CRS-YOLO: EFFICIENT INSTANCE SEGMENTATION FOR CROP ROW DETECTION AND NAVIGATION IN TOBACCO FIELDS
/
CRS-YOLO：用于烟草田作物行检测和导航的高效实例分割

Feng LIU, Bingjie CHEN, Yong PANG, Yue LUO, Baoshan WANG, Chenhui ZHU*), Wanzhang WANG*)
College of Mechanical and Electrical Engineering, Henan Agricultural University, Zhengzhou 450002, China
Tel: +8613513711991; Email: wangwz@henau.edu.cn
Corresponding author: Wanzhang Wang
DOI: https://doi.org/10.35633/inmateh-78-45

Keywords: *Navigation line extraction; Tobacco harvester; Crop row detection; Improved YOLOv11 network*

ABSTRACT
This research aimed to develop a robust crop row detection framework for tobacco fields, where existing methods have not addressed the unique challenges of stratified harvesting, which progressively alters plant morphology from dense lower canopy structures to sparse upper leaf arrangements across multiple collection cycles. CropRowSeg-YOLO (CRS-YOLO), an instance segmentation framework built on YOLOv11, was developed and integrates three core innovations: a Spatial Enhanced Calibration Block (SECB) utilizing depth-wise separable convolutions for spatial feature enhancement; a Hierarchical Adaptive Segmentation Head (HASH) employing asymmetric kernels for long-range dependency capture; and an Enhanced Post-processing Algorithm (EPA) for mask-to-trajectory conversion. Experimental validation in tobacco fields demonstrated 98.4% AP@50 for instance segmentation, with 96.6% precision and 94.2% recall. The model requires 5.1 GFLOPs and 2.7 M parameters, processing individual frames in 2.8 ms, with the complete pipeline executing in 16.99 ms at a resolution of 1920 × 1080.

摘要
本研究旨在开发一种适用于烟草田的稳健作物行检测框架。现有方法无法应对分层收割带来的独特挑战，分层收割会导致植物形态在多个收割周期内逐渐发生变化，从密集的下层冠层结构过渡到稀疏的上层叶片排列。我们开发了 CropRowSeg-YOLO (CRS-YOLO)，这是一个基于 YOLOv11 的实例分割框架，它集成了三项核心创新：利用深度可分离卷积进行空间特征增强的空间增强校准块 (SECB)、采用非对称核进行长程依赖性捕获的分层自适应分割头 (HASH) 以及用于掩码到轨迹转换的增强型后处理算法 (EPA)。在烟草田进行的实验验证表明，实例分割的 AP@50 为 98.4%，精确率为 96.6%，召回率为 94.2%。该模型需要 5.1 GFLOPs 和 2.7M 个参数，处理单个帧需要 2.8 毫秒，整个处理流程在 1920×1080 分辨率下执行时间为 16.99 毫秒。

1. INTRODUCTION

Tobacco harvesting represents one of the most labor-intensive agricultural operations, requiring approximately 450 hours of manual labor per acre with environmental health risks from prolonged exposure to tobacco plants (*Lecours et al., 2012*). The stratified harvesting process demands selective leaf collection across multiple cycles as leaves mature at different rates along the stalk, making mechanization particularly challenging (*Hawks and Collins, 1983*). Autonomous navigation systems have emerged as critical solutions for reducing operator workload and enhancing operational precision in field crop management operations (*Bechar and Vigneault, 2016*). Automatic crop row detection serves as the fundamental determinant of navigation quality, harvest accuracy, and overall system performance in autonomous agricultural machinery. Developing robust crop row detection algorithms specifically adapted to broadleaf crop cultivation environments therefore holds significant practical value for advancing agricultural mechanization and addressing global labor shortage challenges.

Agricultural vision systems for crop row recognition confront three interconnected technical challenges that collectively constrain practical deployment: feature extraction under environmental variability, computational latency in sequential processing pipelines, and annotation scalability for training data generation.

The feature extraction challenge stems from the substantial interfield variability characteristic of agricultural environments; illumination fluctuations, occlusion patterns, and phenotypic diversity create complex visual scenes where crop-background discrimination becomes nontrivial. Early approaches employed

vegetation indices combined with double thresholding based on Otsu's method to achieve crop weed separation, followed by least squares regression for row line fitting (*Montalvo et al., 2012). Zhang et al., (2018*), proposed position clustering algorithms combined with shortest path methods for robust crop row detection in maize fields under varying weed pressure conditions. While computationally efficient, these traditional methods exhibited performance degradation when confronted with curved crop rows, shadow interference, and irregular planting patterns (*Garcia Santillan et al., 2017*). Subsequent deep learning adoption improved robustness through learned feature representations. Semantic segmentation frameworks demonstrated enhanced discrimination capabilities: *Diao et al. (2023)* proposed an improved YOLOv8s network incorporating aurous spatial pyramid pooling faster (ASPPF) structure for accurate corn plant core detection and navigation line extraction. *Li et al. (2023a)* developed morphological anchor point detection methods based on rice stem characteristics to address seedling row identification in paddy field environments. *dos Santos Ferreira et al. (2017)* employed convolutional neural networks for automated weed detection in soybean crops, demonstrating deep learning effectiveness in complex agricultural scenes. Object detection approaches offered alternative formulations: *Ruan et al. (2023)* developed a YOLOR network integrating DBSCAN clustering algorithms for accurate crop row number estimation and individual plant counting in each row. *Yang et al. (2023)* proposed autonomous ROI extraction methods enabling real-time crop row detection in maize fields under varying environmental conditions. *Gong et al. (2021)* introduced root and stalk composite locating point methods for navigation line extraction, achieving precise positioning through multifeatured integration.

The computational efficiency challenge emerges from traditional pipeline architecture, where detection and navigation extraction constitute sequential operations. Classical workflows segment crop regions, extract geometric features, and apply geometric fitting algorithms Hough transform (*Montalvo et al., 2012*), least squares regression (*Zhang et al., 2018*), DBSCAN clustering (*Ruan et al., 2023*) to generate navigation trajectories. This multistage processing introduces cumulative latency incompatible with real-time control requirements in autonomous agricultural machinery. Recent end-to-end frameworks address this limitation through direct parameter regression: *Gai et al. (2021)* employed depth camera based crop row detection and mapping methods for under-canopy navigation of agricultural robotic vehicles, enabling real-time 3D scene understanding. *Guo et al. (2024)* proposed a lightweight YOLOv8sCornNet architecture with depth-wise convolution and PPLCNet modules for efficient corn plant detection and navigation line fitting. *Wang et al. (2023)* introduced improved UNet networks with row attention modules for navigation line extraction in paddy field machines, enhancing spatial feature representation along crop row directions. The E2CropDet framework (*Li et al., 2023b*) modelled individual crop rows as complete and independent detection objects, employing line shaped anchor proposals and contextual aggregation mechanisms to enable end-to-end centerline extraction at 166 FPS inference rates with 5.945 pixels lateral deviation accuracy.

The annotation efficiency challenge represents a practical constraint often overlooked in methodological discussions. End-to-end approaches requiring dense supervision for navigation specific ground truth face substantial annotation costs when scaling across diverse environmental conditions. Manual labeling of trajectory parameters or row centerline coordinates demands significant expert time investment, particularly for datasets spanning multiple growth stages, weather conditions, and field configurations. In contrast, instance segmentation frameworks benefit from semiautomated labelling workflows: foundation models like SAM *(Kirillov et al., 2023*) enable rapid mask generation from simple point or box prompts, requiring minimal manual refinement and substantially reducing per image annotation time. This architectural choice carries implications beyond training efficiency—multistage methods supporting incremental dataset expansion and scenario specific finetuning offer practical advantages for agricultural deployments where environmental diversity necessitates continuous model adaptation.

A critical debate has emerged regarding the relative merits of end-to-end versus multistage methodologies. Proponents of end-to-end learning emphasize conceptual elegance and elimination of handcrafted postprocessing heuristics. However, these approaches face significant practical constraints: extensive manually annotated datasets covering diverse environmental conditions incur substantial annotation investment, and performance may degrade in underrepresented scenarios despite conceptual simplicity. Conversely, while multistage approaches historically suffered from cumulative processing latency in sequential pipelines, recent architectural advances in detection and segmentation technologies have dramatically improved computational efficiency. Modern instance segmentation frameworks (*Ruan et al., 2023; Yang et al., 2023; Li et al., 2023b*) often achieve real-time performance comparable to or exceeding some end-to-end approaches while maintaining interpretable intermediate representations that facilitate debugging and incremental improvement.

