information and is particularly suitable for target localization in complex environmental conditions. Building upon two-dimensional LiDAR, Zhang and Zhou[7] developed a trunk detection algorithm and an adaptive Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm, which significantly reduced the false positive rate in interference-free environments. Bargoti et al.[8] employed a vertically mounted two-dimensional LiDAR for data acquisition in orchard fruit tree applications. However, the scanning and matching processes were computationally demanding and time-intensive. Compared to two-dimensional LiDAR, three-dimensional LiDAR is capable of capturing more comprehensive environmental information, thereby enhancing the accuracy of target detection. Liu et al.[9] employed 3D LiDAR technology for the detection and localization of fruit trees in orchard environments. Using traditional Euclidean clustering, they achieved autonomous navigation based on tree trunk localization. Feng et al.[10] proposed the L1-Tree algorithm, which utilized ground-based LiDAR data to perform identification, detection, and reconstruction of tree trunks. The accuracy of branch parameter estimation reached 95%, and the algorithm's robustness in complex scenarios was improved through an anti-data-missing strategy. Fang[11] introduced a multi-step clustering-based cylindrical surface recognition algorithm, enabling the detection and reconstruction of tree trunks in artificial forests. The reconstructed tree trunk point

Received date: 2025-09-17 **Accepted date:** 2025-12-25
Biographies: Jianhang Yang, MS candidate, research interest: mechanical engineering, Email: yangjianhang2002@163.com; **Mengling Wang**, MS candidate, research interest: mechanical engineering, Email: wangmengling2020@163.com; **Bohan Zhang**, Lecturer, research interest: intelligent vehicles and control, Email: zhangbohan@swu.edu.cn; **Yang Liu**, MS, research interest: intelligent agricultural machinery, Email: 2432676746@qq.com.
***Corresponding author: Jie Ji**, Associate Professor, research interest: intelligent vehicles and agricultural machinery. College of Engineering and Technology, Southwest University, Chongqing 400715, China. Email: jijiess@swu.edu.cn.

clouds demonstrated relatively low average error and minimal error fluctuation.

Compared to LiDAR, camera-captured images provide rich texture and detailed information, thereby enhancing the effectiveness of target feature extraction. Based on camera-driven methodologies, Zhang et al.[12] proposed an improved YOLOv8-based method for precise apple tree trunk recognition, effectively addressing the limitations of low detection accuracy and slow processing speed in existing trunk recognition systems. Liu et al.[13] developed a fast trunk segmentation algorithm that integrates depth and texture features by utilizing Red Green Blue-Depth (RGB-D) camera data to enhance trunk detection performance; however, its effectiveness is notably compromised under weak lighting conditions. Peng et al.[14] enhanced You Only Look Once Version 7 (YOLOv7) to reduce the sensitivity of conventional machine vision systems to environmental disturbances in complex orchard environments, enabling autonomous navigation in orchards through fruit tree trunk root localization. Zhang et al.[15] introduced a multi-object detection algorithm based on Faster Region-based Convolutional Neural Network (Faster R-CNN), which improves both detection accuracy and efficiency by selectively activating feature layers specifically optimized for apple tree trunk detection. Zhao et al.[16] designed an upgraded YOLOv3 with Spatial Pyramid Pooling (YOLOv3-SPP) model integrated with data augmentation techniques, achieving high-precision recognition of specific tree species, with recall rates meeting the requirements of real-world applications.

Although significant advancements have been made in single-sensor technology, its inherent limitations continue to hinder practical applications. LiDAR, due to its sparse linear beams, may result in discontinuous point clouds[9,10], whereas cameras are highly susceptible to variations in lighting conditions and canopy occlusion[13,15]. Therefore, the effective integration of data from these two sensor modalities to achieve more accurate and reliable target detection has become a key research focus in recent years. He et al.[17] proposed a method for rice row identification based on the fusion of machine vision and 2D LiDAR data. They performed target recognition through machine vision and 2D point cloud segmentation, and then spatially fused the results to accomplish centerline fitting of rice rows. Xue et al.[18] introduced an accurate tree trunk detection algorithm that integrates visual camera and 2D LiDAR data. By aligning the regions of interest in both laser and visual data, they achieved effective sensor data fusion, thereby improving the accuracy and robustness of tree trunk detection. Nevertheless, 2D LiDAR suffers from the limitations of acquiring limited environmental information and exhibiting relatively low detection accuracy. Sun et al.[19] proposed a navigation line extraction algorithm based on the fusion of cameras and 3D LiDAR, incorporating discrete factors. They performed target feature extraction using machine vision and 3D point cloud clustering, thereby compensating for the limitation of insufficient data acquisition associated with 2D LiDAR. However, the DBSCAN clustering method employed in this approach is not capable of achieving effective target detection at long distances. Liu et al.[20] introduced a tree trunk detection method based on sensor fusion of LiDAR and cameras. This method applied an adaptive threshold depth map to perform tree trunk point cloud clustering and utilized YOLOv3 for image-based trunk recognition, effectively addressing the distance-related limitations inherent in traditional point cloud clustering techniques. Nevertheless, simple point cloud clustering methods tend to yield relatively lower detection accuracy in complex environments.

In summary, considering the inherent limitations of single-sensor-based trunk detection, this study proposes a novel trunk detection approach that integrates multi-line LiDAR and monocular camera data. The pre-trained PointPillars model is employed for point cloud processing, while an enhanced YOLOv8n model is utilized for image analysis, enabling accurate trunk detection from both sensor modalities. Subsequently, the CIoU method was adopted to fuse the tree trunk detection results based on point clouds and those based on images, respectively, thereby obtaining more stable and reliable tree trunk identification and location information. This fusion strategy not only enhances the accuracy and robustness of trunk detection but also effectively mitigates the limitations associated with individual sensor modalities. Ultimately, the proposed method aims to provide high-precision and low-latency technical support for intelligent garden management systems.

2 Materials and methods

2.1 Framework of the trunk detection system

2.1.1 Components of the hardware system

To enable trunk detection through the sensor fusion of LiDAR and camera data, this study established a dedicated environmental perception system for a self-developed intelligent unmanned weeding platform. The system integrates sensor modules and computing units to ensure accurate data acquisition and improved processing efficiency. Specifically, a USB monocular camera manufactured by Hikvision (Hangzhou, China) was used to capture high-quality environmental image data, while a C16 16-line mechanical LiDAR system produced by LeiShen Intelligent System Co., Ltd. (Shenzhen, China) was employed to obtain high-resolution point cloud data. Furthermore, to meet the requirements of real-time data processing, the NVIDIA Jetson AGX Xavier (NVIDIA Corporation, Santa Clara, CA, USA) was selected as the core computing unit. This embedded platform offers advanced parallel computing capabilities and high energy efficiency, making it well-suited for deploying machine learning models and executing real-time data analysis tasks.

The overall hardware framework design in this study emphasizes the optimization of operational scenarios for individual components and strengthens the collaborative interactions among them. The architecture of the hardware framework, along with the designated installation positions of each component, is illustrated in Figure 1. As shown, the C16 LiDAR is mounted at the center of the top of the equipment to provide optimal 360-degree scanning coverage. Meanwhile, the monocular camera is installed slightly behind and below the LiDAR, oriented horizontally to ensure effective frontal image capture.

2.1.2 Architecture of the software system

To achieve effective fusion detection of LiDAR and visual data, this study proposes a modular software system architecture. The architecture is developed based on the Robot Operating System (ROS) within the Ubuntu 20.04 environment, employing C++ and Python as the primary programming languages, and integrating machine learning and computer vision techniques to ensure both accuracy and real-time performance in data processing.

The software system developed in this study is composed of four core modules: data acquisition, data processing, information fusion, and result output, as illustrated in Figure 2. Specifically, the data acquisition module collects point cloud data and image data of the tree trunk by utilizing a C16 LiDAR and a monocular camera. The data processing module utilizes the NVIDIA Jetson AGX

Xavier computing platform to execute target trunk detection based on the independently acquired point cloud and image data, while also conducting spatial calibration between the LiDAR and the camera. The information fusion module projects the 3D detection box derived from the point cloud onto the 2D detection box of the image by applying the calibrated LiDAR-camera projection matrix, thereby achieving detection box fusion and completing trunk detection. Finally, the result output module delivers the 3D structure and positional information of the detected trunks obtained through the fusion process.

