processes

MDPI

Article

Development of an Optimized YOLO-PP-Based Cherry Tomato Detection System for Autonomous Precision Harvesting

Xiayang Qin [1], Jingxing Cao [2], Yonghong Zhang [1,*], Tiantian Dong [1] and Haixiao Cao [1]

[1] School of Automation, Nanjing University of Information Science and Technology, Nanjing 210044, China; xiayangqin@nuist.edu.cn (X.Q.); 202211180004@nuist.edu.cn (T.D.); 202491000009@nuist.edu.cn (H.C.)
[2] Wuxi Siasun Robot and Automation Company Limited, Wuxi 214101, China; caojingxing@siasun.com
* Correspondence: zyh@nuist.edu.cn

Abstract: An accurate and efficient detection method for harvesting is crucial for the development of automated harvesting robots in short-cycle, high-yield facility tomato cultivation environments. This study focuses on cherry tomatoes, which grow in clusters, and addresses the complexity and reduced detection speed associated with the current multi-step processes that combine target detection with segmentation and traditional image processing for clustered fruits. We propose YOLO-Picking Point (YOLO-PP), an improved cherry tomato picking point detection network designed to efficiently and accurately identify stem keypoints on embedded devices. YOLO-PP employs a C2FET module with an EfficientViT branch, utilizing parallel dual-path feature extraction to enhance detection performance in dense scenes. Additionally, we designed and implemented a Spatial Pyramid Squeeze Pooling (SPSP) module to extract fine features and capture multi-scale spatial information. Furthermore, a new loss function based on Inner-CIoU was developed specifically for keypoint tasks to further improve detection efficiency.The model was tested on a real greenhouse cherry tomato dataset, achieving an accuracy of 95.81%, a recall rate of 98.86%, and mean Average Precision (*mAP*) scores of 99.18% and 98.87% for mAP50 and mAP50-95, respectively. Compared to the DEKR, YOLO-Pose, and YOLOv8-Pose models, the mAP value of the YOLO-PP model improved by 16.94%, 10.83%, and 0.81%, respectively. The proposed algorithm has been implemented on NVIDIA Jetson edge computing devices, equipped with a human–computer interaction interface. The results demonstrate that the proposed Improved Picking Point Detection Network exhibits excellent performance and achieves real-time accurate detection of cherry tomato harvesting tasks in facility agriculture.

Keywords: keypoint detection; YOLO v8; tomato detection; facility agriculture; attention mechanism; deep learning

check for updates

Academic Editor: Ján Piteľ

Received: 9 December 2024
Revised: 15 January 2025
Accepted: 24 January 2025
Published: 27 January 2025

Citation: Qin, X.; Cao, J.; Zhang, Y.; Dong, T.; Cao, H. Development of an Optimized YOLO-PP-Based Cherry Tomato Detection System for Autonomous Precision Harvesting. *Processes* **2025**, *13*, 353. https://doi.org/10.3390/pr13020353

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The fruit and vegetable industry, as the largest and fastest-growing sector in global agricultural production, holds a significant position in the agricultural economy. According to the Food and Agriculture Organization of the United Nations [1], China was the largest tomato producer in the world in 2022, with a production of 68 million tons, accounting for 36.7% of the global tomato yield. In recent years, there has been a notable increase in the number of intelligent tomato production greenhouses employing vertical cultivation systems, also known as Facility Cultivation Systems [2], across China. Among these, cherry tomatoes, as a representative of cluster-type tomatoes, have seen a consistent increase in production share due to their high value. The modern tomato production system is

characterised by high yields and short growing cycles. However, field research indicates that harvesting tasks in most production bases still rely on manual labour. With the diversification of the socio-economic development, the agricultural labour force continues to migrate to other industries, leading to an increasing shortage of human resources in agricultural production. This structural change not only causes widespread labour problems in agricultural production but also drives up labour costs [3]. Against this background, the development of efficient automated fruit and vegetable harvesting robots to enhance the level of agricultural intelligence has become an urgent necessity to solve the current agricultural labour dilemma.

In recent years, breakthrough advancements in machine vision and agricultural machinery technology [4–6] have significantly enhanced the operational efficiency, functional diversity, and remote interaction capabilities of harvesting robots in complex agricultural environments. However, due to the clustered growth pattern of cherry tomatoes, harvesting them poses greater challenges compared to ordinary tomatoes, making the issues caused by agricultural labour shortages even more pronounced. Specifically, this introduces two key problems relevant to the field of agricultural automation. Firstly, there is a need for robotic structures that can efficiently harvest both cluster-type and individual tomatoes [7,8]. Most existing tomato harvesting robots use multi-finger grippers to pick individual tomatoes. However, in current practices, clustered fruits such as grapes and cherry tomatoes are typically harvested in bunches by cutting the stems to preserve their integrity. To accommodate this, some of the latest harvesting robots have adopted cutting-type end effectors. Secondly, there is a lack of research on efficient cluster-type tomato picking point detection, which is essential for automated harvesting robots [9]. This study primarily focuses on addressing this latter issue.

We have observed that many researchers have employed object detection networks to address similar tomato harvesting detection tasks. However, most strategies treat object detection as a preliminary step in the overall process, where regions of interest are first identified and then combined with post-processing steps such as semantic segmentation to obtain the final localization points. In contrast to these stem detection algorithms, this study focuses on individual cherry tomatoes in facility cultivation systems, specifically exploring the feasibility of using keypoint detection algorithms within the YOLO framework to accomplish the harvesting point detection task. This single-task direct detection approach not only simplifies the detection process and avoids error accumulation from multi-step processing but also enables faster and more precise localization of harvesting points, providing a more efficient solution for real-time automated harvesting. Furthermore, this method boasts high computational efficiency, making it well-suited for embedded computing platforms and fully meeting the real-time requirements for rapid robotic operations in facility agriculture environments. This study aims to address the challenges in cherry tomato recognition and determine the optimal picking point locations. Moreover, the robotic arm recognition software has been developed based on the existing harvesting equipment and scenarios to provide support for subsequent research. The principal contribution of this study can be summarised as follows:

(1) A dataset for target recognition and picking point detection, focusing on single cherry tomatoes, was collected and annotated at a tomato planting base. The feasibility of the YOLO keypoint detection algorithm for this problem was then verified.
(2) In this study, the EfficientViT block was integrated into the C2F module in a parallel structure, and the C2FET was proposed as a means of enabling the network to more effectively capture global information.
(3) A Spatial Pyramid Squeeze and Pooling (SPSP) module is proposed for implementation at the interconnection between the backbone network and the neck. The SP

module extracts refined multi-scale features, while the SEWeight module, in conjunction with Softmax, recalibrates channel attention. The SPSP module is an effective means of capturing multi-scale spatial and contextual information.

(4) The concept of Inner-CIoU was introduced to compute the IoU loss using auxiliary bounding boxes. Additionally, a scaling factor was incorporated to adjust the size of these bounding boxes during the loss calculation, which helps improve detection accuracy, particularly for small target points.

(5) A software interface for intuitive interactive recognition has been developed on the Jetson Xavier NX, focusing on various components of YOLO-PP. This development integrates the previously created hardware electronic platform for the tomato harvesting robot in single-fruit picking mode with practical planting scenarios.

2. Related Works

The advancement of intelligent agricultural machinery and the maturity of image classification and segmentation models have increased the application of machine vision technology in agriculture, particularly for inspection and harvesting tasks [10]. The ability to accurately recognise and localise harvesting points is crucial for autonomous robots operating in orchards, where environments are complex and unstructured. Researchers worldwide have extensively studied methods for locating fruit-picking points to enable intelligent harvesting [11], primarily relying on traditional image processing techniques or machine learning and deep learning algorithms [12].