Instance segmentation architectures present a compelling trade-off for agricultural navigation applications, balancing detection accuracy, computational efficiency, and deployment practicality. Unlike semantic segmentation producing pixelwise classifications without object individuation, instance segmentation provides explicit crop row delineation enabling sophisticated navigation logic (e.g., interrow selection based on machine positioning, path planning considering row curvature variations). Compared to end-to-end regression frameworks, instance segmentation decouples perceptual and decision-making modules, facilitating independent optimization of detection accuracy and navigation strategy while maintaining flexibility for task specific postprocessing adaptations. Recent advances in anchor free detection architectures (*Yang et al., 2023; Li et al., 2023b*) and efficient segmentation heads further enhance real-time viability, narrowing the inference time gap with direct regression methods while preserving the advantages of intermediate mask representations.

This research proposes CropRowSegYOLO (CRS-YOLO), an instance segmentation framework for tobacco crop row detection and autonomous navigation guidance. The framework integrates three components: the Spatial Enhanced Calibration Block (SECB) for efficient spatial feature enhancement, the Hierarchical Adaptive Segmentation Head (HASH) for capturing crop row spatial characteristics, and the Enhanced Postprocessing Algorithm (EPA) for smooth navigation path generation. Experimental validation achieves 98.4% AP@50 for instance segmentation with real-time processing capability, demonstrating the framework's effectiveness for practical agricultural navigation applications.

2. MATERIALS AND METHODS

2.1 Dataset Construction

The data source for Experimental Validation includes crop images of tobacco cultivated only in Xiangcheng County, Henan Province, China. Xiangcheng is a typical crop production district in Central China and has very similar environmental variables for growing tobacco. Tobacco field data were acquired using a DJI Mavic 2 Pro UAV with Hasselblad camera (4K/30fps, 3axis gimbal stabilization). The UAV operated at 13 meters above canopy height, replicating the viewing perspective of mechanized harvesters to ensure realistic training conditions for autonomous navigation. Video data were captured using the H.264 compression algorithm and systematically temporally subsampled to create static image data sets by taking every 15th frame every two seconds. Images captured in the course of the experiment were chosen to balance scene diversity with processing and annotation tractability.

Scenario 1 Scenario 2

Fig. 1 - Field data collection scenarios

Scenario 1: Monoculture tobacco fields at early harvesting stage with uniform row structure.
Scenario 2: Intercropping systems with additional crops between tobacco rows, introducing increased background complexity.

Field data collection captured two distinct agricultural scenarios representing different stages and management practices in tobacco cultivation. Scenario 1 represents monoculture tobacco fields at early harvesting stage, characterized by uniform row spacing, and minimal interrow vegetation, corresponding to standard mechanized cultivation practices prior to harvest operations. Scenario 2 represents intercropping systems where additional crops are cultivated between tobacco rows, introducing increased background complexity due to the presence of competing vegetation. Data from both scenarios were collected to enhance model generalization capability across diverse field conditions encountered in real world agricultural operations. Ground truth annotation was performed using a semiautomated workflow that integrated LabelMe annotation software with the Segment Anything Model (SAM). SAM was first employed to generate initial segmentation proposals through zero shot instance segmentation. The generated masks were then manually

reviewed and refined using LabelMe's interactive labelling tools, with particular attention to boundary accuracy and instance level crop row delineation. This semiautomated pipeline significantly reduced annotation effort while ensuring high-quality ground truth through manual quality control.

Only complete crop rows were annotated as individual instance masks for the navigation guidance task. One to three centermost rows in each field served as robot navigation references during autonomous traversal. Dataset partitioning employed stratified random sampling, yielding 700 training images (70%), 200 validation images (20%), and 100 test images (10%) with preserved scenario distribution to ensure robust model evaluation across diverse field conditions.

2.2 CRS-YOLO Framework Architecture

This paper proposes CropRowSegYOLO (CRS-YOLO), a specialized instance segmentation framework that extends the YOLOv11 architecture for agricultural crop row detection and navigation line extraction. As illustrated in Figure 2, CRS-YOLO addresses the unique challenges of elongated crop structure detection through a three stage integrated pipeline: enhanced feature extraction, adaptive segmentation, and intelligent postprocessing.

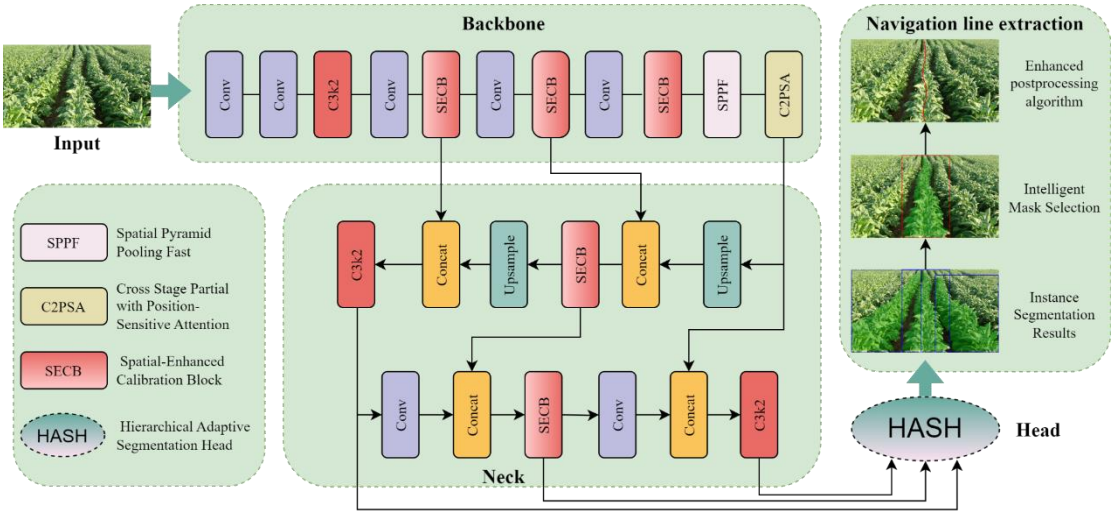

Fig. 2 - Overall structure for CRS-YOLO

The CRS-YOLO framework consists of three main components working in concert. The backbone network employs a hybrid design where standard convolution layers are strategically interspersed with Spatial Enhanced Calibration Blocks (SECB) to capture both local details and spatial relationships crucial for crop row characterization. The neck network utilizes a Feature Pyramid Network (FPN) structure enhanced with SECB modules and Cross Stage Partial with Attention (C2PSA) blocks to achieve effective multiscale feature fusion. The detection head incorporates the proposed Hierarchical Adaptive Segmentation Head (HASH) that simultaneously performs object detection and instance segmentation while avoiding feature conflicts between these dual tasks.

2.2.1 Spatial Enhanced Calibration Block (SECB)

Traditional channel attention mechanisms, such as the Squeeze-and-Excitation (SE) block (*Hu et al., 2018*), have demonstrated significant effectiveness in feature recalibration by explicitly modeling channel interdependencies. However, these mechanisms introduce notable computational overhead and may cause gradient vanishing during backpropagation, particularly when the Sigmoid activation function is employed in the final nonlinearity layer (*Hu et al., 2018*). Moreover, excessive channel recalibration can suppress spatial information critical for accurate instance segmentation of elongated structures such as crop rows. Drawing inspiration from the channel and spatial attention mechanisms in CBFYOLO (*Zhu et al., 2024*), which achieves effective feature enhancement for pest detection in complex environments through its CSEELAN module without excessive computational burden, the Spatial Enhanced Calibration Block (SECB) is introduced. SECB integrates depth-wise separable convolutions for efficient spatial feature extraction (*Howard et al., 2017*) with a lightweight feature calibration mechanism that employs Tanh activation instead of Sigmoid to mitigate gradient vanishing issues. This design prioritizes spatial feature enhancement while maintaining the computational efficiency required for real-time agricultural applications.

SECB retains the core structure of C3k2 but integrates two key components: a spatial enhancement module and a feature calibration module. As illustrated in Figure 3, SECB first applies depth-wise separable convolution to enhance spatial features in the input, then employs a lightweight calibration mechanism to adaptively adjust feature importance based on the enhanced spatial representation.