Figure 1 The hardware architecture of an autonomous field mower

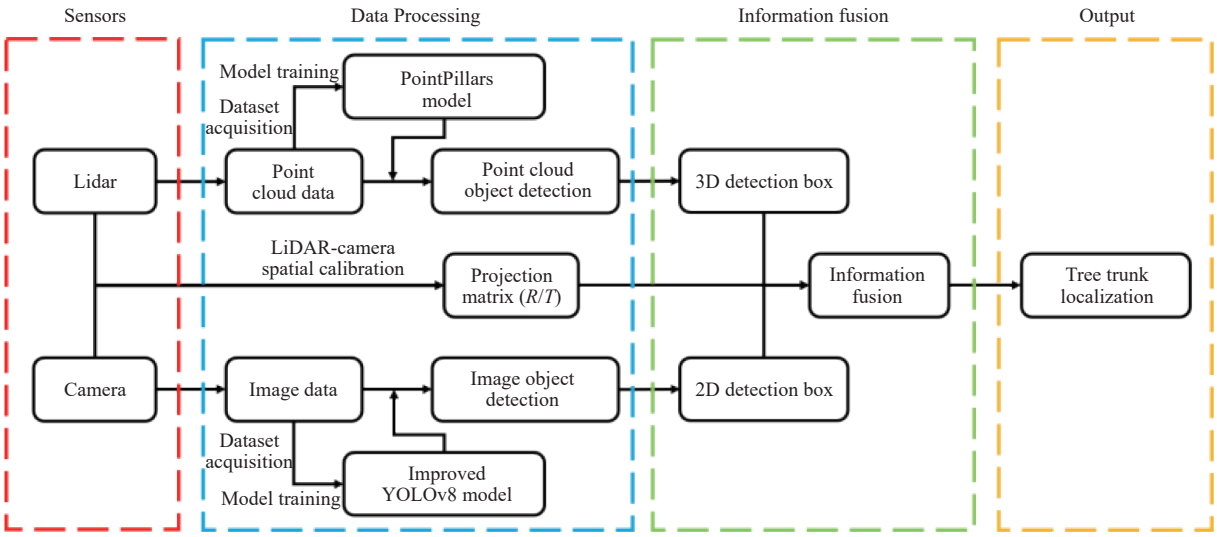


Figure 2 Software architecture for tree trunk detection

2.2 Joint calibration between LiDAR and camera

To address the temporal and spatial misalignments between the C16 LiDAR and the monocular camera during data acquisition, this study conducts a joint calibration of the LiDAR and camera system, thereby ensuring precise fusion of point cloud and image data.

2.2.1 Temporal calibration

Temporal calibration is implemented to resolve data misalignment resulting from inconsistent sampling frequencies across sensors. In this study, the C16 LiDAR operates at a sampling frequency ranging from 5 to 20 Hz, while the monocular camera operates at 60 Hz. As a result, the number of image samples significantly exceeds that of point cloud samples within the same time frame. To mitigate the discrepancy in sampling rates between the LiDAR and the camera, this study employs the time-based nearest neighbor matching method[21], which aligns point cloud data with the corresponding image data that exhibits the smallest timestamp difference. This approach ensures temporal synchronization of data acquisitions between the LiDAR and the camera.

2.2.2 Spatial calibration

Spatial calibration aims to establish the transformation relationship between the LiDAR coordinate system and the camera coordinate system, which comprises rotation and translation matrices. This allows for the precise alignment of point cloud data and image data in three-dimensional space. The spatial relationships among the LiDAR coordinate system, the camera coordinate system, the image coordinate system, and the pixel coordinate system are illustrated in Figure 3. Specifically, the LiDAR coordinate system is denoted as $O_L X_L Y_L Z_L$, the camera coordinate system as $O_C X_C Y_C Z_C$, the image coordinate system as $O_f xy$, and the pixel coordinate system as $O_P uv$.

In this study, the LiDAR coordinate system is defined to serve as the world coordinate system. Let T represent an arbitrary point within this coordinate system. The coordinates of point T in the LiDAR coordinate system are expressed as $T_{LiDAR}=[x_{LiDAR}, y_{LiDAR}, z_{LiDAR}]^T$. Homogeneous coordinates are utilized to describe the positions of relevant points, thereby enabling a unified description of both translation and linear transformations in subsequent

processing stages. Accordingly, the homogeneous coordinate representation of T_{LiDAR} is given by $T_{\text{LiDAR\_h}}=[x_{\text{LiDAR}},y_{\text{LiDAR}},z_{\text{LiDAR}},1]^{\text{T}}$. Similarly, the coordinates of point T in the camera coordinate system are denoted as $T_{\text{Camera}}=[x_{\text{Camera}},y_{\text{Camera}},z_{\text{Camera}}]^{\text{T}}$, with its homogeneous form expressed as $T_{\text{Camera\_h}}=[x_{\text{Camera}},y_{\text{Camera}},z_{\text{Camera}},1]^{\text{T}}$. Additionally, the representation of point T in the pixel coordinate system is denoted as $T_{\text{Pixel}}=[x_{\text{Pixel}},y_{\text{Pixel}}]^{\text{T}}$, and its homogeneous coordinate form is expressed as $T_{\text{Pixel\_h}}=[x_{\text{Pixel}},y_{\text{Pixel}},1]^{\text{T}}$.

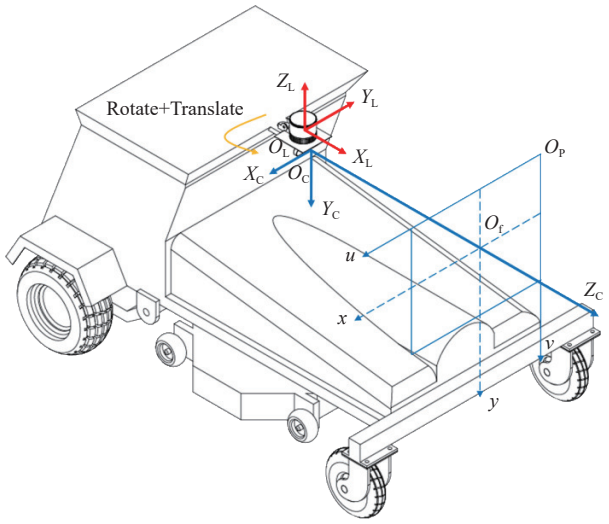


Figure 3 Spatial relationships of the coordinate systems

Furthermore, the projection matrix from the LiDAR coordinate system to the camera coordinate system is represented as $E_{4\times4}$, and its mathematical formulation is given by:

$$E_{4\times4}=\begin{bmatrix}R_{3\times3} & T_{3\times1}\\ 0 & 1\end{bmatrix}=\begin{bmatrix}r_{x1} & r_{x2} & r_{x3} & t_x\\ r_{y1} & r_{y2} & r_{y3} & t_y\\ r_{z1} & r_{z2} & r_{z3} & t_z\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{1}$$

In this equation, $R_{3\times3}$ and $T_{3\times1}$ denote the rotation matrix and translation vector, respectively, which are used to transform coordinates from the LiDAR coordinate system to the camera coordinate system. Based on the transformation relationship between these two coordinate systems, the following equation holds:

$$T_{\text{Camera\_h}}=E_{4\times4}\times T_{\text{LiDAR\_h}} \tag{2}$$

The transformation matrix from the camera coordinate system to the pixel coordinate system is represented by $K_{3\times4}$, which corresponds to the camera's intrinsic parameter matrix. Its mathematical expression is formulated as:

$$K_{3\times4}=\begin{bmatrix}f_x & 0 & v & 0\\ 0 & f_y & u & 0\\ 0 & 0 & 1 & 0\end{bmatrix} \tag{3}$$

Based on the transformation relationship between the camera coordinate system and the pixel coordinate system, the following equation is derived:

$$T_{\text{Pixel\_h}}=\frac{1}{z_{\text{Camera}}}\times K_{3\times4}\times T_{\text{Camera\_h}} \tag{4}$$

In this equation, z_{Camera} represents the perpendicular distance from point T to the camera image plane. The mathematical expression describing the transformation relationship from the LiDAR coordinate system to the pixel coordinate system is therefore formulated as:

$$T_{\text{Pixel\_h}}=\frac{1}{z_{\text{Camera}}}\times K_{3\times4}\times E_{4\times4}\times \text{T}_{\text{LiDAR\_h}} \tag{5}$$

$$\begin{bmatrix}x_{\text{Pixel}}\\ y_{\text{Pixel}}\\ 1\end{bmatrix}=\frac{1}{z_{\text{Camera}}}\times\begin{bmatrix}f_x & 0 & v & 0\\ 0 & f_y & u & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\times\begin{bmatrix}r_{x1} & r_{x2} & r_{x3} & t_x\\ r_{y1} & r_{y2} & r_{y3} & t_y\\ r_{z1} & r_{z2} & r_{z3} & t_z\\ 0 & 0 & 0 & 1\end{bmatrix}\times\begin{bmatrix}x_{\text{LiDAR}}\\ y_{\text{LiDAR}}\\ z_{\text{LiDAR}}\\ 1\end{bmatrix} \tag{6}$$

Based on the aforementioned coordinate system transformation theory, this study employs a checkerboard-based calibration method[22] for spatial calibration. Specifically, multiple checkerboard calibration plates with known dimensions were placed within the experimental environment and simultaneously scanned by the LiDAR and captured by the camera. Subsequently, feature points—such as corner points extracted from both the LiDAR point cloud and the corresponding images—were identified from the acquired data, and the transformation matrix $E_{4\times4}$ between the coordinate systems was computed using these correspondences. Finally, the camera's intrinsic parameter matrix $K_{3\times4}$ and the transformation matrix $E_{4\times4}$ obtained through calibration are summarized in Table 1, and the calibration results are visualized in Figure 4. As demonstrated in the Figure, the LiDAR point cloud data can be accurately projected onto the pixel coordinate system, thereby achieving precise spatial alignment between the LiDAR and the camera.

