IET Cyber-Systems and Robotics

WILEY

ZHEJIANG UNIVERSITY PRESS | IET The Institution of Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

An Integrated Perception-to-Navigation Framework for Autonomous Robots in Corn Fields Using RO-YOLO-Based Root–Stalk Detection

Lantao Guo[1] | Jinliang Gong[1] | Yanfei Zhang[2] | Xingda Wang[1] | Mingfeng Wang[3]

[1]School of Mechanical Engineering, Shandong University of Technology, Zibo, China | [2]School of Agricultural Engineering and Food Science, Shandong University of Technology, Zibo, China | [3]Department of Mechanical and Aerospace Engineering, Brunel University of London, London, UK

Correspondence: Jinliang Gong (gjlwing@sdut.edu.cn) | Mingfeng Wang (mingfeng.wang@brunel.ac.uk)

Received: 9 April 2026 | **Revised:** 21 May 2026 | **Accepted:** 10 June 2026

Keywords: deep learning | machine learning | navigation

ABSTRACT

To address the poor performance of navigation line extraction for agricultural robots in complex open-field corn (*Zea mays* L.) environments during the 6–8 leaf growth stage, this study proposes an accurate and efficient navigation baseline extraction method based on an improved YOLO11n model, specifically designed for complex corn field conditions. First, an improved corn root–stalk detection model, termed RO-YOLO, is proposed (YOLO is short for You Only Look Once). By incorporating the EBlock, a low-resolution self-attention module and a multifeature fusion module into the YOLO11n framework, the detection accuracy of the model is significantly enhanced. Next, corn root–stalk positions are precisely localised based on the bounding boxes generated by the model, and representative feature points of corn crop rows are extracted. Subsequently, a feature point screening strategy based on coarse lateral position partitioning and precise longitudinal near-field filtering is proposed to select valid feature points, and the navigation baseline is fitted using the random sample consensus (RANSAC) algorithm. Finally, the navigation line is extracted by solving the angle bisector. Experimental results demonstrate that the improved RO-YOLO achieves a precision, recall and AP@0.5 (average precision at an intersection over union [IoU] threshold of 0.5) of 91.4%, 88.3% and 92.7%, respectively, for corn root–stalk detection. Moreover, the proposed navigation line extraction algorithm for the corn 6–8 leaf stage attains an average fitting time of only 51.6 ms and an accuracy of up to 92.0%, ensuring both high precision and real-time performance of navigation line extraction.

1 | Introduction

The corn 6–8 leaf stage (V6–V8) corresponds to the early pre-jointing period, during which crop sensitivity to herbicides increases, making chemical weed control unsuitable. Meanwhile, as the efficacy of preemergence herbicides gradually declines, weed growth in the field becomes increasingly pronounced. Therefore, this period represents a critical stage for weeding robots to perform mechanical or laser weeding operations. Against the backdrop of the ongoing transition of agricultural machinery towards autonomy and intelligence, intelligent weeding equipment has become an indispensable tool during the corn growth cycle [1]. How to achieve precise and reliable autonomous navigation has become a core technical challenge for intelligent corn weeding machinery and is also one of the current research hotspots in this field. Owing to its strong real-time performance and low cost, visual navigation has become the most widely adopted method for robot navigation worldwide [2]. Accurately and rapidly identifying corn crop rows and extracting navigation lines, enabling precise navigation for agricultural robots, is crucial for realizing autonomous

operations. However, corn field environments are highly complex and involve various uncertainties, such as weed density, illumination variations, growth stages and crop-row discontinuities, which pose significant challenges to the identification and extraction of navigation lines [3]. Consequently, developing a robust and accurate navigation line extraction algorithm for complex maize field environments is of great significance.

In recent years, researchers have conducted extensive studies in the field of visual navigation technology [4]. Traditional corn navigation line extraction algorithms based on machine vision rely primarily on crop features such as texture and shape, combined with image processing techniques to extract key information of crop rows, which is then processed and fitted to generate the navigation line. However, traditional image processing methods are highly susceptible to environmental factors, resulting in suboptimal accuracy of navigation line extraction and consequently affecting autonomous robot navigation. With the rapid development of artificial intelligence, deep learning-based methods for navigation feature extraction have gradually become a focal point of research [5]. Based on the design principles and functional divisions of deep learning models, deep learning-based methods for navigation feature point extraction can be broadly categorised into two types: those based on object detection algorithms and those based on semantic segmentation algorithms. Semantic segmentation algorithms perform pixel-level classification in images. A substantial body of research has focused on applying these algorithms to navigation line extraction, with typical methods including DeepLabv3+ [6], PSPNet [7] and ENet [8]. Adhikari et al. [9] proposed a paddy field crop row detection model based on a semantic segmentation algorithm for autonomous tractor navigation between crop rows. Similarly, Kim et al. [10] applied a segment-based convolutional neural network (CNN) in orchards to perform path area segmentation, and the target path was detected by identifying boundary lines. However, semantic segmentation models are typically large and computationally intensive, making them difficult to deploy on embedded devices. In contrast, object detection models focus more on identifying and localising targets within images and have shown promising potential for crop row and navigation line extraction. Representative algorithms include the You Only Look Once (YOLO) series [11], Fast R-CNN [12] and single shot multibox detector (SSD) [13]. Diao et al. [14] proposed a navigation line extraction method based on an improved YOLOv8s, in which the algorithm first identifies the corn cores, and the navigation line is then fitted using the least squares method based on the detected core positions. Similarly, de Aguiar et al. [15] employed MobileNets, Inception and YOLO algorithms for real-time detection of grape trunks, providing a foundation for subsequent autonomous field navigation. This study focuses on corn at the 6–8 leaf stage in complex field environments and aims to address the low accuracy and poor real-time performance of current navigation line extraction algorithms. A navigation line extraction method based on corn root–stalk recognition using RO-YOLO is proposed, with the following steps: first, the RO-YOLO model is trained using annotated data to obtain its optimal weight file. The optimal weights are then loaded into the detection network to perform object detection on field images, obtaining bounding box information of the crop targets. Based on the detected bounding boxes, the bottom centre points of the crops are extracted as navigation reference feature points. To reduce the interference of outlier points, the extracted feature points are further screened. Finally, the navigation baseline is fitted based on the selected feature points, and the centre line is calculated. Figure 1 illustrates the detailed flowchart of the corn row navigation line extraction algorithm. The navigation line extraction algorithm proposed in this study integrates deep learning with traditional machine learning techniques. Its core advantage lies in combining the robustness of deep learning-based object detection with the high-precision feature point localisation and fitting methods of conventional approaches, thereby enabling efficient and accurate extraction of navigation lines between corn rows.

2 | Materials and Methods

2.1 | Dataset Construction

This study addresses the task of navigation line extraction for the 6–8 leaf stage by constructing a dedicated corn root–stalk dataset. The dataset images are captured in the experimental fields of Linzi District, Zibo City, Shandong Province, China. The summer corn has a plant spacing of approximately 30 cm and a row spacing of about 60 cm. Image acquisition is performed using an RMONCAM Linboshi-G600 monocular RGB camera with a maximum resolution of 1920 × 1080 pixels. A total of 3000 images of corn at the 6–8 leaf stage under varying lighting conditions are collected, which are dedicated to model training. All images are uniformly annotated using LabelImg and subsequently partitioned into three mutually exclusive subsets in a 7:2:1 ratio for training the RO-YOLO model: a training set (2100 images), a validation set (600 images) and a test set (300 images). To validate the performance of the navigation line extraction algorithm, an additional independent test set comprising 300 images was collected under identical field conditions. This set is entirely separate from the aforementioned data and is dedicated to the final performance evaluation and comparative analysis of the navigation line extraction task. The methodology of this study focuses on utilising localised images of the corn root–stalk zone for navigation line extraction. Consequently, image acquisition is performed only on the specific local areas corresponding to the root–stalk positions.