2.1. Traditional Approach

Traditional machine learning methods for fruit recognition often rely on geometric shapes, color, spectral information, texture, and edge detection [13]. Xiang et al. [14] developed an algorithm using binocular stereo vision and an eight-neighborhood pattern for denoising, identifying different tomato types through edge recognition with an iterative Otsu method and edge curvature analysis. Luo et al. [15–17] used stereo vision and image processing techniques to identify and localize grape clusters, select picking points, and measure grape size and perimeter. Rodrigo Pérez-Zavala et al. [18] combined surface pattern details and edge gradient using Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP) to facilitate the segmentation of grapes in relation to the background. [19] proposed a method using artificial neural networks and genetic algorithms to differentiate green grapes from the background. Bai et al. [20] concentrated on the identification of tomatoes based on their texture, form, and colour characteristics, while Jin et al. [21] proposed an image processing algorithm for grape stem recognition and close-range picking point localization. Although these traditional methods can recognize fruit, their accuracy varies significantly with environmental conditions, making them less suitable for detecting picking points in the complex, unstructured environments of natural orchards. As a result, they have gradually been replaced by more advanced techniques.

2.2. Deep Learning Approach

Traditional image processing methods for picking point recognition often struggle to handle complex object variations and to understand visual relationships between the environment and objects, leading to their gradual replacement by deep learning approaches [22]. Tian et al. [23] enhanced detection performance by using CycleGAN for data augmentation and optimizing feature layers with DenseNet. Yan et al. [24] developed a network for the segmentation of tea buds, designated MC-DM, which employed the use of RGB colour space conversion and the Shi-Tomasi algorithm for the detection of partial corner points of tea buds. The optimal picking point was then identified based on the lowest vertical

coordinate. Sa et al. [25] proposed a multimodal Faster R-CNN model that integrated aspects of transfer learning with the utilisation of multimodal data. Chen et al. presented a methodology that integrated YOLOv3, semantic segmentation, and skeleton extraction for tea bud picking point localization. Wu et al. [26] developed a new model, called the YOLO-Banana model, which built on the YOLOv5 Bottleneck module. They used edge detection to identify the cutting point on each banana, which allowed them to create an accurate segmentation of the fruit.

Further advancements in this field include the method proposed by Zhang et al. [27] for the recognition and localisation of tomato cluster harvesting points, which fuses data from RGB-D sensors with object detection algorithms. Li et al. [28] utilized a refined version of the YOLOv5s algorithm, alongside an advanced DeepLabv3+ model, and incorporated depth imaging to craft a method for pinpointing the three-dimensional coordinates of longan fruit on branches. Qi et al. [29] utilized YOLOv5 for litchi trunk detection, applying the PSPNet model for semantic segmentation to pinpoint the main stem's picking point. Zhang [30] introduced the YOLOv5-GAP algorithm for detecting grape clusters, leveraging image processing and geometric methods to segment the clusters and determine picking points using centroid coordinates. These approaches typically involve two distinct subtasks: object detection of fruit images and fine segmentation of the region of interest (ROI) around the fruit stem. Manual rules specific to the fruit type are then applied to determine the picking point, leading to poor robustness, limited generalization, and difficulty in adapting to different fruits and environments, which poses challenges in reusing these methods for cherry tomato picking point detection.

Keypoint detection algorithms, originally developed for human pose estimation [31], have shown significant potential across various fields. As computer vision and deep learning technologies rapidly advance, keypoint detection has expanded beyond human pose estimation to applications in livestock farming [32], healthcare [33], industrial automation [34], and more. In agriculture, for instance, keypoint detection can enable precise crop positioning and automated management. WU et al. [35] developed a stem position localisation method for grapevines using a top-down approach, combining the Ghost module with HRNet to create a new lightweight HRNet. They used object detection to identify grape clusters and applied keypoint detection for stem localization based on human pose estimation techniques. Chen et al. [36] proposed the YOLOv8-GP model, which integrates object and keypoint detection based on YOLOv8n-Pose. This approach achieved an AP of 89.7% in grape cluster detection while simultaneous detection of grape clusters and keypoints was also demonstrated. Inspired by these works, we developed the YOLO-PP model specifically for identifying the picking points of cherry tomatoes. We introduced innovative module designs that simplify the entire detection process compared to some earlier methods, thereby avoiding the inherent complexity and tediousness of traditional multi-step approaches.

3. Materials and Evaluation Metrics

This section provides a summary of the research methodology used in YOLO-PP, followed by a comprehensive account of the data collection process, the annotation of cherry tomatoes, and the evaluation metrics utilized. As illustrated in Figure 1, the proposed YOLO-PP detection model comprises five principal modules. During the data preparation phase, images of cherry tomatoes were captured and annotated, including both the tomatoes themselves and their picking points. The images were then separated into two discrete sets, designated as the training set and the testing set, respectively. In the model construction phase, the original YOLOv8-pose model was established, the C2FET module was integrated into the backbone and neck networks of YOLOv8, while the SPSP module replaced the

original SPPF. Additionally, the Inner-CIoU was incorporated into the model. In the training, prediction, and results phase, a variety of metrics were employed for the evaluation of the model. Additionally, a series of ablation experiments was designed and conducted for the purpose of validating the effectiveness of the implemented improvements.

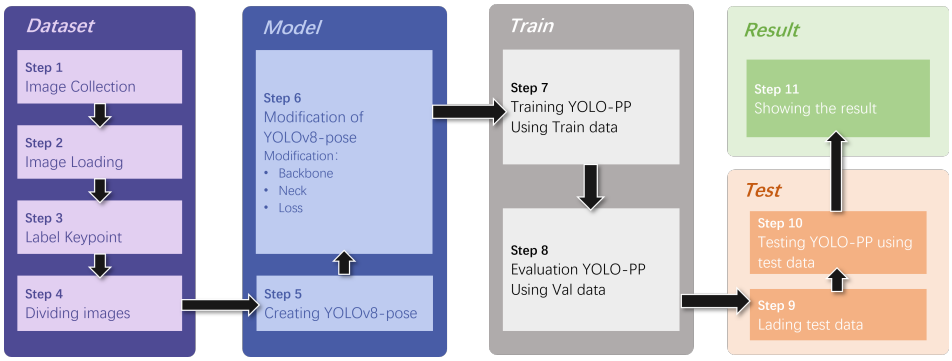


Figure 1. Research flow of our approach.

3.1. Image and Data Acquisition

Our objective is to validate and enhance the performance of the key point detection model specifically for fruit picking point detection tasks. Data collection is carried out with a close-up camera perspective, which is mounted on the end effector to ensure precise and accurate data acquisition. The data were collected from the A+ greenhouse in Kunshan City, Jiangsu Province, China, and focused on the Axia-Axiany and Axia-Xolany cherry tomato varieties grown in a high-bed layout with 80 cm row spacing, which corresponds to the concentrated height of mature tomatoes in the existing manual harvesting mode. The Axiany variety produces bright red, symmetrically clustered fruits with approximately 15 fruits per cluster, while the Xolany variety produces bright brown fruits with 12 to 14 fruits per cluster. Data were collected in March 2024 using handheld close-up photography, as shown in Figure 2b. To increase the efficiency of data collection, we used two different devices: the iPhone 14 camera, which captured images at a resolution of 3024 × 4032 pixels, and the Realsense D435i depth camera, connected to a portable computer, which captured images at a resolution of 1280 × 720 pixels. This dual-device approach ensured both high-quality visual detail and accurate depth information, providing a robust basis for subsequent analysis and model training.