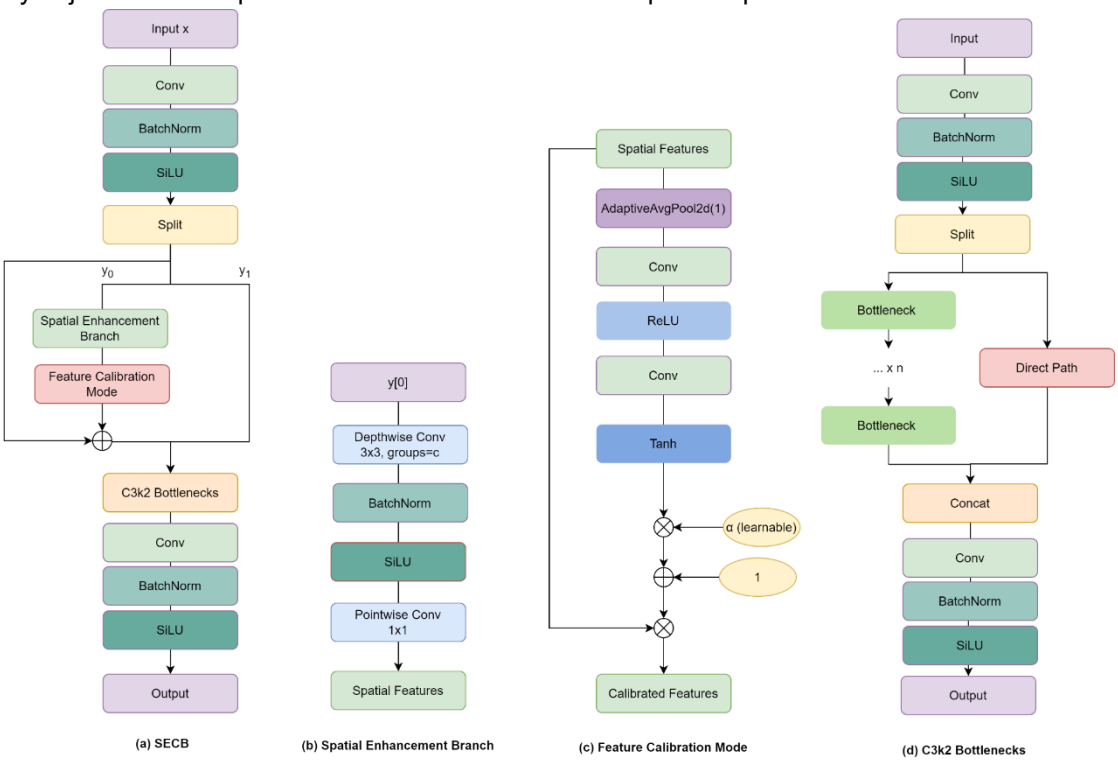

Fig. 3 - Structure for SECB

The formal definition of an input feature map is $X \in \mathbb{R}^{B\times C\times H\times W}$ (where B is batch size, C is channels, H and W are height and width), SECB begins by splitting the features after an initial convolution (cv1 in the code):

$$Y = [Y_0, Y_1] = \text{cv1}(X).chunk(2,1) \tag{1}$$

The spatial enhancement module applies a depth-wise separable convolution to Y_0:

$$S = \text{DWConv}(Y_0, 3, 1, 1) \to \text{BN}(.) \to \text{SiLU}(.) \to \text{PWConv}(., 1) \tag{2}$$

where: DWConv is a depth-wise convolution (groups = C), BN is batch normalization, SiLU is the Sigmoid Linear Unit activation, and PWConv is a pointwise convolution. The feature calibration module uses a lightweight attention mechanism inspired by YOLOv12, employing adaptive average pooling followed by a compressed convolutional path. The calibrated features are then fused with a learnable parameter α (initialized to 0.1) for stability:

$$Calibrated = S \cdot (1 + \alpha \cdot \text{clamp}(Cal, -1, 1)) \tag{3}$$

$$Y_0 = Y_0 + Calibrated \tag{4}$$

The enhanced Y_0 proceeds through standard Bottleneck layers, and the outputs are concatenated and processed by a final convolution (cv2):

$$Output = \text{cv2}(\text{cat}([Y_0, Y_1, \text{Bottlenecks}(Y_0)])) \tag{5}$$

This paper presents a design that preserves the spatial information integrity of the data while introducing only a small computational overhead. This allows for greater ease and efficiency in enhancing the features of interest without any impact on segmentation quality. The application of Tanh actuation to the network guarantees a uniform and balanced flow of gradients, while the incorporation of the clamp operation maintains stability in weight updates during the training process.

2.2.2 Hierarchical Adaptive Segmentation Head (HASH)

Instance segmentation in complex agricultural environments presents significant challenges in balancing multiscale target recognition with computational efficiency. Conventional segmentation heads apply uniform processing across all feature hierarchy levels, failing to exploit the distinct characteristics inherent to features from different semantic depths.

To address this limitation, the Hierarchical Adaptive Segmentation Head (HASH) is proposed, dynamically adjusting processing strategies according to feature hierarchy levels. HASH draws inspiration from two key methodological advances: the efficient residual factorized convolution architecture in ERFNet (*Romera et al., 2018*), which employs asymmetric 1D convolution factorization (3x1 and 1x3 kernels) to reduce computational complexity while maintaining remarkable accuracy; and the depth wise asymmetric bottleneck modules (*Lo et al., 2019*) that utilize asymmetric convolutions for enhanced feature extraction in real-time segmentation. Building upon these foundations, HASH incorporates layer specific processing operations that maximize semantic depth utilization across shallow, intermediate, and deep feature hierarchies. Specifically, deeper layers employ asymmetric kernels (3x1 and 5x1) strategically positioned to optimize computational efficiency for capturing elongated crop row structures, while shallower layers maintain standard convolutions to preserve fine grained spatial details.

The framework employs a dual branch architecture that extends the standard detection segmentation paradigm through layer wise adaptive processors and a collaborative mask prediction mechanism. As demonstrated in Figure 4, the detection branch maintains the conventional YOLO detection head configuration, incorporating a bounding box regression branch (cv2) and a classification branch (cv3). The regression branch performs spatial localization through Distribution Focal Loss (DFL) optimization, while the classification branch utilizes depth-wise separable convolutions for efficient feature extraction and category prediction.

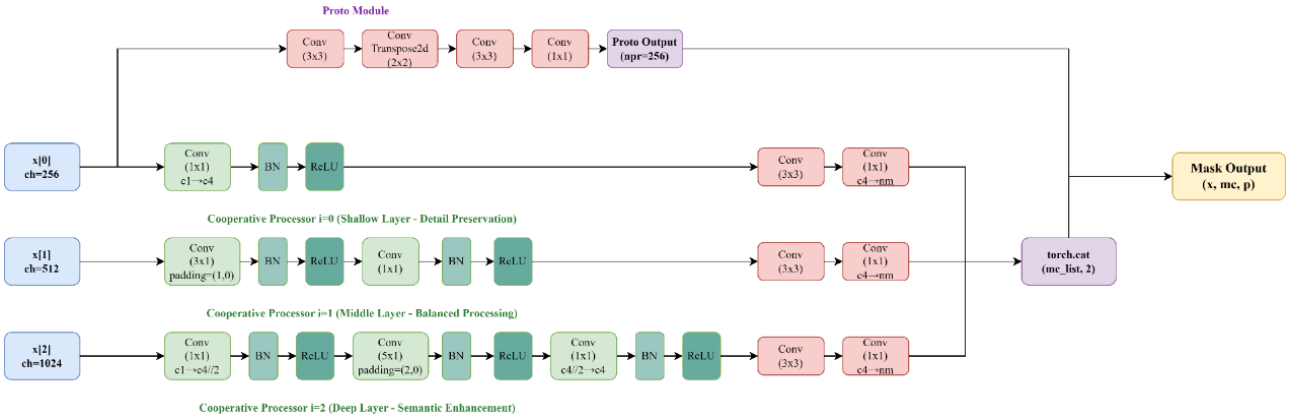

Fig. 4 - Architecture of the prototype mask based segmentation branch

The segmentation branch adopts a prototype mask mechanism, consisting of a prototype generator (Proto) and collaborative feature processors. Unlike traditional methods that treat all feature levels equally, HASH recognizes that shallow features contain rich spatial details, while deep features carry strong semantic information. Consequently, the segmentation branch applies differentiated processing strategies to maximize the utility of features at each level.