2.2 | Model Development

YOLO is a single-stage object detection model that formulates bounding box prediction as a regression task, directly outputting both bounding box coordinates and class predictions from the input image, thereby exhibiting exceptional efficiency [16]. YOLO11, released by Ultralytics, is an object detection algorithm that incorporates significant architectural enhancements over its predecessor, YOLOv8. A core advancement is the novel introduction of feature extraction modules such as C3k2 and C2 position-sensitive attention (C2PSA), contributing to marked improvements in key metrics, including training speed, inference performance and detection accuracy. According to its network width and depth, YOLO11 is offered in five distinct variants. To facilitate deployment on resource-constrained edge devices, the version with the smallest parameter count and computational cost, YOLO11n, is selected as the baseline model in this study. To enhance the model's adaptability within 6–8 leaf stage corn field environments and further improve detection accuracy for corn root–stalk regions, this study primarily

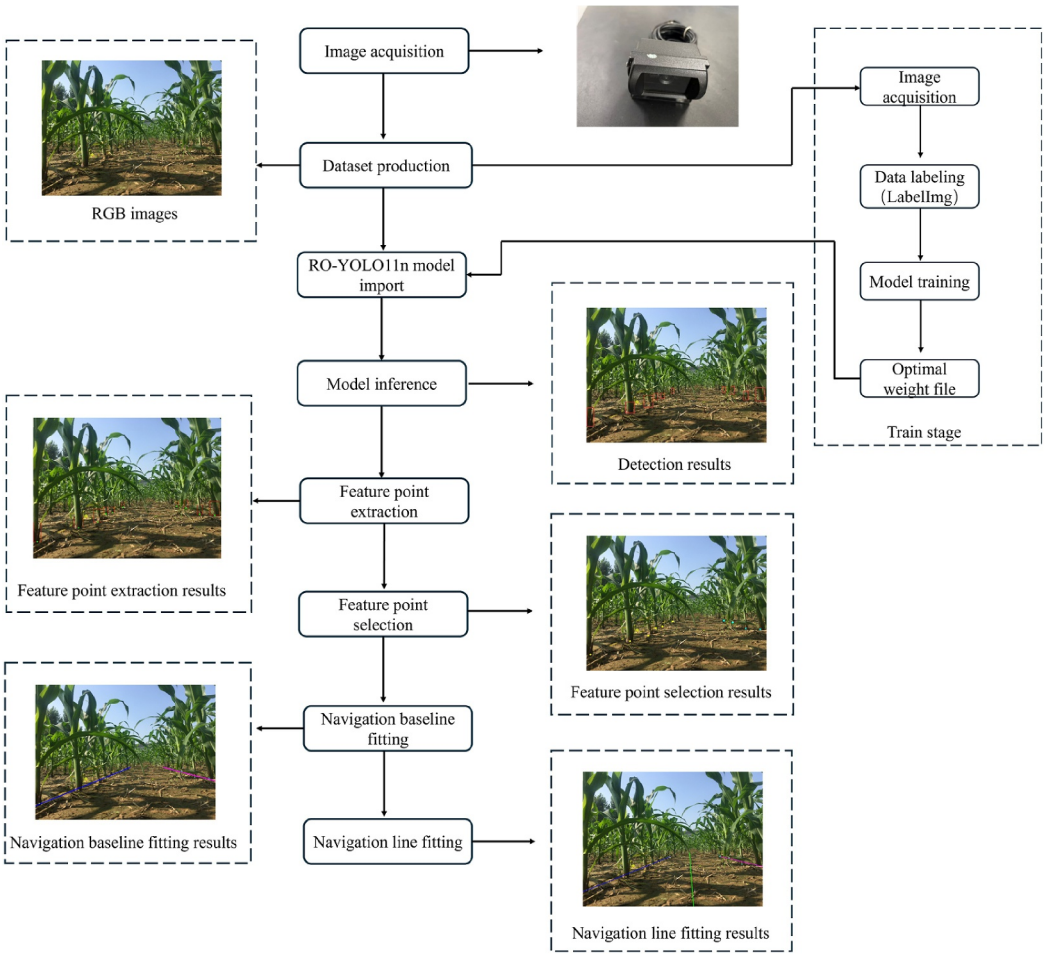

FIGURE 1 | Flowchart of the navigation line extraction algorithm.

introduces improvements in the following three aspects. First, based on the C3k2 module, the EBlock [17] is introduced to redesign the Bottleneck structure, proposing a C3k2-EBlock module to enhance the model's feature representation capability. Second, the attention module in C2PSA is replaced with a low-resolution self-attention (LRSA) [18] module, which effectively enhances the model's robustness to target scale variations. Finally, as the feature concatenation module in the model's neck network fails to highlight critical information sufficiently, it is replaced with a modulation fusion module (MFM) [19], enabling the model to capture more refined local features and contextual dependencies. The network architecture of the proposed RO-YOLO model is illustrated in Figure 2.

2.2.1 | *Enhancement of the C3k2 Module*

The original Bottleneck module within the C3k2 structure employs fixed-scale convolutional kernels. Its localised receptive fields constrain the capacity for modelling global contextual information, making it difficult to effectively capture long-range dependencies. This results in limited expressive capability in corn field scenarios. Particularly in 6–8 leaf stage corn field environments, the varying capture distance between the camera on a robotic platform and the targets leads to significant scale variations of the corn root–stalk regions within the field of view. Coupled with interference from cluttered backgrounds and changes in lighting conditions, these factors contribute to increased missed-detection and false-detection rates for root–stalk recognition by the model. To address these limitations, the original Bottleneck structure was replaced by introducing and modifying the EBlock module. The specific architecture is depicted in Figure 3. The EBlock module comprises two primary components: a spatial attention module (SpAM) and a feed-forward network in the frequency domain (Fre-MLP). The SpAM builds upon the concept of NAFBlock [20]. It initially employs depthwise separable convolution to extract local texture features from the feature map. Subsequently, a gating mechanism (simple gate, SG) combined with simplified channel attention (SCA) performs adaptive weighting on the spatial features. This process enhances the response to critical regions, accentuates corn root–stalk characteristics, suppresses interference from weeds and background noise and ultimately improves detection accuracy. Subsequently, the Fre-MLP module performs global feature modelling in the frequency domain, effectively compensating for the limitations of local convolution in capturing long-range dependencies.

In this module, the features are first subjected to a fast Fourier transform (FFT), decomposing the frequency-domain representation into amplitude (modulus) and phase components. The amplitude spectrum characterises the intensity distribution of features across different frequency components, whereas the phase spectrum primarily encodes the spatial structure of the features. At this stage, only the amplitude spectrum is modelled

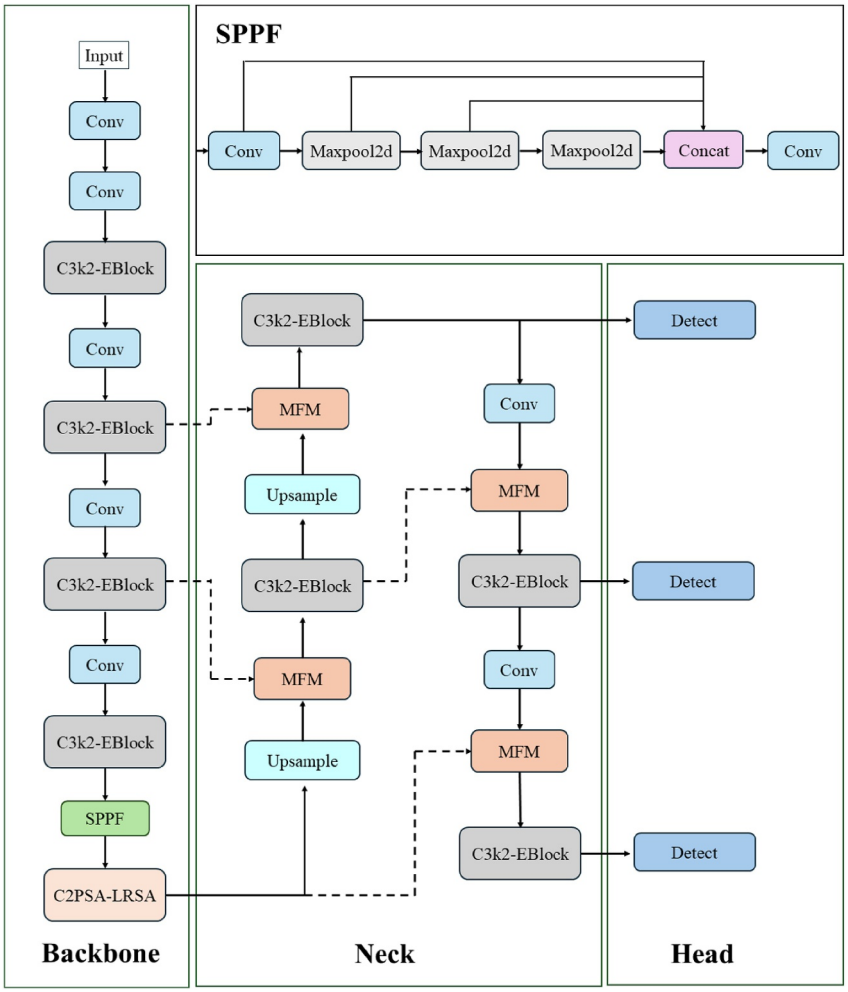

FIGURE 2 | Overall structure of the RO-YOLO model. Conv, convolutional block; SPPF, spatial pyramid pooling-fast module.