(a) (b)

Figure 2. Cherry tomato cultivation and data collection scene. (**a**) Tomato cultivation environment in facility agriculture, (**b**) Field data collection of tomatoes..

During the data collection period, images were gathered under a variety of lighting conditions, including direct and backlighting. Images of cherry tomatoes were captured from a variety of angles, including front, side, and top-down views, as illustrated in Figure 3.

The multi-angle, multi-environment image set significantly enhances the model's noise tolerance and stability. Training the model on such a diverse dataset allows it to more accurately recognise tomatoes under different perspectives and lighting conditions, thereby closely mimicking the scenarios a robot might encounter during actual picking.

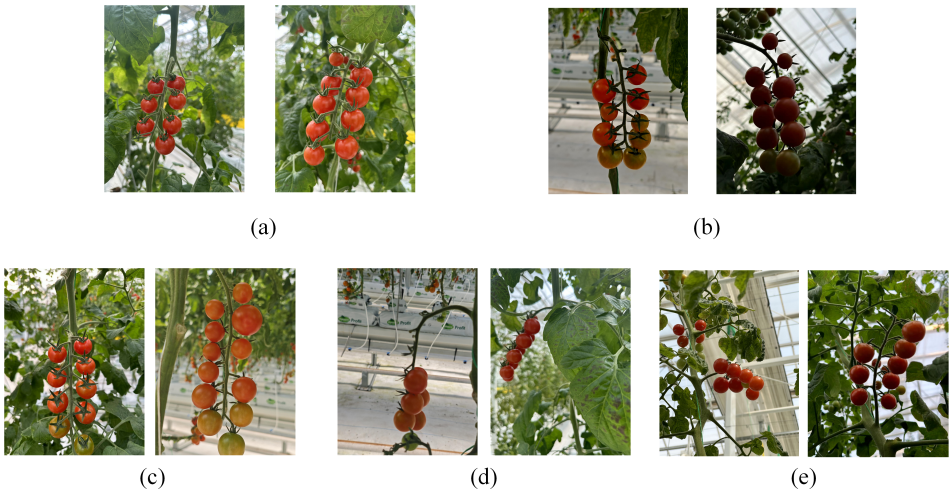


Figure 3. Representative sample datasets in different states: (**a**) Direct light, (**b**) Backlight, (**c**) Front view, (**d**) Side view, (**e**) and Top-down view.

3.2. Annotation of Datasets

We employed the Labelme software (v5.5.0), referenced as [37], to facilitate the annotation process. The annotation format was in accordance with the Pascal VOC standard, as illustrated in Figure 4. The keypoint annotation for cherry tomato target identification and peduncle picking points was carried out in accordance with the following principles: (1) Picking points must be accurately annotated in front, side, and top views; (2) the picking point closest to the uppermost fruit should be selected, ensuring each fruit has only one picking point; (3) the picking point annotation must align with general grasping logic; (4) fruits that are more than 50% occluded or too far from the camera were not annotated. These principles ensured the accuracy, validity, and fairness of the annotations, providing a robust foundation for subsequent data analysis and algorithm training. A total of 919 images were effectively annotated. As there was no separate validation set, these images were split into a training set (736 images) and a test set (183 images) in an 8:2 ratio.

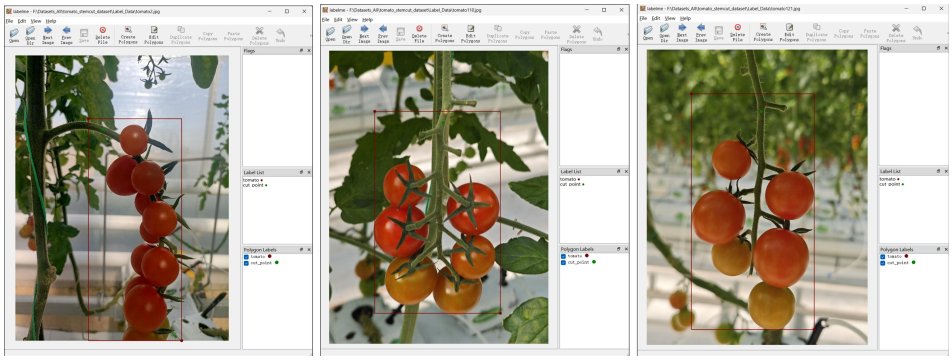


Figure 4. Data annotation process.

3.3. Evaluation Metrics

This study conducted experimental comparisons using evaluation metrics focused solely on picking key points, using the mean Average Precision (mAP) based on the official

MS COCO object keypoint similarity metric Loks [38] as the evaluation criterion. Loks is represented by the equation:

$$L_{oks} = \frac{\sum_i \left[exp^{\left(-\frac{d_i^2}{2s^2k_i^2} \right)} \delta(v_i > 0) \right]}{\sum_i \delta(v_i > 0)} \quad (1)$$

where i is the index of the keypoint label; d_i^2 is used to denote the squared Euclidean distance between the position of the detected keypoint and the position of the ground truth keypoint; s^2 is the area of the image that is occupied by the particular object that has been identified; k_i is a decay constant controlling the keypoint category i; δ is an indicator function, indicating that only visible keypoints in the ground truth annotation are considered for L_{oks}; v_i is the visibility value of keypoint i (where $v_i > 0$ indicates the keypoint is visible). In order to evaluate the accuracy of keypoint recognition, this study employs the use of three performance metrics: precision (P), recall (R), and average precision (AP). These metrics are defined in Equations (2)–(4) below.

$$Precision_{kpt} = \frac{TP_{kpt}}{TP_{kpt} + FP_{kpt}} \quad (2)$$

$$Recall_{kpt} = \frac{TP_{kpt}}{TP_{kpt} + FN_{kpt}} \quad (3)$$

$$AP = \int_0^1 Precision_{kpt} dRecall_{kpt} \quad (4)$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \quad (5)$$

When a tomato stem is correctly identified, a true positive keypoint (TP_{kpt}) is produced, where the predicted keypoint's Loks score exceeds the threshold for stem cutting keypoints. A false positive keypoint (FPkpt) occurs when a non-stem area is mistakenly identified as a stem, with the predicted keypoint surpassing the Loks threshold. A false negative keypoint (FNkpt) refers to the missed identification of actual stem cutting keypoints, where the predicted keypoint does not meet the required Loks threshold.

The Average Precision (AP) is calculated based on the area under the Precision–Recall (P-R) curve, as defined by Equations (2) to (4). The mAP50 is obtained by averaging the Average Precision (AP) values across all categories at a threshold of 0.5. On the other hand, mAP50-95 is derived by computing the AP values at a series of thresholds ranging from 0.5 to 0.95, with a step size of 0.05, and then taking the mean of these AP values according to Equation (5). Due to the primary limitations of mAP50-95, which may overly rely on precision while neglecting recall in certain scenarios, leading to potentially favorable performance in cases of low recall, this study also selects Precision, Recall, mAP50, and Params as evaluation metrics for both baseline model comparison experiments and ablation experiments.

4. Methodologies

4.1. YOLO-PP Architecture Overview

The YOLO (You Only Look Once) algorithm is well-known for its efficient single-stage detection and has been widely studied [39–42]. The latest version, YOLOv8, developed and maintained by Ultralytics [43], includes features for object detection, instance segmentation, and keypoint detection. YOLOv8 uses CSPDarknet53 as its backbone, which incorporates multiple C2f modules to streamline computation by splitting and merging feature maps.