Given multilevel feature maps $X = [X_0, X_1, \ldots, X_{nl-1}]$ from the neck (where nl is the number of levels, typically 3), HASH first generates prototypes from the shallowest feature:

$$P = \text{Proto}(X_0) \tag{6}$$

For each level i, a cooperative processor adapts the input features through layer specific transformations. This design philosophy emphasizes the progression from detail preservation in shallow layers to semantic enhancement in deeper layers, ensuring optimal feature utilization across the hierarchical architecture.

For the shallow layer (i = 0), the cooperative processor employs minimal transformation to preserve fine grained details essential for precise boundary delineation. The processor applies a standard 1×1 convolution followed by batch normalization and ReLU activation to adapt channel dimensions while maintaining spatial resolution.

For the intermediate layer (i = 1), the processor implements balanced processing using asymmetric kernels, inspired by multiscale efficient residual factorized convolution units that efficiently capture elongated object features. This approach proves particularly beneficial for crop row detection, as these structures exhibit pronounced elongation characteristics. The processor employs a 3×1 convolution kernel to enhance computational efficiency while preserving spatial context, followed by channel wise refinement through 1×1 convolution with batch normalization and activation.

For the deep layers (i ≥2), the focus shifts to semantic enhancement through larger asymmetric kernels, building upon asymmetric convolutional feature enhancement techniques for agricultural object boundary detection. The processor utilizes 5×1 kernels to capture fine grained elongated structures characteristic of crop rows. The processing pipeline incorporates channel reduction through 1×1 convolution, asymmetric feature extraction via 5×1 convolution, and subsequent channel expansion, with batch normalization and ReLU activation applied after each convolution operation.

The channel dimension is calculated as $C4 = max(X0//4, nm)$, where nm =32 represents the number of mask prototypes. This formulation ensures adequate channel capacity while maintaining computational efficiency.

Mask coefficient prediction follows the cooperative processing stage, where each processed feature map undergoes 3×3 convolution for spatial refinement followed by 1×1 convolution to generate mask coefficients. The final mask coefficients are obtained by concatenating outputs across all processing levels:

$$MC = cat([MC0, \ldots, MCnl - 1], 2) \quad (7)$$

This layered processing strategy ensures optimal utilization of each feature level based on its inherent characteristics. The use of asymmetric kernels (e.g., 3×1, 5×1) in deep layers helps capture long range dependencies while maintaining computational efficiency.

During inference, bounding boxes are decoded using Distribution Focal Loss (DFL) and distance bounding box transformation functions. The final output integrates detection results with mask coefficients MC and prototype masks P. This hierarchical processing strategy ensures optimal contribution from each semantic level to the final segmentation results while preventing interference between different abstraction levels. The strategic use of asymmetric kernels in deeper layers effectively captures long-range dependencies along crop rows while maintaining computational efficiency.

2.3. Enhanced postprocessing algorithm

Traditional instance segmentation methods directly segment all crop rows and perform navigation line generation operations on each detected instance. This approach not only increases inference time but may also incorrectly classify different crop rows as the same category under severe occlusion conditions. The proposed method selects the optimal crop row before navigation line generation, effectively reducing post Latency while improving accuracy.

Figure 5 illustrates the complete workflow of the proposed Enhanced Postprocessing Algorithm (EPA) across four agricultural scenarios, showing the progression from original images through instance segmentation results, intelligent mask selection to identify the most suitable navigation candidate, and finally enhanced postprocessing to generate smooth navigation lines. The EPA framework combines geometric based mask selection with advanced path optimization techniques, ensuring both computational efficiency and navigation accuracy.

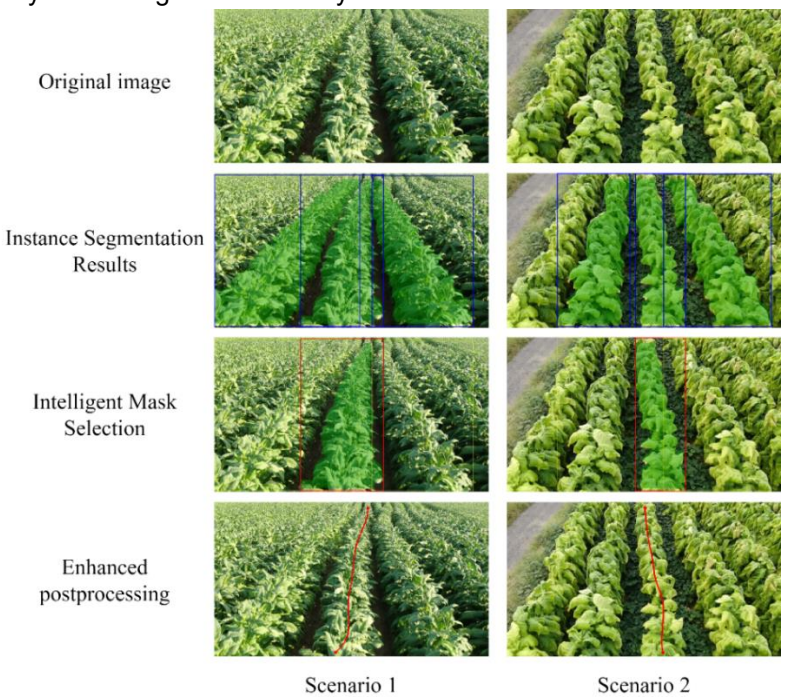

Fig. 5 - Navigation line extraction algorithm pipeline

2.3.1 Intelligent Mask Selection Strategy

Building upon the multi instance detection capabilities of the YOLO model, the intelligent mask selection strategy leverages both bounding box and segmentation mask information to efficiently identify the optimal navigation candidate. Rather than processing all detected crop row instances, this approach significantly improves selection efficiency and accuracy by utilizing YOLO's structured output for rapid decision making based on bounding box geometric properties.

For given candidate mask set $\{M_i\}_{i=1}^{N}$, corresponding bounding box set $\{B_i\}_{i=1}^{N}$, and confidence set $\{C_i\}_{i=1}^{N}$ the algorithm first extracts geometric properties from each bounding box $B_i = (x_1^i, y_1^i, x_2^i, y_2^i)$, including center coordinates and area information. Compared to traditional methods requiring pixelwise mask traversal for calculations, this strategy significantly enhances processing efficiency. The comprehensive scoring mechanism combines three dimensions: spatial position, region size, and detection confidence. The algorithm calculates the distance from bounding box center to image centerline $d_i = \left| bbox\_center\_xi - \frac{W}{2} \right|$, where W represents image width. The improved mask selection scoring formula is:

$$S_i = \alpha \cdot \frac{1}{1+d_i/\tau} + (1-\alpha) \cdot \min\left(\frac{bbox\_area^i}{A_{ref}}, 1\right) \times (0.8 + 0.2 \cdot C_i) \quad (8)$$

This formula introduces detection confidence weight γ based on distance and area evaluation, ensuring priority selection of high confidence detection results. Parameter settings are α = 0.6, τ = 50, A_{ref} = 10000, with constraints d_i < 100 pixels and $bbox\_area^i$ > 500 pixels. After determining the optimal mask, endpoint extraction employs mask geometric property strategies. The algorithm identifies the topmost and bottommost valid rows of the mask, determining endpoint coordinates by calculating the mean of valid pixels in each row. This endpoint localization method ensures path start and end points are positioned at mask geometric centers, guaranteeing path completeness while effectively filtering edge noise interference, establishing a stable foundation for subsequent path optimization.

2.3.2 Navigation Line Extraction Algorithm

After determining the optimal mask through the intelligent selection strategy, the enhanced postprocessing algorithm transforms the selected segmentation mask into a smooth, navigable path. The enhanced postprocessing algorithm combines distance transform and local search optimization to generate smooth navigation paths. The algorithm generates multiple uniformly sampled horizontal lines across the mask, subsequently improving path quality through the optimization of control point positions. For each sampling line, the algorithm first determines the valid pixel range and sets the initial control point at the geometric center position.

To optimize control point positions, the algorithm executes local search within predefined radii, adaptively adjusted according to current row mask width. The optimization objective function comprehensively considers distance transform values and centrality scores:

$$x_{opt}^i = \arg\max_x \beta \cdot D(y_i, x) + (1-\beta) \cdot \frac{1}{1+\left|x - x_{center}^i\right|} \quad (9)$$

where $D(y_i, x)$ represents the distance transform value, and β = 0.7 balances mask centerline proximity with high distance transform values and geometric center approximation. This optimization strategy ensures generated control points are positioned optimally within masks while maintaining relative centrality.