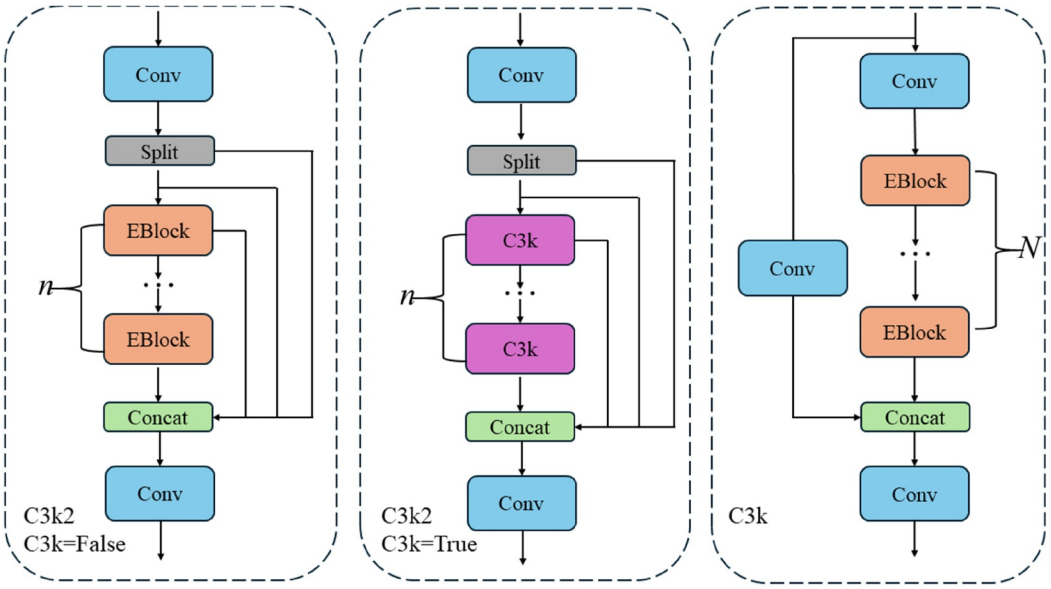

FIGURE 3 | Structure of the C3k2-EBlock module. Concat denotes the concatenation of features along the channel dimension, *n* denotes the number of repeated units in the C3k2-EBlock module. When C3k = False, the repeated unit is an Eblock module; when C3k = True, the repeated unit is a C3k sub-module. *N* represents the number of EBlocks stacked within each C3k module.

using an MLP, whereas the original phase is preserved to avoid disrupting spatial structure information. This approach maintains the stability of corn root–stalk spatial features while enhancing detection robustness under complex lighting conditions. Next, the processed amplitude is recombined with the original phase, and the features are mapped back to the spatial domain using the inverse FFT (IFFT). The frequency-domain enhanced features output by the Fre-MLP module modulate the input features through element-wise multiplication, thereby achieving frequency-selective enhancement. This design enables the model to adaptively enhance frequency components relevant to the target while suppressing background noise, all while preserving the stability of the spatial structure, thereby improving the robustness of feature representation. The EBlock leverages frequency-domain modelling to capture global information, significantly enhancing feature representation while slightly reducing parameter count and computational cost. This enables the model to be deployed on resource-constrained embedded devices. The overall structure of the EBlock is shown in Figure 4, and the complete process can be represented as follows:

$$\boldsymbol{x}_1 = \boldsymbol{x} + \mathrm{SpAM}(\mathrm{LN}(\boldsymbol{x})), \tag{1}$$

$$\boldsymbol{y} = \boldsymbol{x}_1 + \alpha \cdot \boldsymbol{x}_1 \odot \mathrm{FreMLP}(\mathrm{LN}(\boldsymbol{x}_1)). \tag{2}$$

Here, $\boldsymbol{x} \in \mathbb{R}^{C\times H\times W}$ represents the input features (where C, H and W denote the channel number, height and width of the feature map, respectively), $\boldsymbol{x}_1$ denotes the features after the first-stage, SpAM() denotes the spatial attention module, LN() denotes layer normalisation, FreMLP() denotes the feed-forward network in the frequency domain, $\odot$ is the element-wise multiplication, and $\boldsymbol{y}$ is the output feature.

2.2.2 | *Improvement of the C2PSA Module*

In corn field scenarios during the 6–8 leaf stage, corn root–stalk targets typically exhibit pronounced scale variations and complex spatial distributions, while being subject to substantial background interference from soil, crop residues, shadows and weeds. Therefore, the detection model must not only possess the capability to capture global contextual information, but also effectively extract local fine-grained features and multi-scale spatial information, to cope with target scale diversity and complex background disturbances. Although the C2PSA module in YOLO11 enhances feature representation through efficient convolution and self-attention mechanisms, its self-attention design exhibits limitations in extracting and integrating multi-scale contextual information. Specifically, it struggles to simultaneously capture local and global features across different receptive fields. In corn field environments characterised by large scale variations and complex backgrounds, this lack of targeted multi-scale modelling capability can lead to missed detections and false positives.

To overcome the above limitations, this study designs a C2PSA-LRSA module. Specifically, an LRSA module is introduced to replace the original attention mechanism in C2PSA, thereby reconstructing the PSABlock within C2PSA and forming the proposed PSABlock-LRSA, which substitutes the original PSABlock. The detailed structure of PSABlock-LRSA is illustrated in Figure 5. In its design, the LRSA module is specifically optimised for corn field scenarios. The LRSA module reduces computational complexity by performing self-attention calculations on downsampled feature spaces, while enhancing the capture of contextual dependencies through multiscale feature modelling. The specific procedure is as follows: given an input feature map $\boldsymbol{X} \in \mathbb{R}^{H\times W\times C}$, it is first downsampled to a fixed size via adaptive average pooling and then linearly projected to generate the query features. Concurrently, the module employs a pyramid pooling strategy to pool the input feature $\boldsymbol{X} \in \mathbb{R}^{H\times W\times C}$ into four distinct scales: 11 × 11, 8 × 8, 6 × 6 and 4 × 4, thereby extracting features across multiple spatial scales simultaneously. The pooled features at each scale are processed by a 3 × 3 depthwise separable convolution layer for local feature enhancement, capturing fine-grained local details of the targets. Subsequently, all enhanced multiscale features are concatenated along the spatial dimension. After layer normalisation, they are linearly projected to obtain the final key ($\boldsymbol{K}$) and value ($\boldsymbol{V}$) features. During the attention computation phase, the LRSA module partitions the query ($\boldsymbol{Q}$), $\boldsymbol{K}$ and $\boldsymbol{V}$ tensors into h attention heads along the channel dimension, with each head having a dimension of $d = C/h$. Scaled dot-product self-attention is then computed independently on each attention head:

$$\mathrm{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Softmax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^{\mathrm{T}}}{\sqrt{d}}\right)\boldsymbol{V}_i, \tag{3}$$

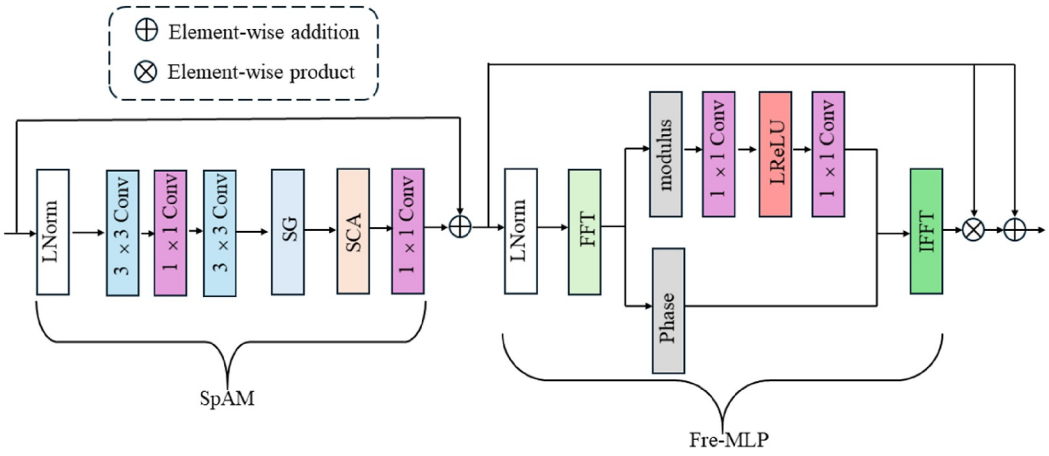

FIGURE 4 | Structure of the EBlock module. LNorm, layer normalisation operation; LReLU, leaky rectified linear unit activation function.