The SPPF module at the end of the backbone extracts features and expands the network's receptive field, helping detect larger objects. For multi-scale detection, YOLOv8 uses the FPN (Feature Pyramid Network) and PAN (Path Aggregation Network) for feature fusion, allowing detection of various object sizes and improving the transmission of semantic and localization information. The YOLOv8 head, based on YOLOX, separates object classification from localization and moves from anchor-based to anchor-free processing, reducing computational demands. The classification part focuses on texture, while the localization part focuses on object edges.

Considering its excellent overall performance and speed advantages in keypoint detection tasks, the lightweight YOLOv8 network was selected as the backbone for this study. Although YOLOv8 is already highly refined in various aspects, it still faces challenges in identifying small targets in complex scenes, particularly for keypoint tasks. The reasons for the inaccuracies in small-range keypoint detection are analyzed as follows: Compared to normal-sized targets, small keypoint targets are more susceptible to interference from other objects, and the features extracted from deeper layers lack substantial information about small-sized targets. This results in small targets being overlooked during the learning process, leading to poor detection performance and significant offset issues. To address these problems, this study proposes the YOLO-Picking Point (YOLO-PP) algorithm, which is an improved version of YOLOv8-Pose. It is designed to rapidly and accurately detect the picking points of cherry tomato stems in controlled environments.

Firstly, we proposed the C2FET parallel architecture module for feature fusion, which integrates C2F with the Efficient ViT module to better combine shallow and deep information. This ensures that the information retained during the network's feature extraction process is more comprehensive. Specifically, the multi-scale linear attention mechanism in C2FET achieves global receptive fields and multi-scale learning, which is subsequently incorporated into the backbone network for more effective feature extraction. Additionally, the Spatial Pyramid Squeeze and Pooling (SPSP) module is added to enhance feature representation, and the Inner-CIoU loss function is employed to optimize regression and improve generalization. Through these optimizations, we successfully developed the YOLO-PP algorithm for cherry tomato stem picking point detection, with its network architecture illustrated in Figure 5.

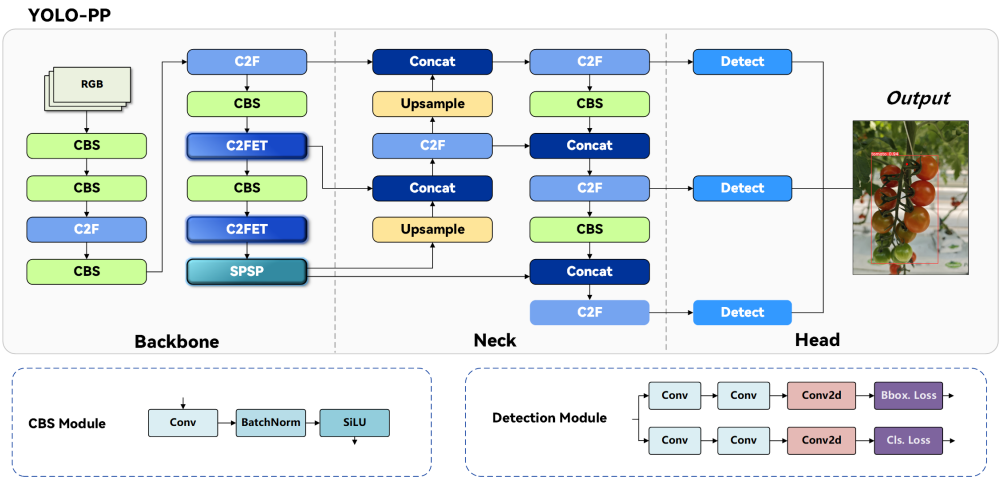


Figure 5. The overall network architecture of YOLO-PP.

4.2. C2FET Module

The robot's visual perception component for harvesting, when facing the dense background of a cherry tomato cultivation scene, needs not only to collect relevant keypoint information from local regions but also to effectively capture the spatial relationships be-

tween keypoints in the image, thus enabling more precise keypoint detection. The YOLOv5 network utilizes multiple C3 structures to increase network depth and receptive fields, as shown in Figure 6. Each C3 structure consists of three standard convolutional modules and one residual module. However, under low-light conditions, the C3 module fails to extract effective low-level features and reduce the impact of insufficient lighting and noise features, thereby affecting target detection performance. In YOLOv8, the C2f module is introduced. During the feature integration process, the feature dimensionality significantly increases. The input features are mapped to an expanded feature space through the initial convolutional layer, followed by feature splitting and reorganization. This is combined with the sequential use of multiple C2f bottleneck blocks to deepen the processing of multi-scale features. The main computational load of the C2f module is concentrated in its bottleneck section, which consists of two convolutional layers, each followed by batch normalization and SiLU activation. These bottlenecks are arranged sequentially, with the output of each upper bottleneck connected to the input of the next layer to facilitate feature fusion. However, C2F is less sensitive to the detection of small targets in dense backgrounds. We aim to design an optimized component that incorporates a lightweight Vision Transformer. Specifically, we retain some C2F modules in the backbone and neck networks, and for the end of the backbone network—the part connected to the neck—we integrate the EfficientViT block [44] into the C2F module, forming the C2FET module. This module consists of two parallel branches, one based on Convolutional Neural Networks (CNN) and the other on Transformer.

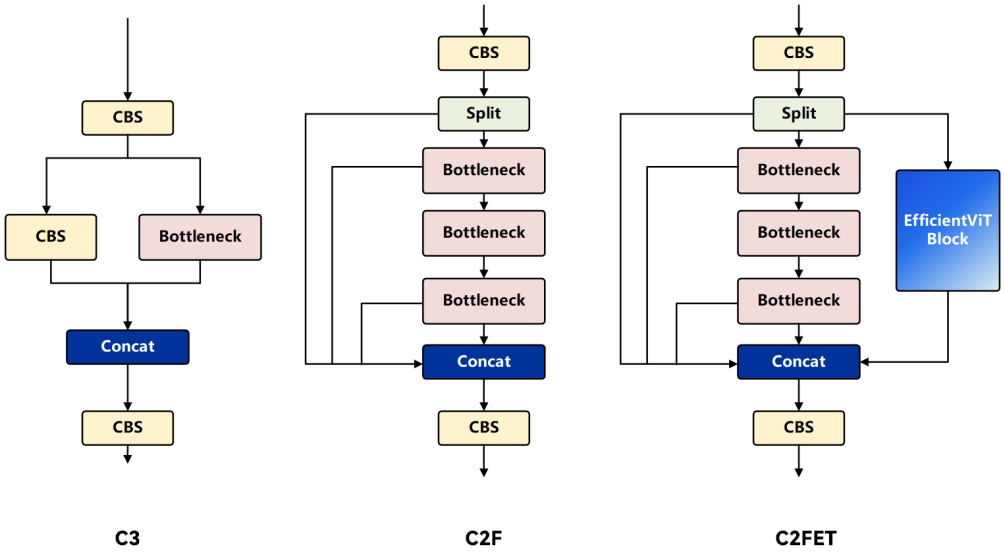


Figure 6. The evolution of the architecture of C3, C2F, and C2FET Modules.

The CNN branch follows the original architecture of C2F, as shown in Figure 6. The input data first passes through the initial CBS module, which then splits the output into two parts. The CBS module is a fundamental component in YOLO, consisting of a convolutional layer with the SiLU activation function. One part is passed directly to the output to preserve the spatial consistency of the information. The other part consists of a convolutional layer followed by several bottleneck layers. Each bottleneck layer employs a residual structure to facilitate effective deep learning network training. In these layers, the channel count of the input feature map is first halved by a 1×1 convolution operation, and then the channel count is restored to its original dimension using a 3×3 convolution operation.