This comprehensive postprocessing pipeline ensures that generated navigation paths are both geometrically accurate and practically smooth, suitable for real-time autonomous navigation applications. The integration of intelligent mask selection with enhanced postprocessing creates an efficient framework that maintains computational efficiency while delivering high quality navigation lines across diverse agricultural scenarios, meeting the stringent requirements of autonomous agricultural vehicle systems.

3. RESULTS AND DISCUSSION

3.1. Parameter settings and model trainings

All experiments were conducted on a platform equipped with Ubuntu 20.04 operating system, utilizing an Intel Core i913900 processor and NVIDIA GeForce RTX 4090 GPU. The software environment included Python 3.10, CUDA version 12.6, and PyTorch version 2.7.1. The proposed CRS-YOLO model

was trained using the following hyperparameters: cache function disabled, image size of 640 pixels, training epochs of 300, batch size of 32, Adam optimizer, initial learning rate of 0.001, and weight decay of 0.0005. These hyperparameters were designed to enhance model convergence and generalization performance, with the Adam optimizer selected for its adaptive learning rate mechanism to effectively address complex feature extraction requirements in instance segmentation tasks.

3.2 Comparative Analysis for Segmentation algorithm

3.2.1 Performance Evaluation Metrics

To comprehensively evaluate CRS-YOLO model performance, this study employed standard object detection and instance segmentation evaluation metrics. For detection accuracy, precision measures the proportion of correctly predicted positive samples among all samples predicted as positive:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

where TP represents true positives and FP represents false positives. Recall measures the proportion of correctly predicted positive samples among all actual positive samples:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

Average Precision measures model performance by calculating comprehensive performance of precision and recall under different thresholds, where $AP@50$ indicates average precision at IoU threshold of 0.5, and $AP@5095$ represents average precision across IoU thresholds ranging from 0.5 to 0.95.

$$AP = \int_0^1 P(R)dR \tag{12}$$

For computational efficiency evaluation, this study employed four key indicators to quantify model deployment feasibility. GFLOPs represents the number of floating point operations executed per second by the model, reflecting computational complexity and theoretical computational load. Parameter count measures the number of parameters in the model in millions, directly reflecting storage requirements and memory usage. Frames Per Second (FPS) indicates the number of frames the model can process per second, serving as an important indicator for evaluating real-time performance and holding critical significance for agricultural automation applications. Latency measures the time required to process each frame from input to output in ms, reflecting the model's frame-by-frame response capability.

3.2.2 Comparative Analysis with Segmentation Methods

Reported in Table 1 is extensive performance, demonstrating that CRS-YOLO is effective for detection and segmentation tasks, achieving 98.4% AP@50 across tasks and the highest performance among the compared nanoscale architectures. It shows 96.7% precision, 93.7% recall and 94.3% AP@5095 on detection and 96.6% precision, 94.2% recall and 89.3% AP@5095 on segmentation.

Table 1

Performance comparison of object detection and instance segmentation models

Model	GFLOPs	Params (M)	Box				Mask			
			P	R	AP@50	AP@5095	P	R	AP@50	AP@5095
YOLOv5n	4.9	2.4	93.8	89.1	95.8	86.2	92.1	91.2	96.6	81.3
YOLOv8n	5.4	2.9	88.1	93.9	95.9	87.5	91.8	91.7	96.1	82.6
YOLOv11n	5.2	2.8	96.1	93.2	97.4	92.5	96.0	93.7	97.8	87.6
YOLOv12n	5.0	2.8	93.7	95.0	97.6	91.8	93.7	95.0	97.2	85.3
Ours	5.1	2.7	96.7	93.7	98.4	94.3	96.6	94.2	98.4	89.3

It was found that combining speed optimizations provides a balanced improvement in both performance and efficiency. CRS-YOLO needs 5.1 GFLOPs and 2.7M parameters nearly identical to YOLOv12n at 5.0 GFLOPs and 2.8M parameters and much lighter than YOLOv11n at 5.2 GFLOPs with 2.8M parameters. Despite this compact FPS friendly architecture, CRS-YOLO is better at AP@50 on both tasks: segment wise it achieves 98.4% where YOLOv11n gets 97.8% and YOLOv12n gets 97.2%; that's 0.6% better than YOLOv11n and 1.2% better than YOLOv12n. Detection AP@50 also shows gains of 0.8–1.0% over these models, with 98.4% compared to 97.4% and 97.6% for YOLOv11n and YOLOv12n, respectively.

Table 2

Performance comparison with semantic segmentation methods

Model	GFLOPs	Params(M)	P	R	Fps	Latency (ms)
Zhang et al.'s	218.3	67.1	92.5	92.1	79.6	12.5
Diao et al.'s	35	19.3	93.4	92. 7	40.6	24.6
VIT	5.4	24.9	85.1	83.8	201.7	4.9
Ours	5.1	2.7	96.6	94.2	356.7	2.8

Semantic segmentation performance analysis in Table 2 establishes CRS-YOLO's exceptional efficiency compared to established benchmarks. The model achieves outstanding results with 96.6% precision, 94.2% recall, and impressive real-time performance with only 2.8 ms latency. Computational complexity analysis reveals significant optimization, with CRS-YOLO requiring only 5.1 GFLOPs compared to 35 GFLOPs for Diao et al.'s method and 218.3 GFLOPs for Zhang et al.'s method, while simultaneously reducing parameter count to 2.7 million. Performance improvements include 3.2% precision enhancement over Diao et al.'s method and more than eightfold FPS acceleration compared to Zhang et al.'s approach. Comparative analysis with Vision Transformer (VIT) approaches demonstrates CRS-YOLO's superior accuracy efficiency balance: while VIT achieves 201.7 FPS, CRS-YOLO substantially outperforms it with 11.5% higher precision and 10.4% higher recall, while maintaining comparable computational overhead and utilizing significantly fewer parameters.

3.2.3 Ablation Study Analysis

CRS-YOLO's performance improvements stem from innovative structural design, including SECB (Spatial Enhanced Calibration Block) and HASH (Hierarchical Adaptive Segmentation Head) modules, rather than simply reducing depth or width. To validate these components, an ablation study was conducted, as shown in Table 3.

Table 3

Ablation study results for SECB and HASH modules in object detection and instance segmentation tasks

SECB	HASH	GFLOPs	Params (M)	Box				Mask			
				P	R	AP@50	AP@5095	P	R	AP@50	AP@5095
✗	✗	9.42	2.84	97.0	93.1	97.3	92.4	95.8	93.4	97.4	87.
✓	✗	9.55	2.9	94.6	94.6	97.4	92.2	95.6	95.6	97.8	87.3
✗	✓	9.08	2.74	95.9	94.6	98.0	92.8	96.4	95.1	98.2	87.7
✓	✓	9.21	2.79	96.7	93.7	98.4	94.3	96.6	94.2	98.4	89.3

The baseline without SECB or HASH achieved 97.09% precision and 97.3% AP@50 in detection, and 97.4% AP@50 in segmentation, with 9.42 GFLOPs. Adding SECB alone reduced GFLOPs to 9.55 but slightly decreased detection AP@50 to 97.48%, indicating improved spatial feature extraction but possible over compression. HASH alone improved detection AP@50 to 98% and segmentation to 98.2% with 9.08 GFLOPs, highlighting its role in multiscale attention. Combining both modules (complete CRS-YOLO) achieved optimal results: detection AP@50 of 98.3%, segmentation of 98.4%, with AP@5095 improving to 94.2% and 89.2% respectively, and GFLOPs of 9.21, representing a 2.3% reduction in GFLOPs compared to baseline while improving detection AP@50 by 1.1%. This confirms that synergistic integration of SECB and HASH achieves better feature fusion and efficiency without relying on deeper networks.

3.3 Navigation Line Extraction Algorithm Experiment

To validate the effectiveness and robustness of the proposed CRS-YOLO model in practical tobacco field navigation line extraction tasks, comprehensive comparative experiments were designed covering multiple complex scenarios. This study selected four typical tobacco field states as test scenarios. The selected scenarios encompass the primary influencing factors of field environments in actual agricultural production, including the impact of plant density on visual environments, the effects of different harvesting methods on field morphology, and the influence of lighting conditions on scene recognition.