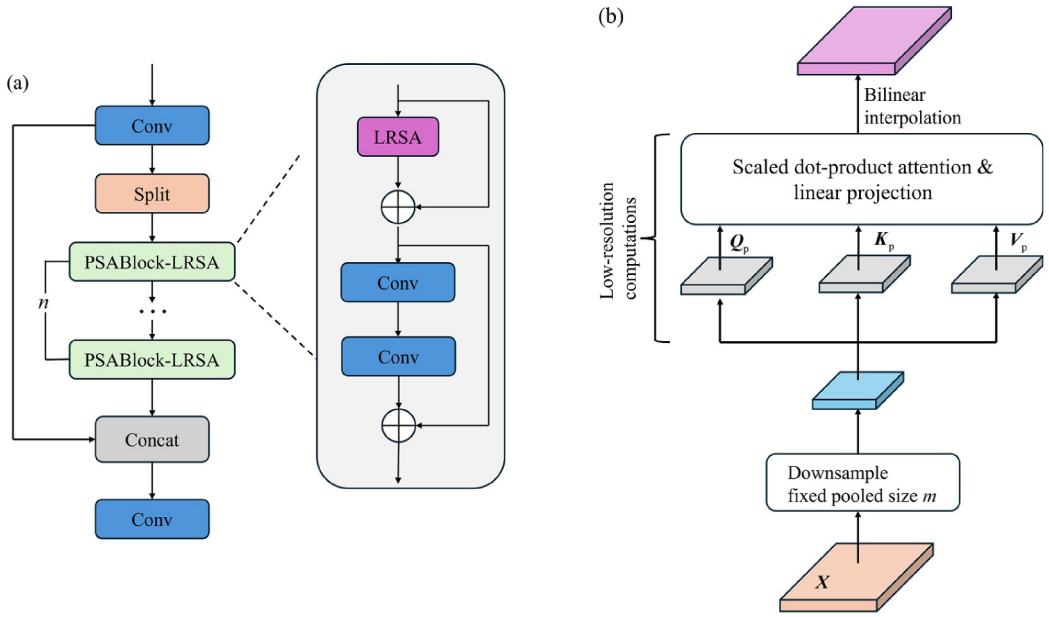

FIGURE 5 | Architecture of the C2PSA-LRSA module: (a) PSABlock-LRSA; (b) LRSA. $\boldsymbol{Q}_p$, $\boldsymbol{V}_p$ and $\boldsymbol{K}_p$ denote the query, key and value generated from pooled features, respectively. The subscript p denotes the pooled feature representation.

where $i \in \{1, 2, ..., h\}$, $\boldsymbol{Q}_i$, $\boldsymbol{K}_i$ and $\boldsymbol{V}_i$ represent the query, key, and value corresponding to the ith attention head, respectively and Attention() denotes the scaled dot-product attention mechanism. Subsequently, the resulting attention output is upsampled to the spatial resolution of the original feature map using bilinear interpolation. Bilinear interpolation provides a smooth upsampling strategy that generates continuous feature maps without introducing additional parameters, thereby avoiding interpolation artefacts.

In summary, the LRSA module downsamples the input features and performs self-attention computation in the reduced-resolution feature space, thereby effectively reducing computational complexity. During the construction of key and value, the LRSA module employs a pyramid pooling strategy combined with depthwise separable convolution for local feature enhancement, enabling the model to capture fine-grained local details while effectively extracting global contextual information. During the attention computation process, the module enhances the feature response in the corn root–stalk region while suppressing irrelevant background features. This significantly improves the model's feature representation capability in complex backgrounds, thereby increasing detection accuracy.

2.2.3 | *Improvements to the Neck Network*

In the YOLO series of detection frameworks, the neck network typically employs feature concatenation to fuse multiscale features extracted from different stages. Although this module offers advantages of simplicity and computational efficiency in general object detection tasks, its fusion operation is fundamentally a simple concatenation along the channel dimension. This approach lacks an effective mechanism to assess the relative importance of feature representations from different scales, leading to inadequate utilisation of highly discriminative features. Specifically, in the corn root–stalk identification task, the target is often susceptible to interference from factors such as lighting, occlusion by leaves and weeds and background noise. Simple concatenation fails to dynamically adjust the weighting of features across different scales, which prevents the emphasis of critical feature information. Consequently, this reduces detection accuracy and robustness, leading to an increase in missed or false detections. To address the aforementioned limitations, an MFM is introduced. The specific structure is illustrated in Figure 6. After modification, MFM is used to replace the original feature concatenation module in the neck network of YOLO11. The core advantage of MFM lies in introducing a scale-aware, channel-wise attention mechanism during multiscale feature fusion. By leveraging global statistical information obtained through global average pooling of the fused features, the module adaptively modulates the response strength of features at different scales. This effectively suppresses background noise and enhances the robustness of corn root–stalk detection in complex scenarios involving occlusions and small targets. The main procedure is as follows: MFM first employs a 1×1 convolution to perform channel alignment on two features $\overline{\boldsymbol{F}}_1 \in \mathbb{R}^{H \times W \times C_1}$ and $\overline{\boldsymbol{F}}_2 \in \mathbb{R}^{H \times W \times C_2}$ from different scales, thereby projecting the multiscale features into a unified feature dimension to generate aligned features $\boldsymbol{F}_1 \in \mathbb{R}^{H \times W \times C}$ and $\boldsymbol{F}_2 \in \mathbb{R}^{H \times W \times C}$. Subsequently, the two features are combined through element-wise addition to construct a global feature representation, followed sequentially by global average pooling, a lightweight multilayer perceptron and softmax normalisation, thereby generating a scale-adaptive attention weight tensor $\boldsymbol{A}_F$ for each channel. Finally, $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are each multiplied element-wise with the corresponding main elements of the attention weight tensor $\boldsymbol{A}_F$ and then summed to obtain the final output $\boldsymbol{O}_F$. The entire process can be expressed as follows:

$$\boldsymbol{F}_1 = f_{1\times1}(\overline{\boldsymbol{F}}_1), \tag{4}$$

$$\boldsymbol{F}_2 = f_{1\times1}(\overline{\boldsymbol{F}}_2), \tag{5}$$

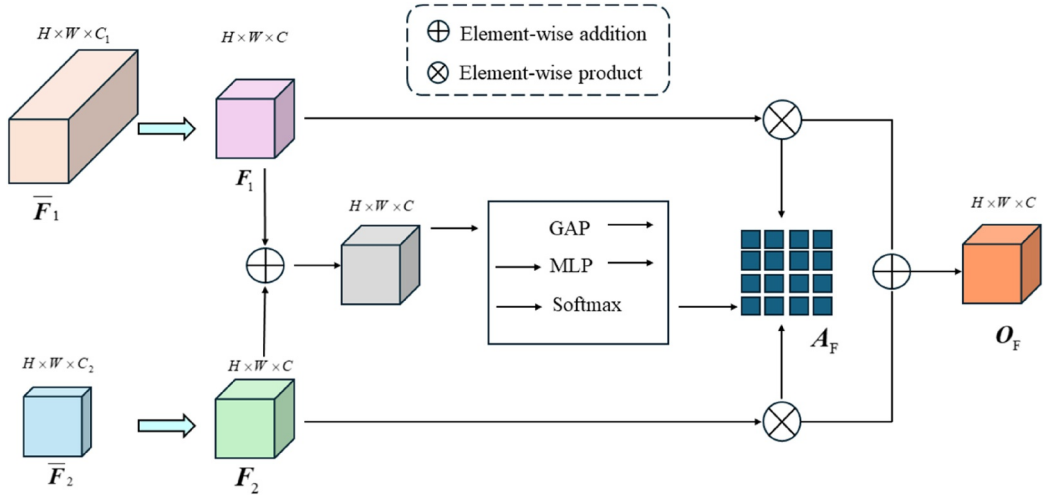

FIGURE 6 | Structure of MFM. GAP, global average pooling; MLP, multilayer perceptron.

$$\boldsymbol{O}_{\mathrm{F}} = \boldsymbol{A}_{\mathrm{F}} \odot \boldsymbol{F}_1 + \boldsymbol{A}_{\mathrm{F}} \odot \boldsymbol{F}_2. \tag{6}$$

Here, $f_{1\times1}$ () denotes the 1 × 1 convolution operation.

Unlike simple concatenation, MFM introduces a channel-conditioned adaptive attention mechanism across different feature branches to achieve selective fusion of multiscale features, effectively overcoming the limitation of traditional feature concatenation, which merely stacks information without selectivity. Specifically, MFM leverages global statistical information to assign normalised attention weights to features of different scales within the same channel. This process highlights the scale responses that are more discriminative for the current target while suppressing redundant or noisy features.