Table 1 Parameters matrix for spatial calibration

	Intrinsic Camera Matrix $K_{3\times4}$	LiDAR-Camera Projection Matrix $E_{4\times4}$
Parameter Matrix	$\begin{bmatrix}723.274 & 0 & 333.141 & 0\\ 0 & 724.983 & 242.458 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}$	$\begin{bmatrix}0.054 & -0.994 & 0.025 & -0.027\\ 0.021 & -0.051 & -0.927 & 0.037\\ 0.973 & 0.058 & -0.07 & -0.001\\ 0 & 0 & 0 & 1\end{bmatrix}$

Figure 4 Calibration procedure and results of LiDAR and camera

2.3 Tree trunk detection based on point clouds

To achieve accurate identification and localization of tree trunks from the three-dimensional point cloud data acquired by LiDAR, this study developed and implemented a deep learning-based trunk detection framework. This framework consists of five key components: point cloud preprocessing, feature extraction, model training, target recognition, and projection of detection results, all of which are designed to enhance the accuracy and efficiency of trunk detection. The point-cloud-based trunk detection framework is illustrated in Figure 5.

2.3.1 Preprocessing of point clouds

The raw point cloud data acquired by LiDAR is voluminous and structurally complex. A substantial portion of this data consists of ground point clouds, which provide minimal contribution to the effective information required for trunk identification. Direct use of such data in subsequent analyses would substantially increase

computational costs while potentially degrading the accuracy of trunk detection. Therefore, during the point cloud preprocessing stage, efficient filtering of ground point cloud data can significantly improve both computational efficiency and detection accuracy.

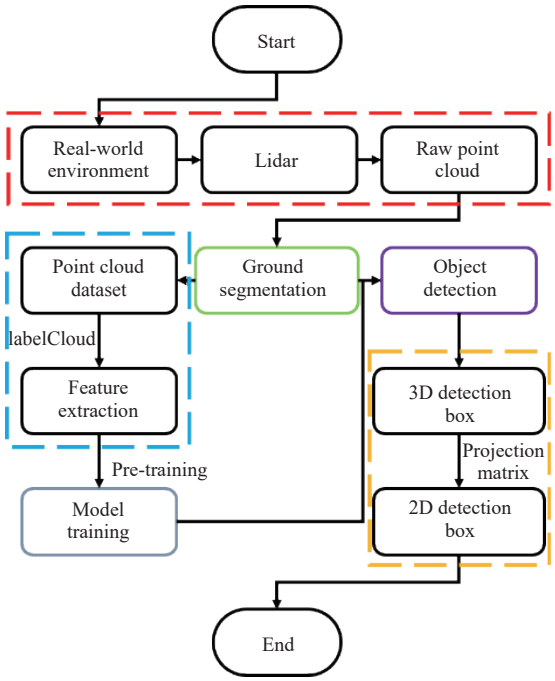


Figure 5 Flowchart for tree trunk detection based on point clouds

To address the challenge of ground point cloud removal, this study proposes a method based on angle thresholds, leveraging the operational principle of multi-line LiDAR. This approach exploits the characteristic that scanning points in the same direction are vertically aligned across all scanning lines during the multi-line LiDAR acquisition process.

In this study, the points on the ground surface intercepted by two vertically adjacent scan lines of the multi-line LiDAR are denoted as point A and point B, with their coordinates expressed as $A(x_0,y_0,z_0)$ and $B(x_1,y_1,z_1)$, as illustrated in Figure 6.

In Figure 6, the vertical height difference h_{AB} between point A and point B is defined as:

$$h_{AB} = |z_0 - z_1| \tag{7}$$

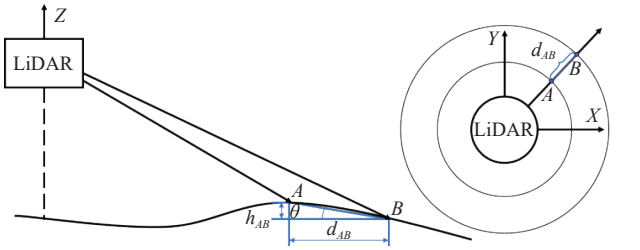


Figure 6 The laser points on the ground from adjacent beams of a multi-line LiDAR

The horizontal distance difference d_{AB} between point A and point B is defined as:

$$d_{AB} = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2} \tag{8}$$

The angle θ formed between the line segment connecting points A and B and the horizontal plane is defined as:

$$\theta = \arctan\left(\frac{h_{AB}}{d_{AB}}\right) = \arctan\left[\frac{|z_0 - z_1|}{\sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}}\right] \tag{9}$$

In an ideal scenario where the LiDAR is perfectly mounted in a horizontal orientation and the ground surface is flat, the theoretical value of the included angle θ should be 0°. However, in practical operating conditions, influence factors such as installation errors, vibrations during system operation, and uneven terrain may cause the included angle θ between ground points to vary within a certain range. According to the topographic analysis of unstructured orchard environments by Guo[23], the ground slope for autonomous mowing operations typically exhibits local undulations. These traversable terrain variations generally result in inter-point angles within a limited range (typically $< 10°$). Therefore, a ground angle threshold of 10° is defined. When $|\theta| < 10°$, points A and B are classified as ground points and are marked for removal. Finally, by traversing the entire point cloud dataset, the ground point cloud can be effectively segmented. The final segmentation results are illustrated in Figure 7.

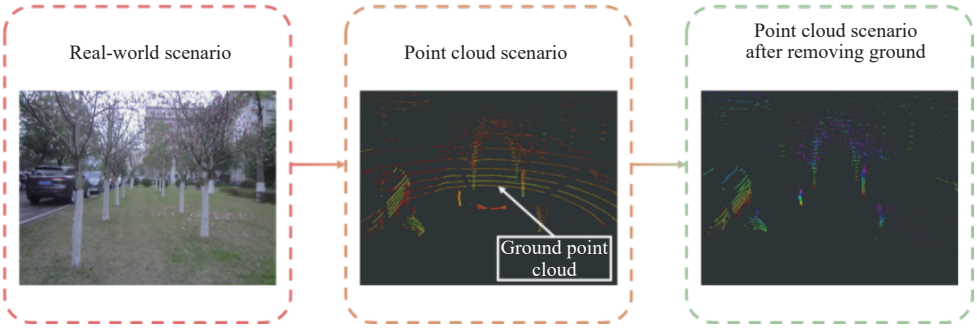


Figure 7 Ground point cloud removal from LiDAR data

2.3.2 PointPillars network

PointPillars[24] is a point cloud object detection network designed based on the principle of data dimensionality reduction. It efficiently extracts geometric features from point cloud data, projects these features into pseudo-two-dimensional (pseudo-2D) images, and subsequently conducts feature extraction, object classification, and bounding box regression on the resulting representations. This method significantly improves both computational efficiency and detection accuracy. In this study, PointPillars is employed for trunk detection, and its network architecture is presented in Figure 8.

As illustrated in Figure 8, PointPillars primarily consists of three core components: the Feature Encoding Network (FEN), the Backbone Network (BN), and the Detection Head (DH). Specifically, in the first stage, FEN transforms the 3D point cloud data into a pseudo-2D image that is compatible with convolution operations. In the second stage, BN applies 2D convolution to the pseudo-2D image in order to extract feature maps. Finally, in the third stage, DH generates object detection results and outputs the corresponding 3D bounding boxes.