3.3.1 Evaluation metrics

A comprehensive evaluation system was established containing four core indicators: smoothness score, center deviation, path efficiency, and Latency to thoroughly assess navigation line extraction algorithm performance. The smoothness score reflects navigation path continuity and stability, calculated by quantifying the standard deviation of angular changes between adjacent path segments:

$$S_{smooth} = \frac{1}{1+\sigma_\theta} \tag{13}$$

where σ_θ represents the standard deviation of angles between adjacent path segments. Values closer to 1 indicate smoother paths. Center deviation measures the degree of deviation of extracted navigation lines relative to reference straight lines:

$$D_{center} = \frac{1}{N_{cd}} \sum_{i=1}^{N} || p_i - p_i' || \tag{14}$$

where p_i is the i −th point on the navigation line, p'_i is the projection point on the reference straight line, and N_{cd} is the total number of sampling points. Smaller values indicate navigation lines closer to ideal straight line paths. Path efficiency is defined as the ratio of reference straight line distance to actual path length:

$$E_{path} = \frac{L_{baseline}}{L_{actual}} \tag{15}$$

where $L_{baseline}$ is the straight line distance from start to end points, and L_{actual} is the cumulative length of the actual navigation path. Values closer to 1 indicate more efficient paths. Latency records the complete computational time from input to output in ms, directly reflecting algorithm real-time performance.

3.3.2 Comparative Analysis of Navigation Algorithms

This study implemented and compared five navigation line extraction algorithms: the A* algorithm employs heuristic search to find optimal paths through cost weighted distance transform maps, the sliding window method achieves rapid tracking through localized centroid calculations, the vertical projection method uses row direction projection features to extract center lines, the K-Means clustering method identifies primary paths through spatial clustering, and the enhanced postprocessing method achieves precise localization by combining distance transform with local optimization. To comprehensively evaluate algorithm performance, systematic testing was conducted on 100 images from the test set, where Scenario 1 represents standard monoculture tobacco fields with uniform row spacing, and Scenario 2 simulates a complex intercropping environment with potatoes planted in furrows between tobacco rows to simulate weeds. After segmenting each image using the improved YOLOv11 model, all five navigation algorithms were executed on each segmentation mask, and performance metrics were recorded. Based on the collected data, three types of visualizations were generated: box plots illustrating the performance distribution characteristics and statistical stability of each algorithm; a comparison table presenting average values across four metrics; and scenario-specific heatmaps showing normalized performance under different scenarios. Figure 6 presents navigation line extraction visualization results from various algorithms under four typical scenarios.

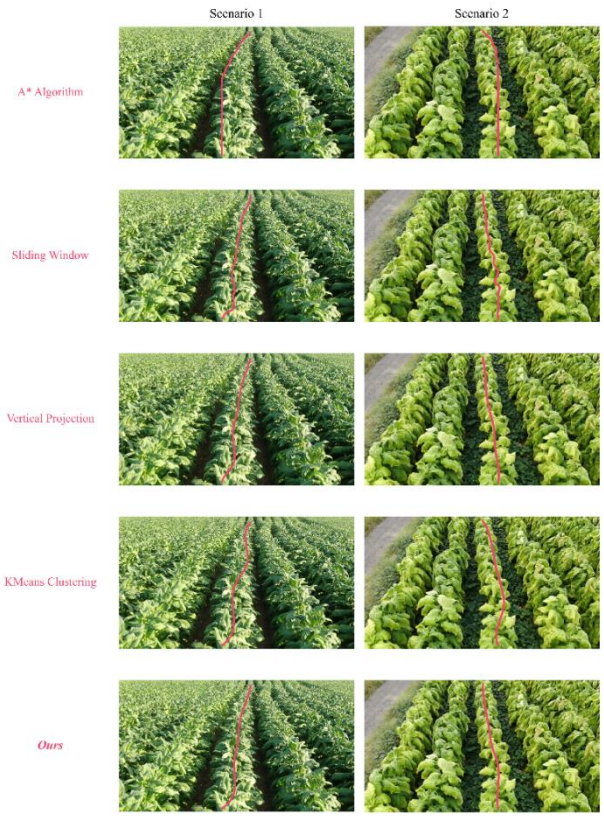

Fig. 6 - Navigation line extraction results under four typical scenarios

Table 4

Performance Comparison of Navigation Line Extraction Algorithms

Method	Smoothness Score ↑	Center Deviation (px)↓	Path Efficiency ↑	Latency (ms)↓
A Star Algorithm	0.934	20.12	0.975	267.73
Sliding Window	0.901	**1.10**	0.955	2.30
Vertical Projection	0.961	4.37	0.972	0.25
K-Means Clustering	0.927	20.04	0.911	93.72
Enhanced Post processing	**0.977**	4.12	**0.990**	1.77

Table 4 summarizes the average performance metrics of each algorithm. The enhanced postprocessing method achieved an average smoothness score of 0.977, substantially outperforming other methods ranging from 0.901 to 0.961. Center deviation was only 4.12 pixels, representing improvements of 79.5% and 79.4% compared to the A* algorithm's 20.12 pixels and K-Means clustering's 20.04 pixels, respectively. Path efficiency reached 0.990, approaching the theoretical optimum of 1.000, while average latency was only 1.77 ms, significantly outperforming the A* algorithm's 267.73 ms and K-Means clustering's 93.72 ms, while remaining competitive with the vertical projection method's 0.25 ms. For autonomous navigation systems requiring operation at 30 frames per second with a single-frame budget of approximately 33 ms, the processing times of the A* algorithm and K-Means clustering completely fail to meet real-time application requirements.

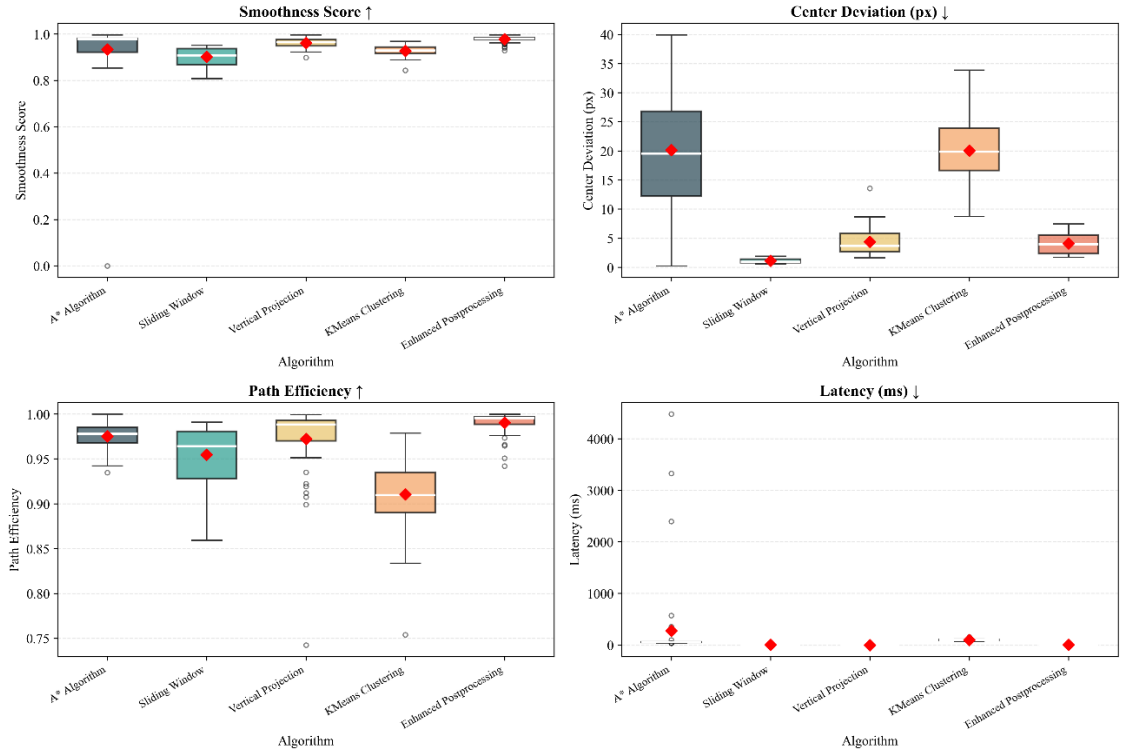

Fig. 7 - Box plot distribution of algorithm performance metrics

Box plots in Figure 7 reveal deeper characteristics of algorithm performance distributions. The A* algorithm's navigation paths clearly deviated from the valid crop row range in multiple scenarios, exhibiting unreasonable situations such as paths crossing crop plants or entering furrow edges. Box plots show its median center deviation exceeding 20 pixels with large interquartile ranges, combined with processing latency exceeding 250 ms, indicating this algorithm is unsuitable for this application scenario. Although the sliding window method achieved a median center deviation of only 1.10 pixels, this reflects an overtracking characteristic. The method strictly follows the mask centroid in each local region, resulting in paths containing excessive inflection points and high frequency oscillations, with a smoothness score of only 0.901, the lowest among all algorithms. Such high curvature paths would cause frequent directional adjustments in agricultural machinery, making it unsuitable as a navigation reference line.