2.3 | Navigation Line Extraction

2.3.1 | *Feature Point Extraction*

After successfully obtaining the bounding boxes of corn root–stalk regions using the improved YOLO11 model, the model outputs the bounding box information. Subsequently, the bottom-centre point of each bounding box is calculated through postprocessing of the obtained coordinates. In this study, these extracted bottom-centre points of the detection boxes serve as the feature points for subsequent navigation line extraction, as illustrated in Figure 7. The calculation process is as follows:

$$x_{\mathrm{c}} = \frac{x_{j2} - x_{j1}}{2}, \tag{7}$$

$$y_{\mathrm{c}} = y_{j2}. \tag{8}$$

Here, x_{j1} and x_{j2} denote the left and right horizontal coordinates of the bounding box, respectively; y_{j2} represents the vertical coordinate of the lower-corner point and $(x_{\mathrm{c}}, y_{\mathrm{c}})$ indicates the vertical coordinate of the bottom-centre point. Compared to the geometric centre of the bounding box, the bottom-centre point is closer to the actual contact position between the corn root–stalk and the ground, making it more suitable for the subsequent extraction and fitting of navigation lines in corn fields. This approach avoids complex feature extraction or keypoint detection by directly utilising the bottom-centre point of the bounding box as the crop row feature point, thereby significantly reducing the time required for feature point localisation.

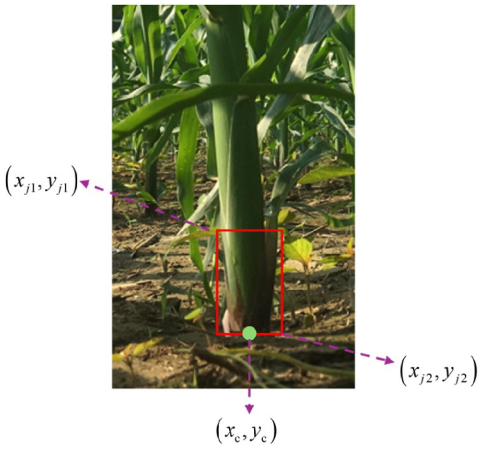

FIGURE 7 | Schematic diagram of feature points.

2.3.2 | *Feature Point Filtering*

To effectively eliminate interference from outlier and noisy points in distant crop rows, and to enable the robot to reliably extract the required feature points from detected multirow crop features, this study proposes a feature point selection algorithm based on coarse lateral partitioning and precise longitudinal near-field filtering. The specific steps are as follows:

1. Feature point extraction. The crop bounding boxes output by the object detection model are converted into feature points by selecting the bottom centre of each bounding box as a crop feature point, thereby forming the initial feature point set.
2. Left–right region partitioning. Taking the vertical midline of the image ($x_1 = w/2$) as the initial reference (w is the width), all feature points are first partitioned into left and right sets according to their x-coordinates. The mean x-coordinate of each set is then computed, and the midpoint between these two means is adopted as the dynamic midline. Using this updated midline, all feature

points in the image are precisely reclassified into left and right regions. This procedure effectively compensates for the row asymmetry caused by camera tilt or misalignment on the robot platform. By leveraging the strong spatial prior that crop rows appear as left–right distributed patterns in the robot's viewpoint, the method provides robust initial region constraints for subsequent processing stages.

3. Near-field candidate feature. For the left and right point sets, the feature points are separately sorted in descending order according to their y-coordinates. The densest M feature points with the largest y-values (where N is a fixed value or a proportion; in this study, $M = 10$) are selected as candidate crop row feature point sets. This step preserves feature points that are closest to the robot and exhibit a continuous distribution in the image, while effectively eliminating distant interference points.
4. Crop row fitting. Based on the filtered left and right crop row feature points, straight lines are independently fitted using the random sample consensus (RANSAC) algorithm. The combination of the proposed feature point selection method with RANSAC can suppress the influence of near-field outliers to a certain extent. During line fitting, RANSAC identifies points that deviate from the main crop row distribution as outliers and excludes them, thereby improving the stability and robustness of navigation baseline extraction.

Near-field feature points correspond to the area closest to the robot's field of view, where corn root and stalk exhibit clear imaging and minimal occlusion. These points serve as the optimal samples for fitting the current interrow path. This method effectively eliminates interference points from distant corn rows and adjacent rows, retaining only the dense, nearby feature points within the current crop row for fitting. This provides a reliable foundation for the precise generation of navigation lines. The detailed steps of our feature point selection algorithm are summarised in Algorithm 1 ($p.x$, Right_coarse.x, and Left_coarse.x denotes the x-coordinate of an individual feature point, respectively).

ALGORITHM 1 | Feature point selection algorithm.

Input: input image, YOLO model weights weight_path, confidence threshold conf, number of selected bottom points per side M

Output: Left_use, Right_use

// Detect corn roots in the input image using the YOLO model to obtain bounding box set

Boxes ← YOLO_Detect(image, weight_path, conf)
if Boxes = ø then
    return ø
end if
for each bounding box $(x_1, y_1, x_2, y_2) \in$ Boxes do
    feature_point←$((x_1+x_2)/2, y_2)$
    Add feature_point to Points
end for
// Obtain the full set of feature points, Points
Compute the image width midline: mid_x_coarse = image.width/2
Left_coarse←{$p \in$ Points | $p.x<$mid_x_coarse}
Right_coarse←{$p \in$ Points | $p.x\geq$mid_x_coarse }
if |Left_coarse|<3 or |Right_coarse|<3 then
    mid_x_fine←mid_x_coarse
else
    mid_x_fine←(mean(Left_coarse.x) + mean (Right_coarse.x))/2
end if
Left_all←{$p \in$ Points | $p.x<$mid_x_fine }
Right_all←{$p \in$ Points | $p.x\geq$mid_x_fine }
// Sort Left_all and Right_all in descending order of y-coordinate
Left_use←first M points of Left_all (or all if fewer)
Right_use←first M points of Right_all (or all if fewer)
Return Left_use, Right_use

2.3.3 | *Navigation Line Fitting*

Since the above feature point selection algorithm cannot eliminate outliers in the near-field region, further measures are required to address issues such as false detections, outlier points and occasional interference from adjacent corn rows in corn root–stalk detection during the 6–8 leaf stage. Therefore, the RANSAC algorithm [21] is introduced to perform line fitting on corn root–stalk feature points, thereby enhancing overall robustness. RANSAC estimates model parameters by repeatedly and randomly sampling minimal subsets from a point set containing outliers and iteratively refining the model until a satisfactory solution is obtained, ultimately yielding an optimal model that is insensitive to outliers. The specific steps are as follows:

1. Randomly select the minimal subset of corn root–stalk feature points required to estimate the model parameters, and use this subset to calculate the line model parameters (for line fitting, the minimal sample size $S = 2$, i.e., two points are randomly selected).
2. All feature points are evaluated against the current model. Based on a preset distance threshold, it is determined whether each point satisfies the model constraints, and the number of inliers is counted.
3. The number of inliers in the current model is compared with that of the previous optimal model, and the model parameters corresponding to the maximum inlier count are recorded.
4. The above steps are repeated until either the preset maximum number of iterations k is reached or the number of inliers in the current model meets the required threshold.

To ensure that the algorithm obtains the correct model with high probability, the number of iterations k must be properly set. Assuming that the proportion of inliers in the data is t, and then

$$t = \frac{n_1}{n_1 + n_2}, \tag{9}$$

where n_1 is the number of inliers and n_2 is the number of outliers. Under the condition that each model calculation uses S

points, the probability that at least one selected point is not an inlier is denoted as $1 - t^S$. Over the course of k iterations, $\left(1 - t^S\right)^k$ represents the probability that at least one sample contains no inliers. Consequently, the probability of sampling the correct S points to estimate the correct model is $P = 1 - \left(1 - t^S\right)^k$.

From the equation above, the required minimum number of iterations can be derived as

$$k = \frac{\ln(1 - P)}{\ln\left(1 - t^S\right)}. \tag{10}$$

The inlier ratio t is typically treated as a prior value, and P represents the desired probability of RANSAC obtaining the correct model. In this case, the optimal number of iterations k can be calculated. However, in practical tasks, the inlier ratio is affected by factors such as detection errors, occlusions and lighting conditions, causing it to vary dynamically. As a result, accurately estimating t is difficult, making it challenging to determine the precise number of iterations using the formula above. To address the aforementioned issue, this study adopts the approach of setting a maximum iteration count. During algorithm execution, the computational complexity is controlled by limiting the maximum number of iterations, whereas the optimal model is dynamically updated based on the inlier count in each iteration. The algorithm may terminate early when the model's inlier count stabilises or meets a predefined condition. In the experiments conducted in this study, RANSAC operates on the results from RO-YOLO detection and the subsequent feature point filtering algorithm. Since the feature points have undergone two stages of screening, the maximum number of iterations was set to 100. This setting ensures model fitting accuracy while maintaining computational efficiency. Experimental results indicate that with this parameter configuration, the algorithm can stably obtain crop row models with high inlier consensus. Following the procedure described, the RANSC algorithm is employed to derive the left and right navigation baselines, respectively.