In the Transformer branch, to balance the computational cost of the self-attention mechanism capable of capturing global long-range dependencies, we introduce a new EfficientViT block. Figure 7b illustrates its specific structure, which mainly includes the

Overlap Patch Embedding module, Cascade Grouped Agent Attention module (CGAA), Token Interaction, and FeedForward Network. This three-layer architecture leverages the cascaded group attention module and a parameter redistribution strategy, combining self-attention layers with memory-efficient FFN layers to reduce memory requirements, thereby improving channel communication. The Multi-Head Self-Attention (MHSA) mechanism enhances model performance by embedding the input sequence into multiple attention heads and computing attention separately and simultaneously. However, due to the high similarity among many attention heads—where many heads learn similar projections of the same complete features—significant computational redundancy arises. Given the limited inference resources on robotic edge devices, the EfficientViT block addresses this issue by adopting the Cascade Grouped Attention (CGA) module, which explicitly decomposes the attention computation for each head. This attention can be expressed as:

$$\begin{cases} \widetilde{X}_{ij} = \mathrm{Attn}(X_{ij}W_{ij}^{\mathrm{Q}}, X_{ij}W_{ij}^{\mathrm{K}}, X_{ij}W_{ij}^{\mathrm{V}}), \\ \widetilde{X}_{i+1} = \mathrm{Concat}\left[\widetilde{X}_{ij}\right]_{j=1:h} W_{i}^{\mathrm{P}} \end{cases} \tag{6}$$

The j-th head is responsible for computing the self-attention on X_{ij}, where X_{ij} represents the j-th partition of the input feature X_i, i.e., $X_i = [X_{i1}, X_{i2}, \ldots, X_{ih}]$, where $1 \leq j \leq h$. h is the total number of heads, and W_{ij}^{Q}, W_{ij}^{K}, and W_{ij}^{V} are projection layers that map the input features to different subspaces. W_i^{P} is a linear layer that projects the concatenated output features back to the same dimension as the input. The attention map for each head is calculated in a cascaded manner, as illustrated in Figure 7c. This involves the progressive addition of the output of each head to the next, resulting in a gradual refinement of the feature representation.

$$X_{ij}^{'} = X_{ij} + \widetilde{X}_{i(j-1)}, \quad 1 < j \leq h \tag{7}$$

Here, X_{ij}' and the output of the $(j-1)$-th head, $X_{e(i,j-1)}$, as computed by Equation (6). The variable X_{ij} is replaced with a new input for use in the self-attention calculation for head j. This design aims to fully leverage the critical role of the V projection in information transmission. At the FFN (FeedForward Network) level, we optimized the issue of parameter redundancy by reducing the expansion ratio from 4 to 2. This adjustment not only lowers the model's complexity but also enhances its efficiency. Key modules are equipped with a sufficient number of channels to capture and learn rich representations in high-dimensional spaces, while redundant parameters in non-critical modules are removed. This design not only prevents the loss of feature information during model training but also improves the model's inference speed. The multi-layer module architecture proposed in this study is structured as follows: Token Interaction + Linear FFN + Cascade Grouped Attention + Token Interaction + Linear FFN. The formula is expressed as follows:

$$X_{i+1} = \prod_{i}^{N} \Phi_i^F(\Phi_i^A(\prod_{i}^{N} \Phi_i^F(X_i))) \tag{8}$$

In dense backgrounds, especially when the target and background share similar colors and are difficult to distinguish, relying solely on local features can easily lead to false positives or missed detections. Taking the cherry tomato harvesting scenario as an example, the stems are similar in color to the green leaf background, and in some cases, partial occlusion between stems and fruits occurs. Relying only on the local features extracted by the CNN branch makes it difficult to accurately distinguish fruit edges or locate picking points. By introducing the Transformer branch, the C2FET module can capture global relationships between targets, such as analyzing the spatial distribution patterns of fruits or the connection relationships between stems and fruits, thereby significantly improving

the accuracy of target recognition and localization. Specifically, the CNN branch focuses on extracting local detailed features, such as the texture and edge information of fruit surfaces, while the Transformer branch captures global semantic information through the self-attention mechanism, such as the long-range dependencies between fruits and stems. Each branch learns different features and complements them through a feature fusion mechanism. During the feature fusion stage, the outputs of the three branches are integrated along the channel dimension to ensure effective fusion of multi-scale features. For instance, the local features extracted by the CNN branch can help precisely locate fruit edges, while the global semantic information provided by the Transformer branch can effectively distinguish fruits from the background, avoiding the misidentification of dense green leaves as fruits.

This design allows the C2FET module to fully leverage the respective strengths of convolutional neural networks and Transformers: the CNN branch excels at capturing local detailed features, while the Transformer branch enhances the model's ability to learn long-range dependencies through the self-attention mechanism. For example, in complex scenes with densely distributed fruits, the Transformer branch can analyze the spatial distribution patterns of fruits, avoiding missed detections caused by occlusion. Experimental results demonstrate that this combined approach achieves more accurate predictions in the task of cherry tomato picking point detection while significantly improving the model's multi-scale learning capability and overall performance.

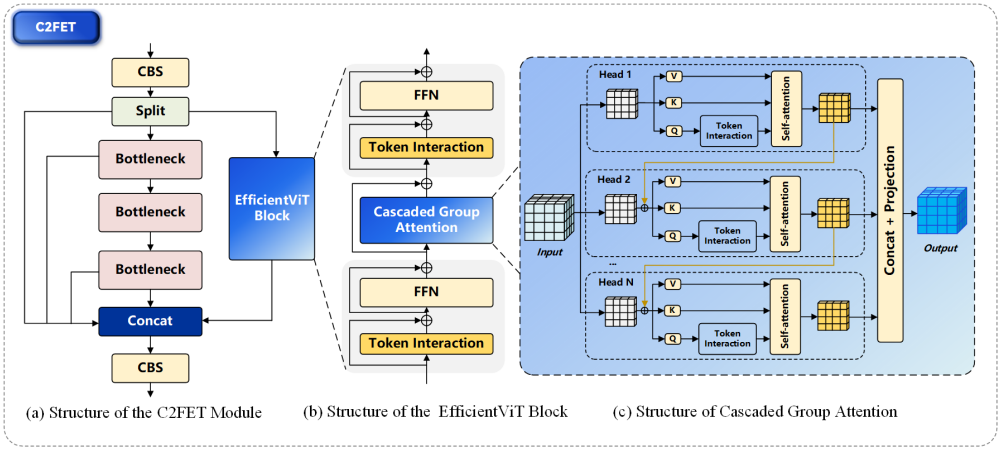


Figure 7. (**a**) The left path integrates a fundamental convolutional component and a series of bottleneck structures. The primary function of these structures is to refine residual features and integrate the outputs of the two independent branches of the C2FET module at the endpoint. (**b**) The constructed Transformer branch adopts a three-layer architecture and incorporates a progressive group attention mechanism. (**c**) The Cascaded Group Attention (CGA) module meticulously deconstructs the computation process of each attention head, customizing feature enhancement for each head to improve the diversity of attention maps.