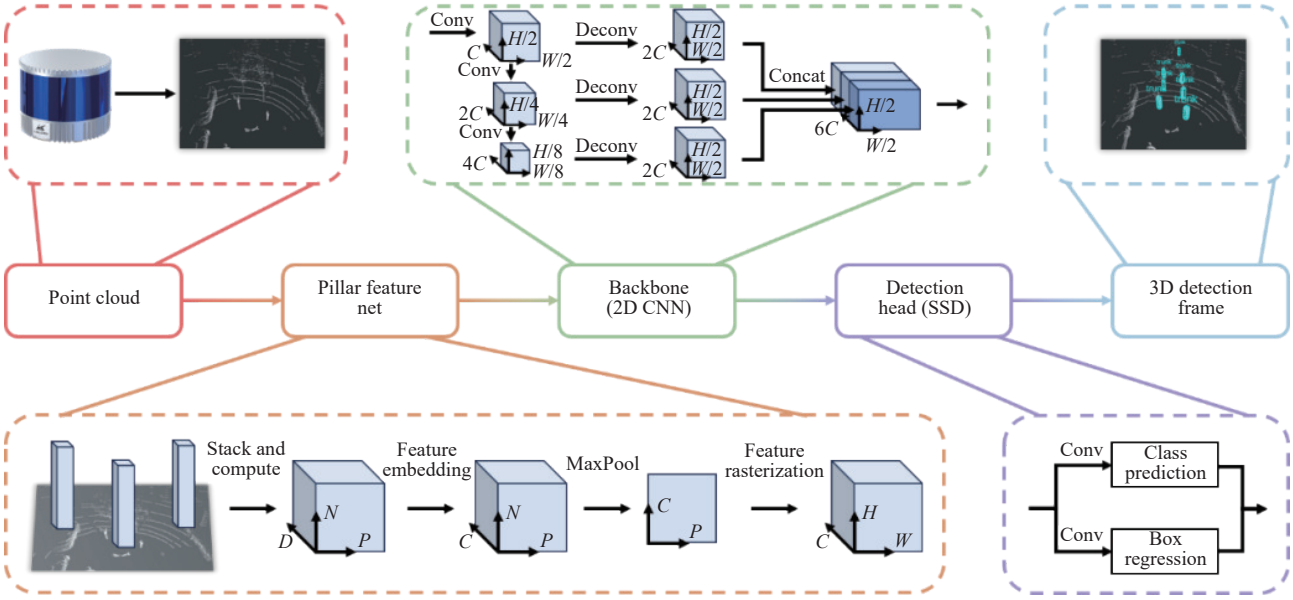


Figure 8 Architecture of the PointPillars network

It is well established that each point in a point cloud contains four parameters: the x, y, z spatial coordinates and the reflectance value r, resulting in a parameter dimension of $D=4$ for the point cloud. In the first stage, the Feature Encoding Network defines spatial dimensions $d \times d$ for P pillars on the XY plane and sets the maximum number N of points per pillar. Based on their x and y coordinates, points are assigned to corresponding pillars. If the number of points exceeds the predefined maximum, random sampling is applied; if it is less than the predefined value, zero padding is used to reach the maximum number of points. Subsequently, for each point within a pillar, FEN computes five additional features: x_c, y_c, z_c, x_p, and y_p (where the subscript c denotes the arithmetic mean of all points within the pillar, and p represents the offset from the center of the pillar in the x and y directions), forming a tensor of size (D,P,N). Next, FEN performs feature embedding for each point, generating a feature vector of length C, resulting in a tensor of size (C,P,N). Then, for each pillar, a max operation is applied across the feature vectors of its constituent points, producing an output tensor of size (C,P). Finally, FEN scatters the extracted features back to their original pillar positions, generating a pseudo-image of size (C,H,W), where H and W denote the height and width of the canvas, respectively.

In the second stage, the Backbone Network first applies top-down convolution to the input pseudo-image of size (C,H,W), generating three sets of feature maps with dimensions $(C,H/2,W/2)$, $(2C,H/4,W/4)$, and $(4C,H/8,W/8)$. Subsequently, BN performs deconvolution on each of these feature maps individually, yielding three upsampled feature maps of size $(2C,H/2,W/2)$. Finally, BN concatenates these three feature maps along the channel dimension, resulting in a single feature map of size $(6C,H/2,W/2)$.

In the third stage, the Detection Head operates in a manner similar to the Single Shot MultiBox Detector (SSD)[25], aiming to directly predict object categories and spatial locations from the feature map. Specifically, DH applies two convolutional operations on the final feature map of size $(6C,H/2,W/2)$ to perform bounding box regression and classification. Subsequently, it matches the prior boxes with ground truth annotations using the 2D Intersection over Union (IoU) metric. Finally, DH transforms the detection results from the 2D image space into 3D world coordinates for bounding box representation.

In summary, PointPillars enables the substitution of 3D convolution with 2D convolution by transforming 3D point clouds into pseudo-2D images. This approach effectively reduces memory consumption, computational complexity, and deployment challenges while maintaining high detection accuracy. Given its strong engineering feasibility, PointPillars is adopted in this study for point cloud-based tree trunk detection.

2.3.3 Training of the PointPillars model

Model training serves as a crucial component in the overall pipeline of tree trunk detection. In this study, a point cloud dataset containing a substantial number of labeled tree trunk instances was utilized to train the PointPillars model. This dataset consists of 1,000 labeled samples, with 800 frames allocated to the training set and 200 frames reserved for the test set. The training platform is conFigd with Ubuntu 20.04 as the operating system, PyTorch 1.2.0 as the deep learning framework, OpenPCDet as the point cloud object detection framework, a six-core Intel® Core™ i7-10750H @ 2.60 GHz CPU, an NVIDIA GeForce RTX 2060 (6 GB) GPU, and 16 GB of DDR4 memory. Following multiple training iterations, the performance results of the PointPillars model are visualized in Figure 9.

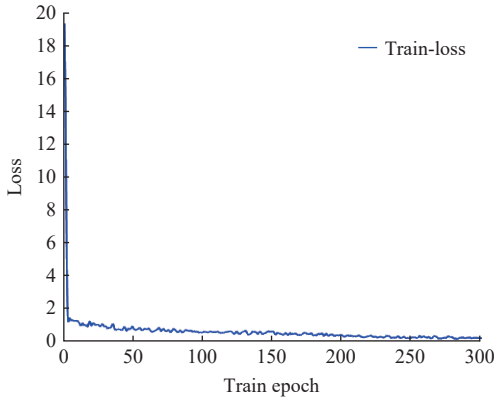


Figure 9 Loss function curve of the PointPillars model training

In Figure 9, as the number of training epochs increases, the value of the loss function (Loss) demonstrates a clear decreasing trend and gradually stabilizes. This indicates that the model progressively enhances its ability to automatically identify and

extract relevant features from the data while effectively reducing prediction errors. Under the condition of IoU=0.7, the average precision (AP) of the bounding box, calculated based on 40 recall levels, achieves 96.0876%, with bird's-eye view average precision reaching 93.1948% and three-dimensional space average precision attaining 84.1758%. In the local test scenario, the outcomes of tree trunk detection using PointPillars are presented in Figure 10. Specifically, Figure 10a shows that the tree trunk point cloud is accurately detected by PointPillars, and the corresponding three-dimensional detection boxes are generated. In Figure 10b, by integrating the LiDAR-camera projection matrix $E_{4\times4}$, obtained through spatial calibration, with the three-dimensional detection boxes of the tree trunk point cloud, the two-dimensional projected detection boxes on the image plane and the pixel coordinates of its center are determined, thereby facilitating the subsequent fusion of tree trunk detection results.

a. Point cloud detection results based on PointPillars

b. Projection of point cloud detection results onto the image plane

Figure 10 Tree trunk detection results using PointPillars

2.4 Tree trunk detection based on images

To achieve accurate detection and localization of tree trunks in images, this study presents an optimized YOLOv8n-based framework for tree trunk detection. The proposed method utilizes deep learning techniques to comprehensively extract and analyze the abundant visual features embedded in the images, thereby enhancing the efficiency and reliability of trunk detection.