The angle bisector of the two navigation baselines corresponds to the navigation line for corn fields at the 6–8 leaf stage. Let the slope of the left navigation baseline be β_1 and the slope of the right navigation baseline be β_2; then the slope of the navigation line, β, satisfies

$$\frac{\beta - \beta_1}{1 + \beta\beta_1} = \frac{\beta_2 - \beta}{1 + \beta\beta_2}. \tag{11}$$

TABLE 1 | Experimental environment configuration.

Configuration	Detail
System environment	Windows 11(24H2)
Central processing unit (CPU)	Intel Core i9-10850K CPU@ 3.7 GHz
Graphics processing unit	NVIDIA GeForce RTX3090 24G
Graphics processing unit acceleration library	CUDA 12.4
Random access memory	64G
Algorithm framework	Pytorch 2.5.1

3 | Results and Analysis

3.1 | Experimental Environment

To ensure standardised, efficient and unbiased experimentation, all experiments involving structural improvements and model training in this study are conducted on a computer with a fixed configuration. The experimental environment specifications are listed in Table 1. During the training process, the input image size is set to 640 × 640 pixels, the batch size to 16, and the total number of training epochs to 300. The model is optimised using stochastic gradient descent (SGD) with an initial learning rate of 0.0001, a weight decay coefficient of 0.0005 and a momentum of 0.937. Unless otherwise specified, all other training hyperparameters used their default values.

3.2 | Corn Root–Stalk Detection Experiment

3.2.1 | Algorithm Performance Metrics for Corn Root–Stalk Detection Experiments

To comprehensively evaluate the performance of the RO-YOLO model, a series of comparative experiments are conducted. First, model accuracy was assessed using precision (P), recall and average precision at an intersection over union (IoU) threshold of 0.5 (AP@0.5). Second, model complexity is measured using the number of parameters (P_{param}) and the floating-point operations required for a single forward pass (F_{FLOP}), with F_{FLOP} expressed in units of GFLOPs (10^9 floating-point operations).

3.2.2 | Ablation Study and Results

To investigate the impact of the proposed improvements on model detection performance, ablation experiments are conducted under the same environment and hyperparameters. In this study, the targeted modules are sequentially added to the baseline model to evaluate the effects of C3k2-EBlock, C2PSA-LRSA and MFM on model performance. The experimental results are presented in Table 2. As modules are incrementally added, the AP@0.5 metric shows an overall upward trend. Compared to Model 1, Model 2 exhibits a slight decrease in some performance metrics but achieves greater model lightweighting. This indicates that the C3k2-EBlock module effectively balances lightweight design with competent feature extraction capability. Subsequently, to compensate for the potential decline in representational capacity due to lightweighting, the LRSA module is introduced to replace the original attention mechanism in the C2PSA module. This enhancement strengthens the model's ability to capture fine-grained features and global contextual information. Consequently, Model 3 demonstrates a significant improvement over Model 1, with a 1.1 percentage points (PPs) increase in AP@0.5. To mitigate interference from complex field backgrounds, MFM is incorporated. By suppressing nontarget features, this module further improves model accuracy, reflected in a 1.6 PPs gain in AP@0.5. Ultimately, the improved model RO-YOLO, which integrates all proposed modules, achieves the increases of 2.1 PPs in recall and 2.5 PPs in AP@0.5 while maintaining a parameter count comparable to the original model. This

TABLE 2 | Results of the ablation experiment.

Model	C3k2-EBlock	C2PSA-LRSA	MFM	P_{Param} ($\times 10^6$)	F_{FLOP} ($\times 10^9$)	P (%)	R (%)	AP@0.5 (%)
1				2.6	6.6	90.8	86.2	90.2
2	√			**2.3**	5.9	90.9	87.7	89.6
3		√		2.6	6.4	90.6	86.4	91.3
4			√	2.6	6.5	91.1	85.7	91.8
5	√	√		**2.3**	5.9	91.2	87.5	92.1
6	√	√	√	**2.3**	**5.8**	**91.4**	**88.3**	**92.7**

Note: Best results are in bold.

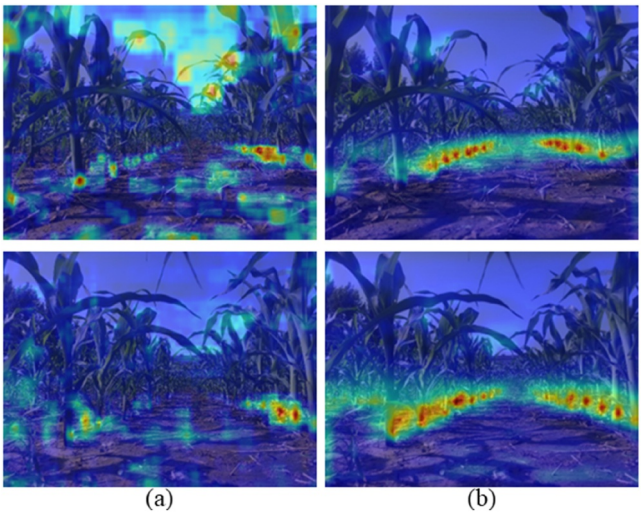

FIGURE 8 | Heatmap visualisation comparison: (a) YOLO11n; (b) RO-YOLO.

result fully demonstrates the model's effectiveness and practicality for the corn root–stalk detection task.

This study employs Grad-CAM to visualise and compare the attention regions of the traditional YOLO11n model and the proposed RO-YOLO model. Figure 8 presents the visualisation results, where brighter areas indicate higher detection confidence. As shown in the figure, RO-YOLO focuses more precisely on the corn root–stalk regions and effectively suppresses interference from background weeds. This demonstrates that RO-YOLO performs better in the task of corn root–stalk identification in field environments.

As shown in Figure 9, the detection results of YOLO11n and the optimised RO-YOLO model on the same dataset are presented. By comparison, the proposed improved RO-YOLO model exhibits higher confidence scores and a reduced rate of missed detections. Despite complex field conditions involving weed interference, lens occlusion and varying lighting conditions, the improved RO-YOLO model demonstrates stronger robustness. Through architectural improvements and a lightweight design, the model significantly enhances detection accuracy while maintaining its lightweight characteristics, making it better adapted to the detection environment in corn fields.

3.2.3 | *Comparative Experiments of Different Models*

To objectively evaluate the performance of the improved RO-YOLO model, comparative experiments are conducted on a unified platform. Selected models for comparison include the YOLO-series models YOLOv8n, YOLOv10n and YOLO12n, the RT-DETR (real-time detection Transformer) baseline models RT-DETR-R18 and RT-DETR-R34 [22], as well as Gold-YOLO-s and Gold-YOLO-m [23]. All models are trained under identical experimental conditions. The experimental results are presented in Table 3. Although the two RT-DETR models exhibit superior performance across multiple evaluation metrics, their excessively large parameter sizes and complex architectures result in high computational and storage overheads. This makes them difficult to deploy on resource-constrained embedded devices and prevents them from meeting the stringent real-time requirements of agricultural robots. In contrast, the proposed RO-YOLO achieves the best recall and AP@0.5 among all models except RT-DETR, while maintaining relatively low parameter counts and computational complexity. At the same time, it demonstrates competitive performance across all metrics, making it well-suited for deployment on embedded devices with strict real-time constraints.

To more intuitively illustrate the detection performance of the above eight models in corn field scenarios, Figure 10 presents a normalised comparison of multiple performance metrics for different models. For the number of parameters and floating-point operations, the values are represented as one minus their normalised values. As shown in the figure, the RO-YOLO model proposed in this paper possesses the smallest number of parameters and the lowest F_{FLOP} among the eight models, while achieving competitive detection accuracy. This makes it more suitable for deployment in resource-constrained scenarios.