4.3. Spatial Pyramid Squeeze and Pooling Module

The SPPF module is situated between the backbone network and the neck network, employing pooling operations at various scales to extract multi-scale feature information. This representation enables the model to more effectively capture the shape, size, and position of the target, thereby enhancing the overall accuracy of the output. However, through extensive testing for the specific task of cherry tomato picking point detection, we have identified certain limitations of the SPPF module. In the complex greenhouse environment, where cherry tomatoes may be occluded by their stems and the color of the fruits may closely resemble the background, the features extracted by the SPPF module are not sufficiently precise in distinguishing these subtle differences. This leads to misjudgments

and omissions in identifying cherry tomatoes and their picking points, ultimately affecting the model's performance. Therefore, we aim to design a new module based on the ELAN structure, which enhances the network's ability to capture spatial information through the use of stacked convolutional layers and progressive feature fusion, as illustrated in Figure 8.

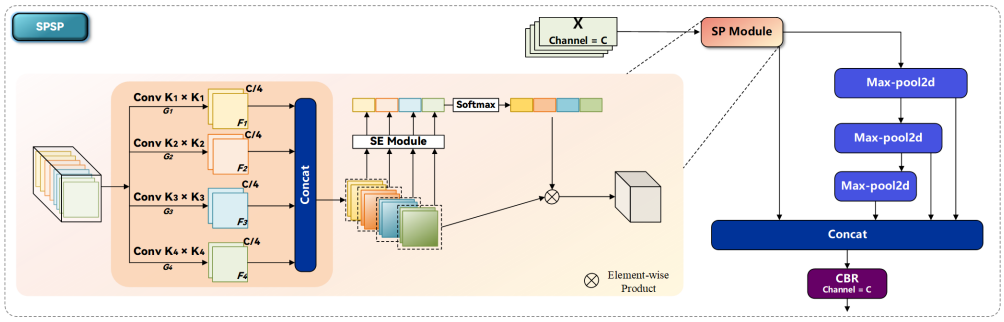


Figure 8. The structure of the SPSP module.

The SPSP module, based on the SPPF, integrates the SP module into the frontend. This module effectively extracts spatial information from feature maps at different scales and establishes long-term channel dependencies. The input of the SP module is a feature map $X \in \mathbb{R}^{C\times H\times W}$, where C is the number of channels, H is the height, and W is the width. The output is a feature map $Y \in \mathbb{R}^{C\times H\times W}$, which contains richer multi-scale spatial information and stronger channel attention.

The specific implementation process is shown in Figure 8. The feature map X is divided into S parts along the channel dimension, and convolution kernels K_i of different scales are applied to extract spatial information, resulting in multi-scale feature maps $F_i \in \mathbb{R}^{C\times H\times W}$. This design enables the extraction of richer positional information from the input tensor and simultaneously processes it across multiple scales. By reducing the channel dimension of the input tensor, spatial information from different scales can be effectively captured in each channel feature map. Each branch independently learns spatial information at different scales and promotes local cross-channel interactions. Additionally, it reduces the number of parameters and computational complexity by adaptively selecting the appropriate kernel size. After the above steps, all feature maps F_i from different scales are concatenated, resulting in a multi-scale fused feature map $F \in \mathbb{R}^{C\times H\times W}$, which contains spatial information from different scales and enhances feature representation capabilities. The computation formula for the concatenation operation is as follows:

$$\begin{cases} F = \mathrm{Cat}([F_1, F_2, \ldots, F_{S-1}]), & F \in \mathbb{R}^{C\times H\times W} \\ F_i = \mathrm{Conv}(k_i \times k_i, G_i)(X), & i = 1, 2, 3 \ldots, S-1 \end{cases} \tag{9}$$

The overall multi-scale preprocessed feature map F is obtained through concatenation of feature maps at different scales, represented by $F_i \in \mathbb{R}^{C'\times H\times W}$. The size of the i-th convolution kernel is defined as $k_i = 2 \times (i+1) + 1$, while the size of the i-th group is denoted as G_i. This method, known as group convolution, is introduced to handle input tensors with varying convolution kernel scales. The rule is as follows: if $K > 3$, the group size G is $2K - 1$; otherwise, if $K = 3$, the group size G is set to 1.

For each scale feature map F_i, the SE (Squeeze-and-Excitation) module is used to compute the channel attention weights $Z_i \in \mathbb{R}^{C\times H\times W}$. Then, all the weight vectors Z_i from different scales are concatenated to obtain a multi-scale channel attention vector $Z \in \mathbb{R}^{C\times H\times W}$. Z contains channel information from different scales, which enhances the feature discrimination capability. To further promote the interaction of attention information while retaining the original channel attention vector and integrating cross-dimensional

vectors, we use the Softmax function to adjust the multi-scale channel weights, generating a comprehensive weight vector that contains both spatial location and channel attention information. This step achieves effective interaction between local and global channel attention.

$$\begin{cases} Z = Z_0 \oplus Z_1 \oplus \cdots \oplus Z_{s-1} \\ \text{softmax}(z_i) = \dfrac{\exp(z_i)}{\sum_{j=1}^{n} \exp(z_j)} \end{cases} \tag{10}$$

In the final step, an element-wise multiplication is performed between W and F to obtain a multi-scale feature information attention-weighted feature map $Y \in \mathbb{R}^{C \times H \times W}$. The element-wise multiplication operation enables matching and fusion between the feature map and the attention weights, thereby improving feature quality and classification performance. This step enables effective interaction between local and global channel attention. Next, the calibrated channel attention vectors are merged and concatenated to form the final channel attention vector. Finally, the recalibrated multi-scale channel attention weights are multiplied with the corresponding scale feature maps, completing the weighted fusion of attention.

4.4. Loss Function

The loss function for object detection typically focuses on introducing new loss terms to accelerate convergence, such as a series of IoU-based loss functions, including GIoU, DIoU, CIoU, and EIoU, which have largely addressed related issues. YOLOv8 uses the CIoU loss function by default, balancing the center point distance and aspect ratio. However, in practical applications, especially for irregularly shaped objects like cherry tomatoes, it struggles in distinguishing green stems with colors similar to the background in cluttered vertical greenhouse environments. It cannot adaptively adjust according to different detectors and detection tasks, lacking strong generalization capability. We hope that the YOLO-PP model in the future can adapt to detecting picking points for various fruits and vegetables while overcoming the limited generalization ability of existing IoU-based loss functions [45,46]. To achieve this, we adopted the Inner-IoU loss function proposed by Zhang et al. [42].

This function introduces an auxiliary bounding box with a scaling factor to adjust its size, improving computational efficiency and convergence. In Figure 9, the Ground Truth (GT) frames are represented as b^{gt}, while the Anchor frames are denoted as b. The centers of the true frame and its inner frame are (x_c^{gt}, y_c^{gt}), and the centroids of the anchor frame and its inner frame are (x_c, y_c). The width and height of the real and anchor frames are wgt, hgt, and w, h, respectively. The scaling ratio, typically between [0.5, 1.5], adjusts the auxiliary frame size: if the ratio is less than 1, the smaller auxiliary frame quickly improves accuracy for high IoU samples; if greater than 1, the larger frame helps with low IoU samples. The Inner-IoU equation is defined as follows:

$$b_l^{gt} = x_c^{gt} - \frac{w^{gt} * ratio}{2}, b_r^{gt} = x_r^{gt} + \frac{w^{gt} * ratio}{2} \tag{11}$$

$$b_t^{gt} = y_c^{gt} - \frac{h^{gt} * ratio}{2}, b_b^{gt} = y_c^{gt} + \frac{h^{gt} * ratio}{2} \tag{12}$$