2.4.1 The enhanced YOLOv8n framework

The YOLO (You Only Look Once) series is recognized as one of the most widely adopted single-stage object detection frameworks, distinguished by its high processing speed and detection accuracy. Specifically, compared to earlier versions of the YOLO algorithm, YOLOv8[26] demonstrates enhanced processing speed while preserving high detection accuracy, achieved through refinements in network architecture and loss function design, thereby achieving an effective balance between detection precision and computational cost. Among the various model variants within the YOLOv8 family, YOLOv8n (with "n" signifying "nano") represents the most lightweight and compact architecture. It is specifically optimized for rapid inference and minimal computational overhead, while preserving a satisfactory level of detection accuracy. These characteristics render it particularly well-suited for deployment in real-time applications constrained by limited hardware resources. Therefore, this study adopts the YOLOv8n variant as the base model for tree trunk detection, owing to its exceptional real-time performance and high detection accuracy.

The standard YOLOv8n algorithm primarily utilizes conventional convolutional layers and C2f modules, which can lead to degraded performance in practical detection scenarios due to limited computational resources available on mobile devices. To enhance real-time capabilities for subsequent trunk fusion detection while maintaining high detection accuracy in image-based detection, this study incorporates the Ghost Convolution module[27], the C2f Ghost module[28], and the CBAM (Convolutional Block Attention Module) attention mechanism[29] into both the backbone and detection head of the standard YOLOv8n architecture. This architectural enhancement effectively reduces the model's parameter count and computational complexity without compromising detection accuracy. To assess the effectiveness of the proposed enhanced YOLOv8n algorithm, both the original and modified versions were trained and evaluated under identical experimental settings using the same dataset. The results are summarized in Table 2.

Table 2 Comparative experiments of standard and enhanced YOLOv8n

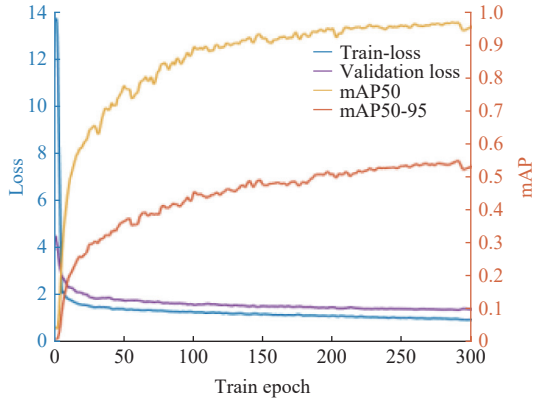
Models	Parameters	GFLOPs	Model Size	mAP50	mAP50-95
Standard YOLOv8n	3 005 843	8.1	6.3MB	0.81	0.54
Enhanced YOLOv8n	1 730 881 (57.6%)	5.1 (63.0%)	3.8MB (60.3%)	0.783 (96.7%)	0.518 (95.9%)

Through comparative experiments, the enhanced YOLOv8n model demonstrates a 42.4% reduction in parameter count, a 37% decrease in computational complexity, a 39.7% reduction in model size, a 3.3% decline in mAP50 (mean average precision at IoU= 0.5), and a 4.1% drop in mAP50-95 (mean average precision with IoU ranging from 0.5 to 0.95) compared to the standard YOLOv8n. These results indicate that the enhanced model significantly reduces computational resource requirements while preserving a high level of detection accuracy, thereby demonstrating strong potential for engineering applications. Consequently, this study adopts the enhanced YOLOv8n for image-based trunk detection.

2.4.2 Training of the enhanced YOLOv8n model

The training phase constitutes a critical step in the development and validation of the enhanced YOLOv8n model for tree trunk detection in image space. In this study, a manually annotated RGB image dataset containing 1000 labeled samples was utilized to train the model. The dataset was divided into 800 images for training and 200 for testing, adhering to the same data partitioning strategy employed in the training of the PointPillars model. The model was implemented using PyTorch 1.8.0 on a Windows 11 system, with the same hardware configuration detailed in Section 2.3.3. Following an iterative training process, the detection performance of the enhanced YOLOv8n architecture was assessed and is presented in Figure 11.

As shown in Figure 11a, after multiple training iterations, the loss curve and average precision (AP) curve collectively demonstrate the model's convergence behavior throughout the training process. With an increasing number of iterations, the training loss gradually decreases and eventually stabilizes at a low value, while the average precision steadily improves and reaches a plateau. This trend indicates that the model successfully learns to extract discriminative features for tree trunk detection. Field testing results show that the average detection accuracy for tree trunks within a 10-meter range reaches 97.2%, with an average frame rate of approximately 90 FPS. These results confirm that the trained model satisfies both the accuracy and real-time performance requirements for tree trunk detection, which are essential objectives of this study on image-based detection. The detection outcomes obtained using the enhanced YOLOv8n in the local test scenario are presented in Figure 11b. Specifically, two-dimensional bounding boxes of detected tree trunks were generated and will be utilized in the subsequent fusion stages of the detection pipeline.



a. Training visualization of the enhanced YOLOv8n

b. Detection outcome in a practical scenario

Figure 11 Training results of the enhanced YOLOv8n model

2.5 Point cloud and image fusion for detection of tree trunks

In the preceding section, this study utilized the trained PointPillars model and an enhanced version of the YOLOv8n model to conduct point cloud-based and image-based tree trunk detection, respectively. As a result, two-dimensional bounding boxes were generated from both LiDAR point clouds and image data. LiDAR-based detection offers precise distance and positional information regarding tree trunks. However, due to the limited number of LiDAR beams, objects with three-dimensional structures similar to those of tree trunks may be incorrectly detected. On the other hand, camera-based detection captures detailed texture features but is highly susceptible to variations in environmental lighting conditions. As noted by Jiang et al.[30], in unstructured orchard environments, variable illumination—such as strong backlighting and uneven shadows—can significantly alter pixel intensity distributions. This degradation often leads to the loss of critical texture details or the generation of visual artifacts. Consequently, in certain lighting scenarios, objects with textures resembling those of tree trunks may be misclassified. To address these limitations, this study proposes a sensor fusion approach that integrates both point cloud and image data, aiming to extract more comprehensive feature representations for improved tree trunk detection.

The positional coordinates and dimensional parameters (i.e., length and width) of tree trunks have been determined using LiDAR-based and image-based methods, respectively, and the two-dimensional detection boxes for both point cloud and image data were extracted on the image plane. To effectively fuse these detection results, a fusion strategy based on CIoU (Complete Intersection over Union) was implemented, following the methodology proposed by Zheng et al.[31]. Specifically, IoU[32] is defined as the ratio of the intersection area to the union area of two bounding boxes, serving as a quantitative measure of their spatial overlap. The calculation formula is expressed as follows:

$$\mathrm{IoU}=\frac{\mathrm{Area}(A\cap B)}{\mathrm{Area}(A\cup B)} \tag{10}$$

where, A and B represent the two arbitrary bounding boxes. Compared to conventional IoU, the CIoU method employed in this study not only accounts for the overlapping area but also integrates the distance between box centers and differences in aspect ratios as additional optimization factors. This comprehensive evaluation framework enables more accurate and robust similarity measurements between detection boxes, thereby enhancing the reliability and effectiveness of fused tree trunk detection.

The mathematical formulation of CIoU is defined as follows:

$$\mathrm{CIoU}=\mathrm{IoU}-\left[\frac{\rho^2(b_{\mathrm{LiDAR}},b_{\mathrm{Camera}})}{c^2}+\alpha v\right] \tag{11}$$

where, $\rho(b_{\mathrm{LiRAR}},b_{\mathrm{Camera}})$ denotes the Euclidean distance between the center points of the LiDAR-based detection box and the camera-based detection box. c represents the diagonal length of the smallest enclosing region that encompasses both the predicted box and the ground truth box. α is defined as the weight coefficient, and its formulation is expressed as follows:

$$\alpha=\frac{v}{(1-\mathrm{IoU})+v} \tag{12}$$

In Equations (11) and (12), v represents the aspect ratio consistency between the LiDAR-based detection box and the camera-based detection box, and its mathematical formulation is defined as follows:

$$v=\frac{4}{\pi^2}\left[\arctan\left(\frac{w_{\mathrm{LiDAR}}}{h_{\mathrm{LiDAR}}}\right)-\arctan\left(\frac{w_{\mathrm{Camera}}}{h_{\mathrm{Camera}}}\right)\right]^2 \tag{13}$$

The parameters involved in the CIoU metric are visualized in Figure 12.