3.3 | Navigation Line Extraction Experiment

To systematically validate the accuracy, real-time performance and scene robustness of the proposed corn row navigation line extraction algorithm, a dedicated test dataset is constructed in this study. The dataset focuses on two representative and challenging scenarios—dense planting and sparse planting—and contains a total of 300 images for targeted evaluation. The evaluation metrics include the single-frame static image angular error θ, the average fitting time t and the accuracy q. As shown in Figure 11, the extracted row centreline is compared with the

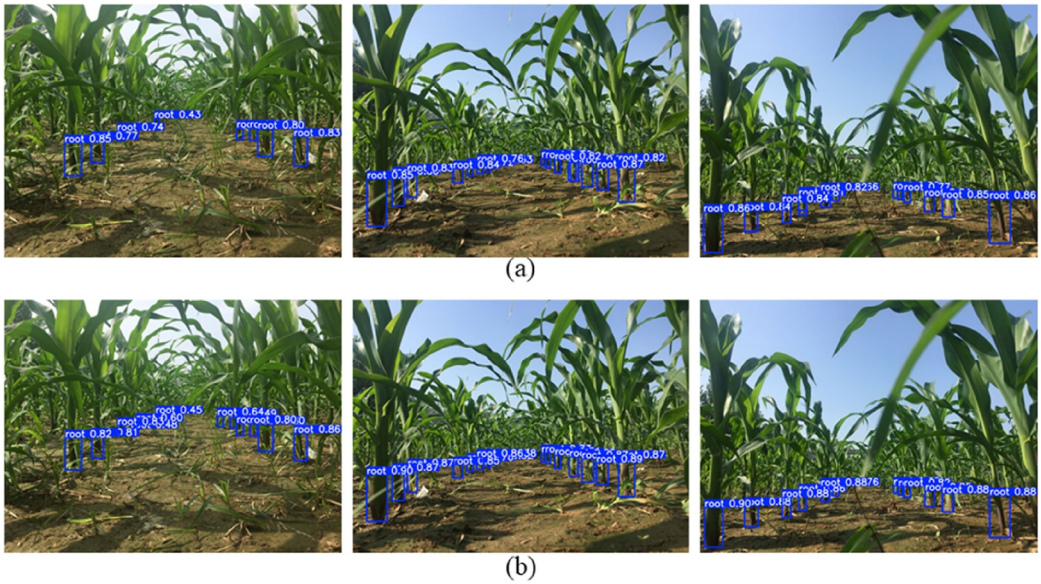

FIGURE 9 | Comparison of detection results: (a) YOLO11; (b) RO-YOLO.

TABLE 3 | Comparison results of different models.

Model	P_{Param} (M)	F_{FLOP} (G)	P (%)	R (%)	AP@0.5 (%)
YOLOv8n	3.2	8.7	90.7	80.6	89.2
YOLOv10n	**2.3**	6.7	90.1	86.4	90.3
YOLO12n	2.6	6.7	91.5	83.1	90.5
Gold-YOLO-s	21.5	46	87.3	78.2	88.5
Gold-YOLO-m	41.3	87.5	89.6	79.6	91.6
RT-DETR-R18	20.2	58.8	93.8	92.1	94.1
RT-DETR-R34	31.4	90.6	**94.6**	**93.8**	**94.9**
RO-YOLO	**2.3**	**5.8**	91.4	88.3	92.7

Note: Best results are in bold.

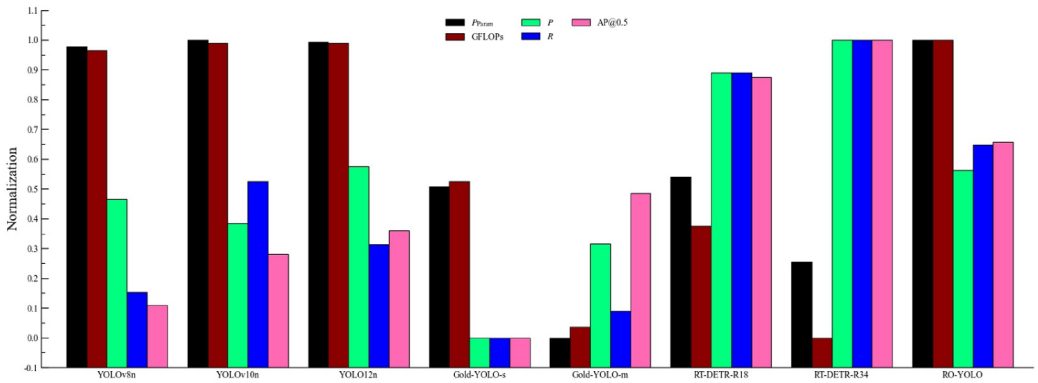

FIGURE 10 | Normalisation analysis of multiple indicators.

manually annotated row centreline to obtain the angular error, defined as the acute angle between the two navigation lines. If the angular error falls within the range of 0°–5°, the corresponding navigation baseline is regarded as accurate [24]. The average fitting time reflects the time required by the algorithm to process the corn row navigation line for each image. The accuracy is measured as the ratio of the number of navigation lines with an angular deviation smaller than 5° to the total number of navigation lines. The experimental results are summarised in Table 4. The results indicate that the proposed algorithm achieves a navigation line accuracy of 92.0% with an average processing time of 51.6 ms, satisfying the requirements for both high accuracy and real-time performance. Moreover, the method demonstrates strong robustness under common field conditions, such as dense weed infestation and missing plants. Figure 12 illustrates the navigation line generation process and presents visualisation results for two representative scenarios: sparse planting and dense planting. The results

visually confirm that the proposed method can robustly and consistently generate accurate navigation lines across different field environments, thereby validating its effectiveness and practical applicability for navigation line extraction in corn fields at the 6–8 leaf stage.

Real-time performance is generally defined as the capability of a system to process continuous data streams without noticeable delay. To evaluate the real-time performance of the proposed navigation line extraction algorithm under the typical resource constraints of edge computing devices, experiments are conducted on an Intel Core i9-10850K@3.7GHz CPU. Inference speed is adopted as the quantitative evaluation metric. As shown in Table 5, the algorithm achieves a single-frame inference latency of approximately 82.7 ms (about 12.1 frames per second [FPS]) on the CPU platform, demonstrating its feasibility for deployment on edge CPU devices. Under the typical low-to-medium speed operating conditions of weeding robots, this frame rate ensures that the physical displacement between two consecutive frames remains at the centimetre level. This guarantees that the vision sensor can provide stable and continuous coordinates to the weeding actuator, thereby enabling the robot to maintain a stable travelling speed without reducing operational accuracy.

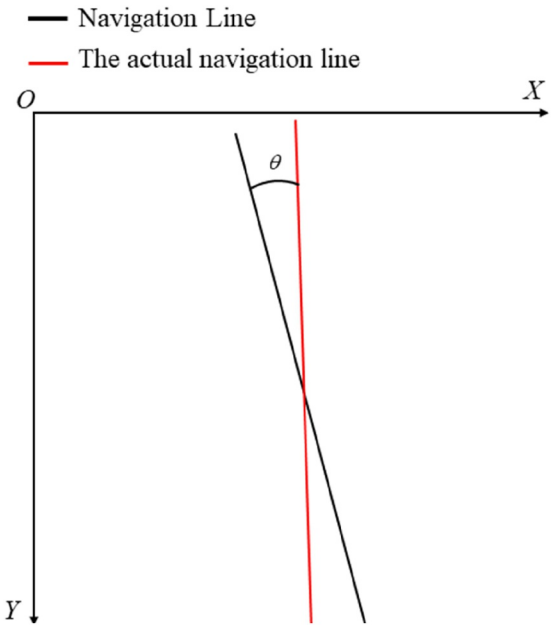

FIGURE 11 | Definition of navigation line recognition deviation.

TABLE 4 | Definition of navigation line recognition deviation.

Number of images	Average angle error (°)	Average accuracy rate (%)	Average fitting time (ms)
300	3.6	92.0	51.6

4 | Conclusion

Aiming at the navigation requirements in 6–8 leaf stage corn fields, this study proposes a navigation line extraction algorithm based on a deep learning network. First, a corn root–stalk recognition network named RO-YOLO is constructed based on an improved YOLO11n architecture. During the model design, the original C3k2 module is replaced with the proposed C3k2-EBlock module, which enhances feature extraction capability while maintaining model lightweightness. Subsequently, a novel attention mechanism, LRSA, is introduced into the C2PSA module, enabling collaborative modelling of local detail information and global dependencies. Finally, a new fusion module, MFM, is adopted to replace the original concatenation operation in the neck network, which suppresses background noise and interference features, thereby improving the capture of fine-grained features of corn root–stalk. Experimental results demonstrate that the improved RO-YOLO model achieves a

TABLE 5 | Real-time performance experiment.