$$b_l = x_c - \frac{w * ratio}{2}, b_l = x_c + \frac{w * ratio}{2} \tag{13}$$

$$b_t = y_c - \frac{h * ratio}{2}, b_b = y_c + \frac{h * ratio}{2} \tag{14}$$

$$inter = \left(\min\left(b_r^{gt}, b_r\right) - \max\left(b_l^{gt}, b_l\right)\right) * \left(\min\left(b_b^{gt}, b_b\right) - \max\left(b_t^{gt}, b_t\right)\right) \quad (15)$$

$$IoU^{inner} = \frac{inter}{union} \quad (16)$$

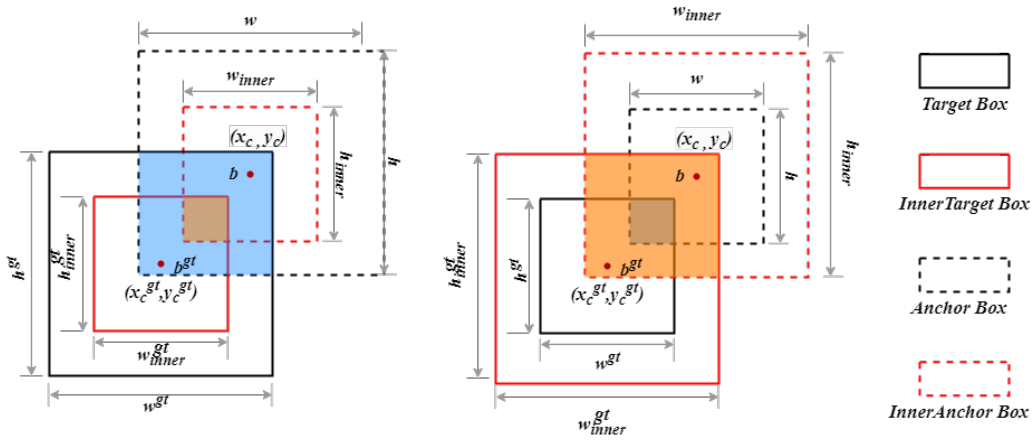


Figure 9. The representation of Inner-IoU and visual explanation.

By using Inner-CIoU with a scaling factor less than 1 to optimize CIoU, we can accelerate sample regression, improve the network's generalization ability, and speed up the model's convergence. YOLOv8 employs CIoU (Complete IoU Loss) for box-IoU, which builds on DIoU by adding a shape loss. The CIoU loss is calculated as follows:

$$L_{CIoU} = 1 - IoU + \frac{\rho^2\left(b, b^{gt}\right)}{c^2} + av \quad (17)$$

Here, b and b^{gt} represent the centers of the predicted and ground truth bounding boxes, respectively; ρ^2 is the Euclidean distance between these centers; and c is the diagonal length of the smallest rectangle that can cover both bounding boxes. The Inner-CIoU introduces a scale factor `ratio`, which controls the size of the auxiliary bounding box during loss calculation. The value of the `ratio` ranges from [0.5, 1]. For samples with higher IoU, using a smaller auxiliary bounding box to calculate the loss can accelerate convergence. On the other hand, for samples with lower IoU, using a larger auxiliary bounding box helps to improve the regression performance. By focusing on the shape and scale of the auxiliary bounding box, Inner-CIoU can better adapt to targets of various shapes in real-world scenarios. The calculation for Inner-CIoU used in this study is shown in Formula (13):

$$L_{inner-CIoU} = L_{CIoU} + IoU - IoU^{inner} \quad (18)$$

5. Experiments and Evaluation

5.1. Experimental Setup

In order to confirm that the use of the improved YOLO-PP model leads to better overall detection performance in the cherry tomato picking point accuracy recognition task, we conducted experiments using the cherry tomato dataset collected in the A+ greenhouse, and we carried out all the research processes on a Windows 11 device using a CPU of Intel(R) Core(TM) i5- 12600KF with NVIDIA GeForce RTX 4090 with 24 G RAM, and conducted all the comparison experiments under Python 3.8.18 Pytorch1.13.1 with CUDA 11.6 to ensure the uniformity of the experimental environment. In the final embedded device and interactive interface experiment, we used NVIDIA Jetson Xavier NX as the edge computing device for the cherry tomato picking task, as shown in Table 1. During the training process,

in order to ensure the consistency of the input images, we first resize them to the required size while maintaining a fixed aspect ratio. The Adam optimizer is used for training, with an initial learning rate of 0.01 and a batch size of 12. Other specific environmental parameters used are presented in Table 2.

Table 1. Experimental environment.

Hardware/Software	Configuration
CPU	Intel(R) Core(TM)i5-12600KF
Memory (GB)	16 G
GPU	NVIDIA GeForce RTX 4090
Graphics Memory (GB)	24 G
Training Envionment	CUDA 11.6
Operating System	Windows 11 (64-bit)
Development Environment	Python 3.8.18 & Pytorch 1.13.1
Embedded Device	NVIDIA Jetson Xavier NX

Table 2. Experimental hyperparameters settings of YOLO-PP.

Parameter	Value
Initial learning rate	0.01
Optimizer	Adam
Momentum	0.937
Weight Decay	0.0005
Batch size	12
Input image size	960 × 960
Training Epochs	150

5.2. Comparison of Network Models

To verify the superiority of the proposed YOLO-PP model, we selected three representative models—DEKR, YOLO-Pose, and YOLOv8-Pose—for comparative experiments. Our research focused on performance metrics related to keypoint detection, systematically grouping and testing the adaptability and performance of these models under different image resolutions while keeping the backbone architecture consistent. The experimental results are summarized in Table 3. In the YOLO-Pose experiment setup, CSPDarknet53 m was chosen as the backbone network, whereas CSPDarknet53 s was selected for YOLOv8-Pose and YOLO-PP. This selection aimed to minimize the influence of backbone variation on performance during comparisons. The results show that under an input image resolution of 640 × 640 pixels, YOLO-PP achieved 98.86% in Recall and 99.18% in mAP50, significantly outperforming DEKR, YOLO-Pose, and YOLOv8-Pose. The limitations of YOLO-Pose can be attributed to its coupled head structure, where a single branch simultaneously learns both the target's location and category information. This approach may encounter challenges when handling complex object detection tasks. In contrast, models like YOLO-PP and YOLOv8-Pose, which adopt a decoupled head structure, use multiple branches to optimize different tasks independently, thereby achieving better overall performance.

We also paid special attention to the performance of the YOLO-PP model under varying input resolutions. Notably, under consistent input resolution conditions, the C2FET module in YOLO-PP demonstrated unique advantages. By integrating CNN and Transformer branches, this module effectively extracts and enhances features, unlocking greater potential when processing high-resolution images. Specifically, when the input resolution increased to 960 × 960 pixels, YOLO-PP achieved 98.86% in Recall, 99.18% in mAP50, and 98.87% in mAP50-95. Compared to YOLOv8-Pose, YOLO-PP achieved notable improvements of 1.14%, 0.23%, and 0.63% in these respective metrics. The line graph in

Figure 10 visually demonstrates that these improvements are particularly pronounced under high-resolution image conditions. Compared to the 640 × 640 input condition, YOLO-PP exhibits a more significant performance gain when processing high-resolution images, further proving its superiority in handling high-resolution image processing.

Table 3. Cherry tomato picking point detection. Comparative experiments of different models.