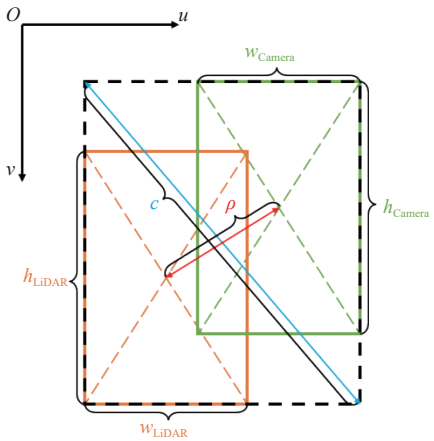


Figure 12 Parameters defined in CIoU method

In summary, this study has demonstrated the necessity of fusion detection, clarified the methodologies for implementing fusion detection, and elaborated on the theoretical foundations underlying this approach. The workflow of the LiDAR-camera fusion algorithm based on CIoU proposed in this study is presented in Figure 13, with the detailed processing steps outlined as follows:

1) Data subscription: Subscribe to the ROS topics associated with the 2D detection boxes derived from both point cloud and image data. Based on the current frame's point cloud detection results, locate the corresponding image detection results through time synchronization.

2) CIoU value calculation: Select a single detection box from the set of 2D image detection boxes and compute its CIoU values in comparison with all the 2D point cloud detection boxes. Determine the point cloud detection box that achieves the maximum CIoU value.

3) Matching determination: Evaluate whether the maximum CIoU value obtained in Step 2 exceeds the predefined CIoU threshold. If it does, the corresponding detection boxes are classified as a matched pair; otherwise, they are designated as unmatched.

4) Information fusion: Utilizing the positional coordinates and dimensional parameters (i.e., width and height) of the matched 2D point cloud and image detection boxes on the image plane, fuse the associated information and publish the resulting fused detection box topic.

5) Iterative processing: Repeat Steps 2-4 until all 2D image detection boxes in the current frame have been processed. Following the completion of all match determinations, advance to the fusion computation for the subsequent frame.

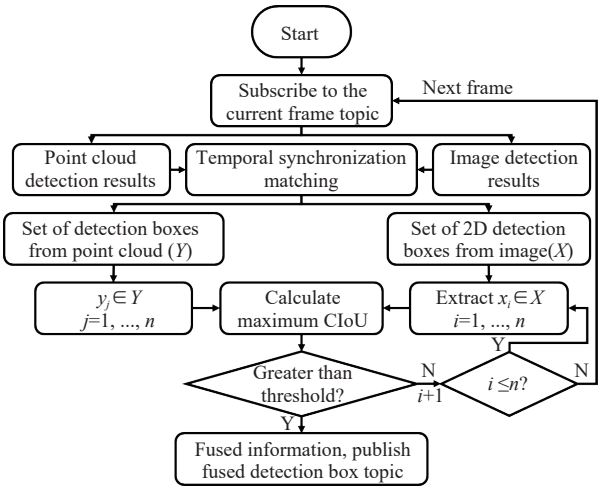
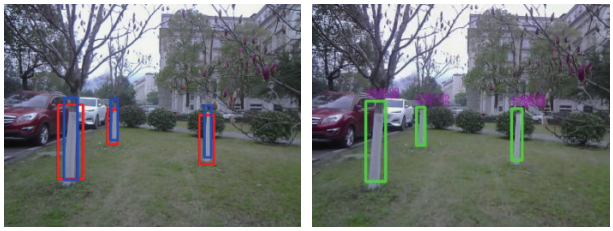


Figure 13 Flowchart of the LiDAR-camera fusion based on CIoU

Based on the aforementioned LiDAR-camera fusion detection method, the CIoU-based fusion results of the 2D detection boxes for point cloud and image data are displayed in Figure 14. In this Figure, the blue box denotes the 2D image detection box obtained using the enhanced YOLOv8n model, the red box represents the 2D point cloud detection box generated by the PointPillars algorithm, and the green box illustrates the fused detection box resulting from the CIoU-based integration of the two aforementioned boxes. The pixel coordinates of each detection box's center are displayed above the corresponding bounding boxes. By integrating the precise positional information from the point cloud with the rich texture features captured by the camera, the green box demonstrates a significant improvement in the accuracy and reliability of tree trunk detection.



a. Detection outcomes of LiDAR and camera b. Detection outcomes of fusion

Figure 14 Diagram of CIoU fusion result

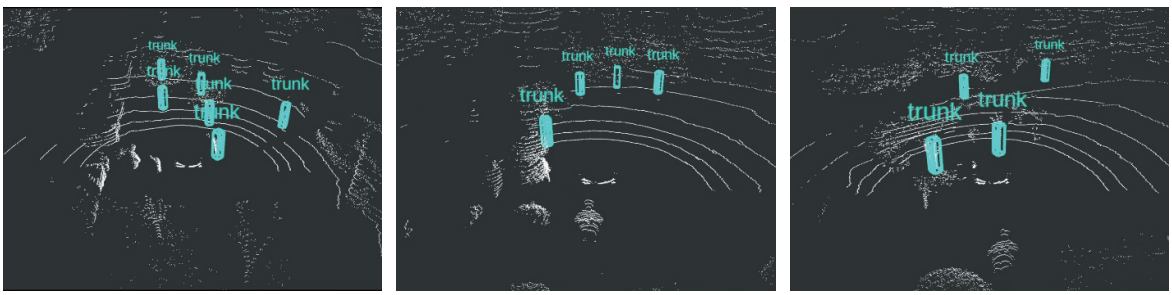
In summary, the tree trunk detection method proposed in this study, which integrates LiDAR and camera data, effectively addresses the limitations and shortcomings of single-sensor systems. By combining positional information from LiDAR with texture features derived from the camera, the detection boxes demonstrate improved accuracy and enhanced practical applicability. Consequently, this research employs a CIoU-based fusion detection approach to achieve superior detection performance.

3 Results and discussion

To evaluate the accuracy and reliability of the trunk detection method proposed in this study, a garden scenario was selected to simulate a garden environment for a controlled field experiment. In this experimental environment, the height of the trees is approximately between 2 and 3 m, the detectable diameter of the tree trunks ranges from 0.15 to 0.25 m, the height of the tree trunks varies between 0.4 and 0.7 m, and the row spacing between trees is about 1 to 2 m. Furthermore, some tree trunks were partially occluded by branches and foliage. Considering the actual requirements of mowing operations, the lawnmower is operated at a speed of 0.4 m/s. Data collection and performance evaluation of the trunk detection method were carried out within these experimental settings. Through repeated trials, the accuracy of the detection results was analyzed to comprehensively assess the effectiveness of the proposed fusion method.

3.1 Trunk detection results using single-sensor and multi-sensor methods

Three representative frames were randomly selected to assess the performance of the proposed detection method. These frames encompassed a variety of lighting conditions, different levels of trunk occlusion, and varying degrees of background complexity, thereby illustrating the robustness and adaptability of the fusion detection method in diverse real-world scenarios. Detailed detection results are presented in Figures 15-18.



a. Frame one b. Frame two c. Frame three

Figure 15 Detection results of trunks based on PointPillars network

a. Frame one b. Frame two c. Frame three

Figure 16 Point cloud trunk detection results projected onto the image

a. Frame one b. Frame two c. Frame three

Figure 17 Trunk detection results based on the enhanced YOLOv8n network

a. Frame one b. Frame two c. Frame three

Figure 18 Detection results based on the fusion of LiDAR and camera

Figure 15 presents the trunk detection results achieved by the PointPillars network based on point cloud data, where the light blue 3D detection boxes indicate the positions of tree trunks identified by the trained model in this particular scene. In addition, the PointPillars network effectively suppresses interference from noise and non-target objects, accurately detects trunk regions within the designated detection area, and generates corresponding 3D bounding boxes within a complex point cloud environment. However, due to the inherent characteristics of LiDAR point cloud data, the network may misclassify cylindrical-shaped objects as trunks, thereby highlighting its limitations under certain geometric conditions, as illustrated in Figure 15c.

In Figure 16, the red 2D detection box represents the projection of the corresponding 3D detection box onto the image plane. Although this method can provide approximate tree trunk position information, the size of the detection box is generally larger than the actual trunk dimensions due to conversion errors between the point cloud and the image coordinate system, which results in reduced localization accuracy.

The image-based trunk detection results obtained using the enhanced YOLOv8n model are shown in Figure 17, where the blue 2D detection boxes represent the trunks detected within this frame. Across three randomly selected frames, the proposed enhanced YOLOv8n model effectively utilizes image texture information to accurately detect tree trunks and generate corresponding 2D bounding boxes. However, due to the presence of complex textures in the outdoor background within the image data, the model may erroneously interpret such background textures as tree trunks. This outcome highlights a fundamental limitation of pure vision-based object detection methods, as illustrated in Figure 17b.