The vertical projection method demonstrated good average performance with processing latency of only 0.25 ms, but box plots revealed significant outliers in center deviation and path efficiency dimensions, indicating severe performance degradation in certain specific scenarios. The K-Means clustering method's latency distribution exhibited longtail behavior, with some samples exceeding 100 ms. Extreme outliers far from the box appeared in center deviation and path efficiency dimensions, indicating the method cannot stably extract navigation lines in complex environments.

The enhanced postprocessing method demonstrated comprehensive and stable advantages in box plots, with all four performance dimensions showing compact interquartile ranges, high medians, and virtually no outliers. The smoothness score box was compactly distributed above 0.97, center deviation remained below 5 pixels with minimal variability, path efficiency approached 1.0, and processing latency remained stable below 2 ms. This consistency stems from the organic integration of distance transform global constraints and local optimization. Distance transform provides robust identification of mask central regions, avoiding interference from local noise, while local search optimization ensures paths maintain smooth transitions while satisfying center constraints.

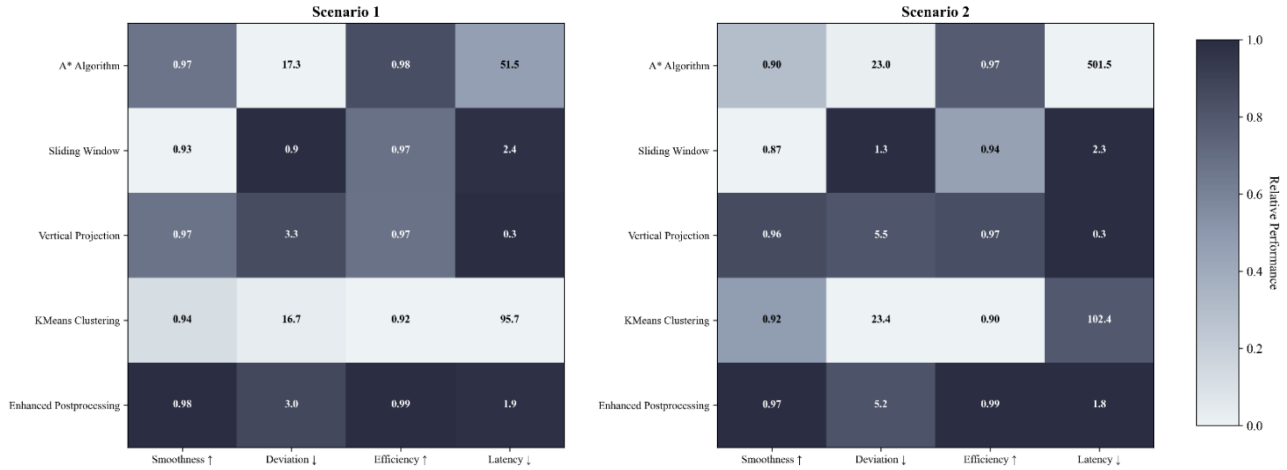

Fig. 8 – Cross-scenario performance heatmap of navigation algorithms

The scenario specific heatmaps are shown in Figure 8 and illustrate how environmental adaptability is incorporated into the algorithms. The heatmaps use a gray-blue gradient format where deep blue signifies peak performance, normalized scores nearing 1.0, and light grey represents the lowest level of performance. A review of scenario performances showed that for most algorithms there was decreased performance in center deviation and smoothness metrics between scenario 1 and scenario 2 (A* algorithm and K-Means clustering latencies fell to 501.5 ms and 102.4 ms further indicating their sensitivity to scenario complexity). Vertical projection maintained low latency but significantly decreased the center deviation metrics in scenario 2, and the postprocess method produced dominant deep blue areas for both scenarios, with scenario 1 having smoothness values of 0.98, center deviation of 3 pixels, path efficiency of 0.99, and a latency of 1.9 ms, and scenario 2 having values of 0.97, 5.2 pixels, 0.99, and 1.8 ms, respectively. The intercropping environment in scenario 2 caused a slight increase in center deviation, but it was significantly smaller than increases seen with the other three metrics remaining basically unchanged or slightly improved over scenario 1. When evaluated in multiple scenarios, the algorithm exhibited a high degree of robustness through its ability to maintain stable performance. The basis for this performance can be attributed to the application of distance transformation methods to provide global constraints in locating accurately the center of crop rows. This approach has also provided further adaptability to the system through a variable local optimization radius that automatically adjusts according to the width of the mask being processed, enabling the algorithm to effectively respond to variations in row density and curvature.

3.4 Integrated Performance Evaluation

The study has carried out performance benchmarking of the algorithms used in this research by performing benchmarking with video sequences at 30FPS and testing through 2 different operational contexts and also measuring the performance from acquisition all the way through calculating navigational paths for these have been included in all metrics. The test procedure consisted of 200 frame, 1280 x 720 test sequence video clips and the results of comparing how perform on CPU only compared to GPU only are shown in Table 5.

Table 5

Performance Metrics Across Different Scenarios

Scenarios	GPU		CPU	
	FPS	Latency (ms)	FPS	Latency (ms)
1	83.8	11.93	18.28	54.71
2	99.4	10.06	18.08	55.32

The results show that the GPU implementation results in a much higher throughput, resulting in an average of 94.6 FPS and 10.64 ms of processing delay. These performance metrics are consistent with the demands of real-time applications that use the algorithms to control autonomous vehicles and that control precision will be adversely affected if there is a higher time lag. The CPU implementations can only achieve approximately 18 FPS with an average processing delay of 53 ms. These results show that while the system can be implemented on various hardware, the computational efficiency of each of those hardware platforms will vary significantly based on the processing architecture chosen.

Table 6

Performance Metrics Across Different Resolutions

Resolution	GPU		CPU	
	FPS	Latency (ms)	FPS	Latency (ms)
720×480	116.58	8.58	36.75	27.21
1280×720	93.44	10.7	18.25	54.8
1920×1080	58.86	16.99	7.47	133.86

Performance metrics in relation to resolution in Table 6, show different areas of importance for the flexibility of deployment. GPU accelerated system configuration, which can not only achieve a high Frames Per Second rate, across all resolutions, but also can be used to provide sufficient responsiveness for real-time agricultural applications where there is an immediate need for response. The CPU based processing methods, as expected, do display an inverse correlation to the resolution, which is the expected behavior in terms of the number of frames processed by the system's CPU per second as a function of the resolution, with 720 x 480 resolution leading to 36.75 frames per second being processed, which decreases to 7.47 fps for the 1920 x 1080 resolution, and is due to the higher pixel density being processed, resulting in an increase in the number of calculations needed to produce the same number of frames. Despite the reduction of frame processing at higher resolutions, the system will be functionally operational in all configurations tested, providing real world opportunities for deployment, from low resource to high resource agricultural computing systems, to be driven by specific applications.

3.5 Limitations and future work

Several limitations in the current implementation warrant acknowledgment and provide directions for future research. Data generalizability represents a primary constraint, as the training dataset was collected exclusively from two regions in Henan Province across four operational scenarios. This geographical limitation may restrict model performance in tobacco growing regions with different planting densities, climatic conditions, or cultivation practices. Additionally, the dataset size of 600 images, while sufficient for proof of concept validation, remains relatively small for comprehensive deep learning model training across diverse agricultural environments.

Crop specific optimization constitutes another significant limitation. The SECB and HASH modules, despite their generalizable design principles, incorporate parameter settings and processing strategies specifically optimized for tobacco plant morphology and growth patterns. Direct application to other row crops with substantially different structural characteristics, such as corn or soybeans, would necessitate architectural modifications and extensive retraining procedures.

Algorithmic assumptions in the Enhanced Postprocessing Algorithm impose constraints on field topology compatibility. The current implementation assumes relatively linear crop row arrangements, which may not adequately handle irregular field topography, terraced agricultural areas, or nonstandard planting patterns commonly encountered in mountainous or sloped terrain.