Configuration	Average fitting time (ms)
Intel Core i9-10850K@3.7GHz	82.7
Intel Core i9-10850K@3.7GHz+ NVIDIA GeForce RTX3090 24G	51.6

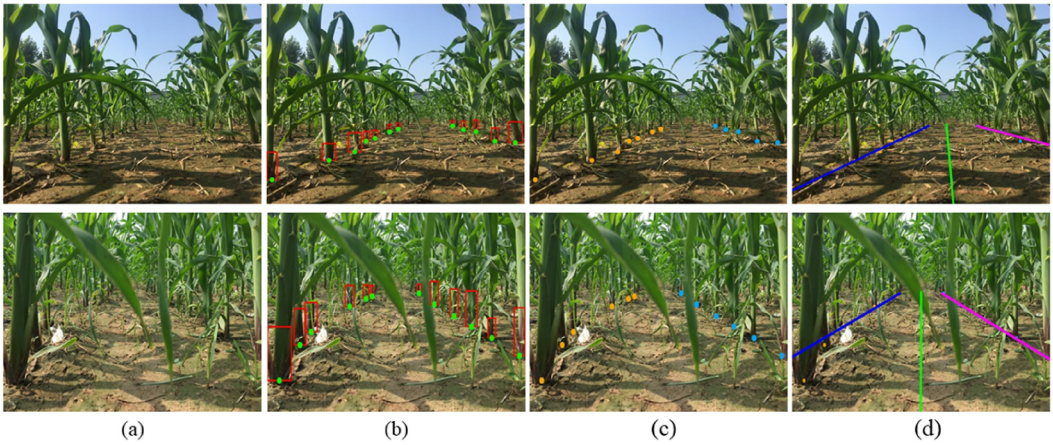

FIGURE 12 | Procedure of navigation line generation: (a) Test image; (b) detection of feature points; (c) filtering of feature points; (d) result of navigation line extraction.

significant enhancement in both recall and mAP@0.5 compared to the original YOLO11n model. Subsequently, reference points are extracted based on the network outputs, and a feature point selection strategy combining coarse lateral partitioning and precise longitudinal near-field filtering is proposed to eliminate distant outliers and classify crop row feature points. Finally, the navigation baselines are fitted using the RANSAC algorithm based on the selected feature points, and the navigation line is obtained by computing the angle bisector of the two navigation baselines.

The proposed corn crop row centre navigation line detection algorithm achieves an average fitting time of only 51.6 ms while maintaining real-time performance and a high accuracy of 92.0%, demonstrating that the proposed method provides a reliable solution for navigation line extraction in corn fields at the 6–8 leaf stage and exhibits clear practical value. However, the data used in this study are primarily collected from a single region and a single corn variety. Although multiple lighting conditions are included, further validation is still required in cross-regional scenarios and in corn fields with different varieties. Future studies will further expand the dataset by incorporating image data from different leaf stages, regions, weather conditions and growth stages, with the aim of continuously improving the robustness and adaptability of the algorithm under nonideal conditions.

Funding

This work was supported by the China Scholarship Council (CSC) (Grants 202401040020 and 202401040021) and the Key Research and Development Program of Shandong Province (Science and Technology-based Small and Medium-sized Enterprises Innovation Capability Enhancement Project) (Grant 2025TSGCCZZB0802).

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author.

References

1. P. Chattopadhyay, H. P. Patel, and V. Parmar, "Internet of Things (IoT) in Smart Agriculture," in *2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC)* (IEEE, 2022), 536–540, https://doi.org/10.1109/ICESC54411.2022.9885655.

2. J. Zhou, S. Geng, Q. Qiu, Y. Shao, and M. Zhang, "A Deep-Learning Extraction Method for Orchard Visual Navigation Lines," *Agriculture* 12, no. 10 (2022): 1650, https://doi.org/10.3390/agriculture12101650.

3. C. L. Nkwocha and N. Wang, "Deep Learning-based Semantic Segmentation With Novel Navigation Line Extraction for Autonomous Agricultural Robots," *Discover Artificial Intelligence* 5, no. 1 (2025): 73, https://doi.org/10.1007/s44163-025-00301-0.

4. Z. Diao, S. Ma, J. Li, et al., "Navigation Line Detection Algorithm for Corn Spraying Robot Based on Improved LT-YOLOv10s," *Precision Agriculture* 26, no. 3 (2025): 46, https://doi.org/10.1007/s11119-025-10243-3.

5. A. Kamilaris and F. X. Prenafeta-Boldú, "Deep Learning in Agriculture: A Survey," *Computers and Electronics in Agriculture* 147 (2018): 70–90, https://doi.org/10.1016/j.compag.2018.02.016.

6. L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder With Atrous Separable Convolution for Semantic Image Segmentation," *Computer Vision—ECCV 2018* (Springer, 2018): 833–851, https://doi.org/10.1007/978-3-030-01234-2_49.

7. H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2017), https://doi.org/10.1109/CVPR.2017.660.

8. A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmen Tation," *arXiv preprint arXiv:1606.02147* (2016), https://doi.org/10.48550/arXiv.1606.02147.

9. S. P. Adhikari, G. Kim, and H. Kim, "Deep Neural Network-Based System for Autonomous Navigation in Paddy Field," *IEEE Access* 8 (2020): 71272–71278, https://doi.org/10.1109/ACCESS.2020.2987642.

10. W.-S. Kim, D.-H. Lee, Y.-J. Kim, T. Kim, R.-Y. Hwang, and H.-J. Lee, "Path Detection for Autonomous Traveling in Orchards Using Patch-Based CNN," *Computers and Electronics in Agriculture* 175 (2020): 105620, https://doi.org/10.1016/j.compag.2020.105620.

11. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2016), 779–788, https://doi.org/10.1109/CVPR.2016.91.

12. R. Girshick, "Fast R-CNN," in *2015 IEEE International Conference on Computer Vision (ICCV)* (IEEE, 2015), 1440–1448, https://doi.org/10.1109/ICCV.2015.169.

13. W. Liu, D. Anguelov, D. Erhan, et al., "SSD: Single Shot MultiBox Detector," in *Computer Vision—ECCV 2016* (Springer, 2016), 21–37, https://doi.org/10.1007/978-3-319-46448-0_2.

14. Z. Diao, P. Guo, B. Zhang, et al., "Navigation Line Extraction Algorithm for Corn Spraying Robot Based on Improved YOLOv8s Network," *Computers and Electronics in Agriculture* 212 (2023): 108049, https://doi.org/10.1016/j.compag.2023.108049.

15. A. S. Pinto de Aguiar, F. B. Neves dos Santos, L. C. Feliz dos Santos, V. M. de Jesus Filipe, and A. J. Miranda de Sousa, "Vineyard Trunk Detection Using Deep Learning—An Experimental Device Benchmark," *Computers and Electronics in Agriculture* 175 (2020): 105535, https://doi.org/10.1016/j.compag.2020.105535.

16. Q. Liu, J. Lv, and C. Zhang, "MAE-YOLOv8-Based Small Object Detection of Green Crisp Plum in Real Complex Orchard Environments," *Computers and Electronics in Agriculture* 226 (2024): 109458, https://doi.org/10.1016/j.compag.2024.109458.

17. D. Feijoo, J. C. Benito, A. Garcia, and M. V. Conde, "DarkIR: Robust Low-Light Image Restoration," in *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2025), 10879–10889, https://doi.org/10.1109/CVPR52734.2025.01016.

18. Y.-H. Wu, S. C. Zhang, Y. Liu, et al., "Low-Resolution Self-Attention for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 47, no. 9 (2025): 8180–8192, https://doi.org/10.1109/tpami.2025.3577035.

19. Y. Zhang, S. Zhou, and H. Li, "Depth Information Assisted Collaborative Mutual Promotion Network for Single Image Dehazing," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2024), 2846–2855, https://doi.org/10.1109/CVPR52733.2024.00275.

20. L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple Baselines for Image Restoration," in *Computer Vision—ECCV 2022* (Springer, 2022), 17–33, https://doi.org/10.1007/978-3-031-20071-7_2.

21. M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting With Application s to Image Analysis and Automated Cartography," *Communications of the ACM* 24, no. 6 (1981): 381–395, https://doi.org/10.1145/358669.358692.

22. Y. Zhao, W. Lv, S. Xu, et al., "DETRs Beat YOLOs on Real-time Object Detection," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2024), 16965–16974, https://doi.org/10.1109/CVPR52733.2024.01605.

23. C. Wang, W. He, Y. Nie, et al., "Gold-YOLO: Efficient Object Detector via Gather-and-Distribute Mechanism," *arXiv preprint arXiv: 2309.11331* (2023), https://doi.org/10.48550/arXiv.2309.11331.

24. T. Liu, Y. Zheng, J. Lai, et al., "Extracting Visual Navigation Line Between Pineapple Field Rows Based on an Enhanced YOLOv5," *Computers and Electronics in Agriculture* 217 (2024): 108574, https://doi.org/10.1016/j.compag.2023.108574.