Figure 18 presents the trunk detection results based on the fusion of LiDAR and camera data. The green 2D detection boxes in these three frames represent the tree trunks identified through CIoU-based fusion detection, which integrates both the 3D positional information from LiDAR and the texture features extracted from images. Across three randomly selected scene frames, the proposed fusion-based detection method achieves accurate and robust trunk identification in complex environments, while effectively overcoming the limitations associated with LiDAR-only or camera-only approaches.

3.2 Analysis of the detection accuracy

To quantitatively evaluate the accuracy of the fusion detection method in real-world application scenarios, this study selected trunk coordinate error and target detection accuracy as key evaluation metrics and carried out a comprehensive analysis of these two indicators.

3.2.1 Analysis of the trunk coordinate error

The trunk coordinate error refers to the deviation between the

trunk positions detected using the fusion method and their corresponding ground-truth positions. This metric serves as a critical indicator for assessing the accuracy of trunk detection. Within the LiDAR coordinate system, ten representative tree trunks successfully identified by the fusion method were selected for evaluation. The detected coordinates were obtained using the C16 LiDAR, whereas the actual coordinates were measured using the DELIXI DLX-B2605 laser rangefinder. Subsequently, the coordinate errors in the x-direction $|e_x|$ and y-direction $|e_y|$ between the detected and actual positions were computed and summarized in Table 3.

Table 3 Analysis of trunk coordinate errors

Number	x_{detect}/m	y_{detect}/m	x_{true}/m	y_{true}/m	$\|e_x\|$/m	$\|e_y\|$/m
1	2.345	1.362	2.372	2.349	0.027	0.023
2	3.071	1.216	3.039	3.079	0.032	0.040
3	3.315	1.063	3.352	3.384	0.037	0.032
4	3.985	1.085	4.029	3.992	0.044	0.037
5	4.861	1.142	4.829	4.905	0.032	0.076
6	5.595	1.161	5.641	5.704	0.046	0.063
7	6.711	1.057	6.635	6.571	0.076	0.064
8	7.192	0.829	7.105	7.023	0.087	0.082
9	8.593	1.276	8.692	8.789	0.099	0.097
10	9.611	1.185	9.714	9.609	0.103	0.105

The calculation expressions for $|e_x|$ and $|e_y|$ are formulated as follows:

$$|e_y| = |y_{detect} - y_{true}| \tag{14}$$

$$|e_x| = |x_{detect} - x_{true}| \tag{15}$$

Where, x_{detect} and y_{detect} denote the horizontal and vertical coordinates of the center of the fused detection box within the LiDAR coordinate system, whereas x_{true} and y_{true} represent the corresponding horizontal and vertical coordinates of the actual tree trunk position in the same coordinate framework.

Based on the data recorded in Table 3, the mean errors in the x-direction, denoted as $\overline{e_x}$, and in the y-direction, denoted as $\overline{e_y}$, can be calculated as follows:

$$\overline{e_x} = \frac{1}{n}\sum_{i=1}^{n}|e_{xi}| = \frac{1}{10}\sum_{i=1}^{10}|e_{xi}| = 0.0583 \text{ m} \tag{16}$$

$$\overline{e_y} = \frac{1}{n}\sum_{i=1}^{n}|e_{yi}| = \frac{1}{10}\sum_{i=1}^{10}|e_{yi}| = 0.0619 \text{ m} \tag{17}$$

Based on the measured average errors, the trunk coordinate error curve can be plotted, as illustrated in Figure 19. The experimental results demonstrate that the average error in the x-direction for the selected 10 samples is 0.0583 m, and in the y-direction it is 0.0619 m. These results indicate that the proposed fusion detection method is capable of providing sufficiently accurate trunk position information for autonomous lawn mowers, thereby fulfilling the positioning requirements for weeding operations in garden environments.

3.2.2 Comparison of the target detection precision

The accuracy of target detection serves as a critical metric used to evaluate the effectiveness of a fusion detection algorithm in correctly identifying objects. This metric is generally quantified through the calculation of True Positives (*TP*), False Positives (*FP*), and False Negatives (*FN*). Specifically, *TP* refers to actual tree trunks that are correctly identified; *FP* indicates non-trunk objects that are mistakenly classified as tree trunks; *FN* represents actual tree trunks that are not detected by the algorithm.

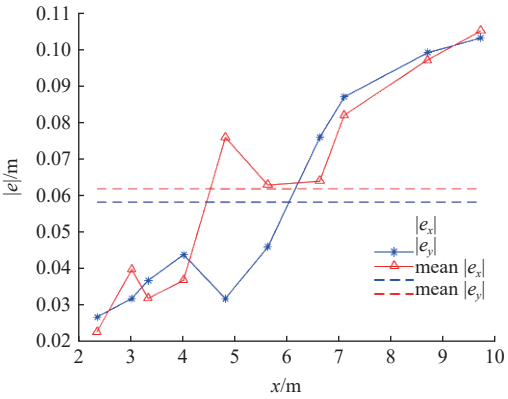


Figure 19 Trunk coordinate error curve

Based on the aforementioned definitions, this study employs three evaluation metrics—Precision (*P*), Recall (*R*), and *F*1 Score—as key indicators for assessing the accuracy of target detection. Specifically, Precision (*P*) denotes the proportion of detected tree trunks that are correctly identified among all detected samples; Recall (*R*) refers to the proportion of correctly identified tree trunks relative to the total number of actual tree trunk instances; the *F*1 Score represents the harmonic mean of Precision and Recall, functioning as a comprehensive measure of the overall performance of the detection method. Higher values of *P*, *R*, and *F*1 are indicative of greater detection accuracy within the scope of this study. The mathematical formulations for these metrics are presented as follows:

$$P = \frac{TP}{TP + FP} \tag{18}$$

$$R = \frac{TP}{TP + FN} \tag{19}$$

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{20}$$

This study randomly selected 50 scene frames and conducted multiple experimental trials. The values of *TP*, *FP*, and *FN* obtained from different detection methods were systematically recorded. The detailed experimental results are summarized in Table 4.

Table 4 Experimental results of target detection accuracy

Detection Methods	*TP*	*FP*	*FN*	Precision	Recall	*F*1 Score
PointPillars	170	21	7	89.01%	96.05%	92.39%
Enhanced YOLOv8n	168	13	9	92.82%	94.92%	93.85%
Euclidean Distance Fusion	156	18	21	89.66%	88.14%	88.89%
Proposed CIoU Fusion	163	11	14	93.68%	92.09%	92.88%

As indicated in Table 4, PointPillars exhibits strong performance in terms of Recall but tends to misclassify cylindrical non-trunk objects in complex environments. Although the enhanced YOLOv8n method achieves the highest *F*1 score (93.85%), its detection reliability is compromised under variable lighting conditions, and it lacks the depth information required for navigation.

To further validate the effectiveness of the proposed strategy, this study compared it with a classic Euclidean Distance Fusion method. As shown in the table, the traditional distance-based method yielded a lower *F*1 score (88.89%). This performance gap is

primarily attributed to the method's reliance solely on center-point distance, which often leads to mismatches when tree trunks are densely clustered or when slight calibration errors occur.

In contrast, the proposed CIoU fusion method, by integrating overlap area, center distance, and aspect ratio constraints, achieves the highest Precision (93.68%). While a slight trade-off in Recall is observed compared to single sensors, the fusion method effectively minimizes false positives (*FP*=11). This indicates superior robustness in addressing challenges such as occlusion and visual mimicry. Most importantly, unlike the image-only approach, the proposed method enables the simultaneous acquisition of accurate 3D positional data and rich textural information, satisfying the operational requirements of autonomous mowers.

4 Conclusions

This study, motivated by the need for accurate tree trunk detection in autonomous mowing applications, proposes a sensor fusion detection approach utilizing data from LiDAR and camera sensors. The main conclusions are summarized as follows: 1) A point cloud preprocessing method based on angle threshold theory is developed to effectively remove ground point clouds, thereby enhancing the efficiency of subsequent point cloud processing. 2) An enhanced YOLOv8n model is developed by incorporating Ghost convolution and the CBAM attention mechanism, retaining 95.9% of the original model's detection accuracy while reducing the model parameters by 57.6%. This modification fulfills the real-time detection requirements of autonomous mowing systems. 3) The proposed CIoU fusion strategy takes into account the overlap degree, center distance, and aspect ratio of detection boxes, thereby enabling the simultaneous acquisition of both category and spatial information of tree trunks. Experimental results demonstrate that the average lateral and longitudinal positioning errors for tree trunks are 0.0583 m and 0.0619 m, respectively, while the accuracy of the fusion detection method reaches 93.68%, satisfying the precision requirements for tree trunk detection and localization.