Future research directions should address these limitations through several strategic approaches. Dataset expansion across multiple geographical regions, tobacco varieties, and seasonal

conditions would enhance model robustness and generalizability. Crosscrop adaptation studies could investigate transfer learning strategies to extend the framework's applicability to diverse agricultural crops while minimizing retraining requirements. Algorithmic enhancements should incorporate adaptive curve fitting mechanisms capable of handling nonlinear crop row arrangements and irregular field geometries. Additionally, multimodal integration combining RGB imagery with depth sensors or LiDAR data could improve detection reliability under challenging environmental conditions, while real-time optimization research could further reduce computational overhead for deployment on resource constrained agricultural machinery.

4. CONCLUSIONS

To address the high annotation costs of end-to-end models and the prolonged processing times of traditional multistage navigation line extraction methods in tobacco layered harvesting scenarios, this paper developed CropRowSegYOLO (CRS-YOLO), an efficient crop row segmentation framework based on improved YOLOv11. The framework integrates the Spatial Enhanced Calibration Block (SECB) and Hierarchical Adaptive Segmentation Head (HASH) to enhance spatial feature representation and mitigate stratified harvesting interference. The Enhanced Postprocessing Algorithm (EPA) converts segmentation masks into smooth navigation paths through distance transform optimization and B-spline interpolation.

Experimental validation demonstrates that CRS-YOLO achieved 98.4% AP@50 and 89.3% AP@5095 for instance segmentation, with 96.6% precision and 94.2% recall. The model requires 5.1 GFLOPs and 2.79 million parameters, processing individual frames in 2.8 ms. For navigation line extraction, the EPA framework achieved 0.977 smoothness score, 4.12pixel center deviation, and 0.990 path efficiency with 1.77 ms processing latency. Comparative analysis across five algorithms showed that EPA reduced center deviation by 79.5% and 79.4% compared to A* algorithm and K-Means clustering respectively, while maintaining computational efficiency competitive with projection based methods. Cross scenario evaluation confirmed stable performance across both standard monoculture fields and complex intercropping environments.

This study establishes a practical framework for navigation line extraction in tobacco layered harvesting scenarios. The framework can also serve as an annotation tool for end-to-end navigation algorithms. Future research could explore semi-supervised learning approaches to develop end-to-end navigation line extraction systems for agricultural applications.

ACKNOWLEDGEMENT

This work was supported by “Development and application of fully automatic layered harvesting machine in hilly and mountainous areas” of China National Tobacco Corporation tobacco agricultural machinery research project (110202301010), "Research and Application of Integrated Operation Machine for Topping, Bud Suppression, and Pesticide Spraying in Mountainous Areas" of Science and Technology Project of China National Tobacco Corporation Henan Company (2023410000240029), and State key laboratory of intelligent agricultural power equipment open project "Hydraulic control system design of hydraulic drive tracklaying vehicle"(SKLIAPE 2024011).

REFERENCES

[1] Bechar, A.; Vigneault, C. (2016). Agricultural robots for field operations: Concepts and components. *Biosystems Engineering*. 149, 94-111. https://doi.org/10.1016/j.biosystemseng.2016.06.014

[2] Diao, Z., Guo, P., Zhang, B., Zhang, D., Yan, J., He, Z., Zhao, S., Zhao, C. (2023). Navigation line extraction algorithm for corn spraying robot based on improved YOLOv8s network. *Computers and Electronics in Agriculture*. 212, 108049. https://doi.org/10.1016/j.compag.2023.108049

[3] dos Santos Ferreira, A., Freitas, D.M., da Silva, G.G., Pistori, H., Folhes, M.T. (2017). Weed detection in soybean crops using ConvNets. *Computers and Electronics in Agriculture*. 143, 314-324. https://doi.org/10.1016/j.compag.2017.10.027

[4] Gai, J., Xiang, L., Tang, L. (2021). Using a depth camera for crop row detection and mapping for under-canopy navigation of agricultural robotic vehicle. *Computers and Electronics in Agriculture*. 188, 106301. https://doi.org/10.1016/j.compag.2021.106301

[5] Garcia-Santillan, I.D., Montalvo, M., Guerrero, J.M., Pajares, G. (2017). Automatic detection of curved and straight crop rows from images in maize fields. *Biosystems Engineering*. 156, 61-79.

https://doi.org/10.1016/j.biosystemseng.2017.01.013

[6] Gong, J., Wang, X., Zhang, Y., Lan, Y., Mostafa, K. (2021). Navigation line extraction based on root and stalk composite locating points. *Computers and Electronics in Agriculture*. 92, 107115. https://doi.org/10.1016/j.compeleceng.2021.107115

[7] Guo, P., Diao, Z., Zhang, B., Zhang, D., Yan, J. (2024). Navigation line extraction algorithm for corn spraying robot based on YOLOv8s-CornNet. *Journal of Field Robotics*. 41, 1485-1501. https://doi.org/10.1002/rob.22360

[8] Hawks, S.N., Collins, W.K. (1983). *Principles of Flue-Cured Tobacco Production*. N.C. State University: Raleigh, NC, USA.

[9] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv*. arXiv:1704.04861. https://doi.org/10.48550/arXiv.1704.04861

[10] Hu, J., Shen, L., Sun, G. (2018). Squeeze-and-Excitation Networks. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 7132-7141. https://doi.org/10.1109/CVPR.2018.00745

[11] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollar, P., Girshick, R. (2023). Segment Anything. *In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, 2023, pp. 4015-4026. https://doi.org/10.1109/ICCV51070.2023.00371

[12] Lecours, N., Almeida, G.E.G., Abdallah, J.M., Novotny, T.E. (2012). Environmental health impacts of tobacco farming: A review of the literature. *Tobacco Control*. 21, 191-196. https://doi.org/10.1136/tobaccocontrol-2011-050318

[13] Li, B., Liu, C., Wang, J., Kang, S., Li, D. (2023). Rice seedling row detection based on morphological anchor points of rice stems. *Biosystems Engineering*. 226, 71-85. https://doi.org/10.1016/j.biosystemseng.2022.12.012

[14] Li, D., Li, B., Kang, S., Feng, H., Long, S., Wang, J. (2023). E2CropDet: An efficient end-to-end solution to crop row detection. *Expert Systems with Applications*. 227, 120345. https://doi.org/10.1016/j.eswa.2023.120345

[15] Lo, S.Y., Hang, H.M., Chan, S.W., Lin, J.J. (2019). Efficient Dense Modules of Asymmetric Convolution for Real-Time Semantic Segmentation. *In Proceedings of the ACM Multimedia Asia*, Beijing, China, 15-18 December 2019, pp. 1-6. https://doi.org/10.1145/3338533.3366558

[16] Montalvo, M., Pajares, G., Guerrero, J.M., Romeo, J., Guijarro, M., Ribeiro, A., Ruz, J.J., Cruz, J.M. (2012). Automatic detection of crop rows in maize fields with high weeds pressure. *Expert Systems with Applications*. 39, 11889-11897. https://doi.org/10.1016/j.eswa.2012.02.117

[17] Romera, E., Alvarez, J.M., Bergasa, L.M., Arroyo, R. (2018). ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation. *IEEE Transactions on Intelligent Transportation Systems*. 19, 263-272. https://doi.org/10.1109/TITS.2017.2750080

[18] Ruan, J., Wang, Y., Yang, D., Li, H., Hu, J. (2023). Crop row detection method for unmanned agricultural machinery based on YOLO-R. *Biosystems Engineering*. 232, 1-12. https://doi.org/10.1016/j.biosystemseng.2023.06.010

[19] Wang, Q., Liu, F., Zeng, Y., Liu, Z. (2023). Navigation line extraction for paddy field machines using improved U-Net with row attention module. *Computers and Electronics in Agriculture*. 210, 107911. https://doi.org/10.1016/j.compag.2023.107911

[20] Yang, Y., Zhou, Y., Yue, X., Zhang, G., Wen, X., Ma, B., Xu, L., Chen, L. (2023). Real-time detection of crop rows in maize fields based on autonomous extraction of ROI. *Expert Systems with Applications*. 213, 118826. https://doi.org/10.1016/j.eswa.2022.118826

[21] Zhang, X., Li, X., Zhang, B., Zhou, J., Tian, G., Xiong, Y., Gu, B. (2018). Automated robust crop-row detection in maize fields based on position clustering algorithm and shortest path method. *Computers and Electronics in Agriculture*. 154, 165-175. https://doi.org/10.1016/j.compag.2018.09.014

[22] Zhu, L., Li, X., Sun, H., Han, Z. (2024). Research on CBF-YOLO Detection Model for Common Soybean Pests in Complex Environment. *Computers and Electronics in Agriculture*. 216, 108515. https://doi.org/10.1016/j.compag.2023.108515