Experimental verification demonstrates that the proposed detection method exhibits strong robustness in mowing environments by effectively mitigating interference factors such as branch and leaf occlusion in point cloud data, as well as similar texture features in image data. The positioning accuracy satisfies the requirements for autonomous mowing operations. However, a relatively high missed detection rate is observed in highly dynamic operational environments. Future research will focus on optimizing the fusion weights to further enhance detection performance in dynamic scenarios and exploring multi-target tracking algorithms suitable for more complex environments.

Acknowledgments

This work was supported by Key Technology Innovation and Application Development Projects in Chongqing, China (Grant No. cstc2021jscx-gksbX0003).

[References]

[1] Wei P, Yan X J, Yan W T, Sun L, Xu J, Yuan H Z. Precise extraction of targeted apple tree canopy with YOLO-Fi model for advanced UAV spraying plans. Computers and Electronics in Agriculture, 2024; 226: 109425.

[2] Sun J W, Chen Z X, Song R H, Fan S, Han X, Zhang C F, et al. An intelligent self-propelled double-row orchard trenching and fertilizing machine: Modeling, evaluation, and application. Computers and Electronics in Agriculture, 2025; 229: 109818.

[3] Xiang M Q, Gao X M, Wang G, Qi J T, Qu M H, Ma Z Y, et al. An application oriented all-round intelligent weeding machine with enhanced YOLOv5. Biosystems Engineering, 2024; 248: 269–282.

[4] Li Y L, Zhang Z Y, Wang X F, Fu W, Li J B. Automatic reconstruction and modeling of dormant jujube trees using three-view image constraints for intelligent pruning applications. Computers and Electronics in Agriculture, 2023; 212: 108149.

[5] Zhu D J, Xie L Z, Chen B X, Tan J B, Deng R F, Zheng Y Z, et al. Knowledge graph and deep learning based pest detection and identification system for fruit quality. Internet of Things, 2023; 21: 100649.

[6] Cheng X L, Wu X T, Zhu Y F, Zhao Y, Xi B Y, Yan X F, et al. New dielectric-based smart sensor with multi-probe arrays for in-vivo monitoring of trunk water content distribution of a tree in a poplar stand. Computers and Electronics in Agriculture, 2024; 227: 109585.

[7] Zhang Y Y, Zhou J. Laser radar based orchard trunk detection. Journal of China Agricultural University, 2015; 20(5): 249–255. (in Chinese)

[8] Bargoti S, Underwood J P, Nieto J I, Sukkarieh S. A pipeline for trunk detection in trellis structured apple orchards. Journal of Field Robotics, 2015; 32(8): 1075–1094.

[9] Liu W H, He X K, Liu Y J, Wu Z M, Yuan C J, Liu L M, et al. Navigation method between rows for orchard based on 3D LiDAR. Transactions of the Chinese Society of Agricultural Engineering (Transactions of CSAE), 2021; 37(9): 165–174. (in Chinese)

[10] Feng Y H, Su Y J, Wang J T, Yan J B, Qi X T, Maeda E E, et al. L1-Tree: A novel algorithm for constructing 3D tree models and estimating branch architectural traits using terrestrial laser scanning data. Remote Sensing of Environment, 2024; 314: 114390.

[11] Fang H. Large-scale sparse point cloud cylinder recognition and application. Beijing: Beijing Forestry University, Master's dissertation. 2022; 47p. doi: 10.26949/d.cnki.gblyu.2022.000643. (in Chinese)

[12] Zhang H J, Sun Z L, Qi X C, Cao X P, Ren S, Wang J X. Accurate apple tree trunk recognition method based on improved YOLO v8. Transactions of the Chinese Society for Agricultural Machinery, 2024; 55(S1): 246–255, 262. (in Chinese)

[13] Liu H, Zhu S H, Shen Y, Tang J H. Fast segmentation algorithm of tree trunks based on multi-feature fusion. Transactions of the Chinese Society for Agricultural Machinery, 2020; 51(1): 221–229. (in Chinese)

[14] Peng S B, Chen B Q, Li J B, Fan P X, Liu X Y, Fang X, et al. Detection of the navigation line between lines in orchard using improved YOLOv7. Transactions of the Chinese Society of Agricultural Engineering (Transactions of the CSAE), 2023; 39(16): 131–138. (in Chinese)

[15] Zhang J, Karkee M, Zhang Q, Zhang X, Yaqoob M, Fu L S, et al. Multi-class object detection using faster R-CNN and estimation of shaking locations for automated shake-and-catch apple harvesting. Computers and Electronics in Agriculture, 2020; 173: 105384.

[16] Zhao Y H, Yu T, Liu X X. Urban street tree recognition method based on machine vision. 2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP). IEEE: Xi'an, China, 2021; pp.967–971. https://doi.org/10.1109/ICSP51882.2021.9408928.

[17] He J, He J, Luo X W, Li W C, Man Z X, Feng D W. Rice row recognition and navigation control based on multi-sensor fusion. Transactions of the Chinese Society for Agricultural Machinery, 2022; 53(3): 18–26, 137. (in Chinese)

[18] Xue J L, Fan B W, Yan J, Dong S X, Ding Q S. Trunk detection based on laser radar and vision data fusion. Int J Agric & Biol Eng, 2018; 11(6): 20–26. doi: 10.25165/j.ijabe.20181106.3725.

[19] Sun K, Zhang Y F, Gong J L. Multi-sensor data fusion and navigation line extraction method based on discrete factor. Journal of South China Agricultural University, 2022; 43(5): 92–98. (in Chinese)

[20] Liu Y, Ji J, Zhao L J, Feng W, He Q, Wang X K. Trunk detection method based on fusion of LiDAR and camera data. Journal of Southwest University (Natural Science Edition), 2024; 46(2): 183–196. (in Chinese)

[21] Jiang Q, An D, Hang H Y, Liu J H, Guo Y C, Chen L Q, et al. Maize crop row detection algorithm based on fusion of LiDAR and RGB camera. Transactions of the Chinese Society for Agricultural Machinery, 2024; 55: 263–274. (in Chinese)

[22] Feng X, Li J, Yu C S, Qian J Y, He Y. Extrinsic parameter calibration of LiDAR and camera based on edge correlation point cloud. Application Research of Computers, 2023; 40: 2537–2542. (in Chinese)

[23] Guo L M. Systematic design of mountain orchard management machine. Master's dissertation. Luoyang: Henan University of Technology, 2023; 76p. doi: 10.27791/d.cnki.ghegy.2023.000769. (in Chinese)

[24] Lang A H, Vora S, Caesar H, Zhou L, Yang J, Beijbom O. PointPillars:

Fast encoders for object detection from point clouds. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE: Long Beach, CA, USA, 2019; pp.12689–12697. doi: 10.1109/CVPR.2019.01298.

[25] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C Y, et al. Ssd: Single shot multibox detector. Computer Vision-ECCV 2016: 14th European Conference. 2016; pp.21–37. doi: 10.1007/978-3-319-46448-0_2.

[26] Sohan M, Sai Ram T, Rami Reddy C V. A review on Yolov8 and its advancements. International Conference on Data Intelligence and Cognitive Informatics, 2024; pp.529–545. doi: 10.1007/978-981-99-7962-2_39.

[27] Han K, Wang Y H, Tian Q, Guo J Y, Xu C J, Xu C. Ghostnet: More features from cheap operations. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE: Seattle, WA, USA, 2020; pp.1577–1586. doi: 10.1109/CVPR42600.2020.00165

[28] Tang Y H, Han K, Guo J Y, Xu C, Xu C, Wang Y H. GhostNetv2: Enhance cheap operation with long-range attention. Advances in Neural Information Processing Systems, 2022; 35: 9969–9982.

[29] Woo S, Park J, Lee J Y, Kweon I S. CBAM: Convolutional block attention module. Computer Vision-ECCV, 2018; pp.3-19. doi: 10.1007/978-3-030-01234-2.

[30] Jiang A, Noguchi R, Ahamed T. Tree trunk recognition in orchard autonomous operations under different light conditions using a thermal camera and faster R-CNN. Sensors, 2022; 22(5): 2065.

[31] Zheng Z H, Wang P, Liu W, Li J Z, Ye R G, Ren D W. Distance-IoU loss: Faster and better learning for bounding box regression. Proceedings of the AAAI Conference on Artificial Intelligence, 2020; 34(7): 12993–13000.

[32] Yu J H, Jiang Y N, Wang Z Y, Cao Z M, Huang T. Unitbox: An advanced object detection network. Proceedings of the 24th ACM International Conference on Multimedia, 2016; pp.516–520. doi: 10.1145/2964284.2967274.