Model	Backbone	Size	Recall	mAP50	mAP50-95	Params
DEKR	HRNet-W32	640	98.23	82.24	79.85	29.5 M
YOLO-Pose	CSPDarknet53 m	640	96.17	89.72	87.46	21.2 M
YOLOv8-Pose	CSPDarknet53 s	640	97.54	98.37	96.76	11.4 M
YOLO-PP	CSPDarknet53 s	640	97.51	98.49	97.29	12.7 M
YOLO-Pose	CSPDarknet53 m	960	96.23	88.35	87.51	21.2 M
YOLOv8-Pose	CSPDarknet53 s	960	97.72	98.95	98.24	11.4 M
YOLO-PP	CSPDarknet53 s	960	**98.86**	**99.18**	**98.87**	12.7M

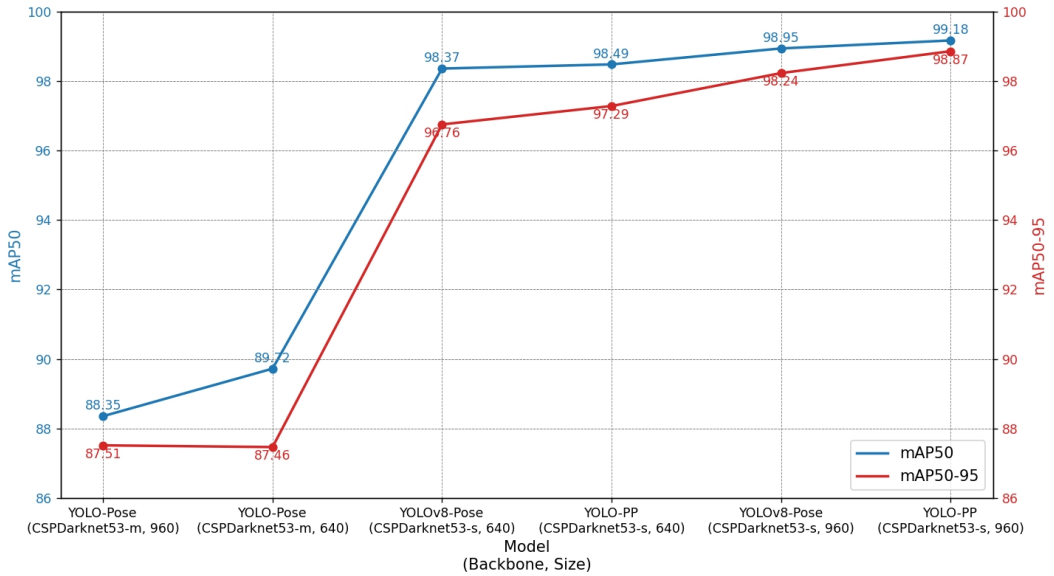


Figure 10. Comparison of mAP@50 and mAP@50-95 results for different models.

Our goal is to deploy an efficient and accurate object detection model on a picking robot to enable fast identification and localization of cherry tomato picking points. Thus, we focused on both parameter efficiency and real-world performance. The YOLO-PP model we designed maintains high detection accuracy while increasing the parameter count by only 11.4% compared to the baseline models. In subsequent experiments, we also tested the detection speed and frame rate of YOLO-PP on the NVIDIA Jetson Xavier NX edge computing platform, with details provided in the following sections.

The detection and comparison results are shown in Figure 11, illustrating the predictions of YOLO-Pose, YOLOv8-Pose, and YOLO-PP for cherry tomato picking points across various scenarios. Red bounding boxes represent the detection results of the cherry tomato bodies, which are not the primary focus of this study. Our focus is on the models' predictions of picking points on the tomato stems, highlighted with red dots. It can be observed that YOLO-Pose experienced one instance of missed keypoint detection and two instances of low-quality keypoint localization. In Scenario 1, the YOLO-Pose model failed to predict the picking point, whereas YOLO-PP predicted a point closer to the central axis of the stem. In Scenario 2, only YOLO-PP successfully predicted the correct picking point, while YOLO-Pose and YOLOv8-Pose deviated toward different areas. In Scenario 4, our algorithm accurately identified a stable picking point on the main stem, while the

other models mistakenly selected a leaf adjacent to the tomato, which would result in a picking task failure for the robot. The results of Scenarios 5 and 2 exhibit similar deviations, where YOLO-PP proves to be the most accurate. In complex environments, YOLO-PP consistently avoided missing picking points, and its detection results were more aligned with confidence scores. Compared with the other two models, our model predicts picking points closer to the stem, making it more consistent with the expectations of practical human-intuitive operations.



Figure 11. Actual detection picking point results of the different network models.

To comprehensively evaluate the performance of the YOLO-PP model, we also tested its detection effectiveness in various special scenarios. First, as shown in Figure 12a, we present a set of detection scenarios under different lighting conditions, with brightness decreasing from left to right. YOLO-PP demonstrated remarkable robustness, with only a single fruit far from the main stem being missed under the darkest lighting condition. Figure 12b displays detection results in more complex scenarios, such as multi-target picking point detection (although our sub-camera primarily focuses on a single target). YOLO-PP consistently avoided missing picking points. The last three images depict some occlusion scenarios, where YOLO-PP accurately provided correct picking coordinates even with leaf obstructions. Particularly in the last image, despite a cluster of cherry tomatoes located behind and above the main target, YOLO-PP not only avoided confusing the rear target with the front one but also effectively prevented false detection of the severely occluded and non-target objects above. This demonstrates that YOLO-PP exhibits excellent adaptability and stability in complex lighting, multi-target, and occlusion scenarios, effectively addressing various challenges in practical picking tasks.



Figure 12. Performance of YOLO-PP in special cases: (**a**) Detection results under different lighting conditions; (**b**) Detection results in multi-target and occlusion scenarios.

5.3. Ablation Experiment

In this section, we evaluate the performance of YOLO-PP through a series of ablation experiments. Table 4 presents the results of these experiments, highlighting the significant improvements our proposed algorithm achieves in key performance metrics, including Precision, Recall, and mAP50, compared to the original YOLOv8-Pose network. Specifically, the introduction of the C2FET module increased Precision by 0.14%, underscoring its significant contribution to precise target localization. The subsequent integration of the Inner CIoU module further boosted Precision by 0.46%, alongside notable improvements in Recall and mAP. Optimizing the backbone network using the SPSP Module resulted in an additional 0.54% increase in Precision. Overall, our algorithm improved Precision by 1%, Recall by 1.14%, and mAP50 by 0.23% compared to the baseline model. These substantial enhancements demonstrate the effectiveness and superiority of our algorithm in object detection tasks.

Table 4. Ablation experiment results.

SPSP Module	C2FET Module	Inner CIoU	Precision (%)	Recall (%)	mAP50 (%)
✗	✗	✗	94.81	97.72	98.95
✗	✗	✓	93.49	98.12	98.85
✗	✓	✗	94.95	97.76	98.96
✓	✗	✗	94.53	97.61	98.78
✗	✓	✓	95.27	97.80	99.15
✓	✓	✓	**95.81**	**98.86**	**99.18**

Figure 13 visually illustrates the performance of YOLO-PP across 150 training steps for various evaluation metrics. The purple-highlighted line clearly indicates the convergence trend of YOLO-PP metrics. In Figure 13a, the Precision results clearly show the superiority of YOLO-PP over other models. Figure 13b,c further reveal that YOLO-PP not only maintains high accuracy but also exhibits lower fluctuation, indicating excellent stability. Figure 14 displays the loss function curves from the ablation experiments, demonstrating the significant role of the InnerCIoU loss function in accelerating model convergence. Compared to traditional loss functions, InnerCIoU offers more effective weight adjustments and faster stabilization. Its superior ability to capture subtle differences in bounding box regression provides more precise feedback during the optimization process.

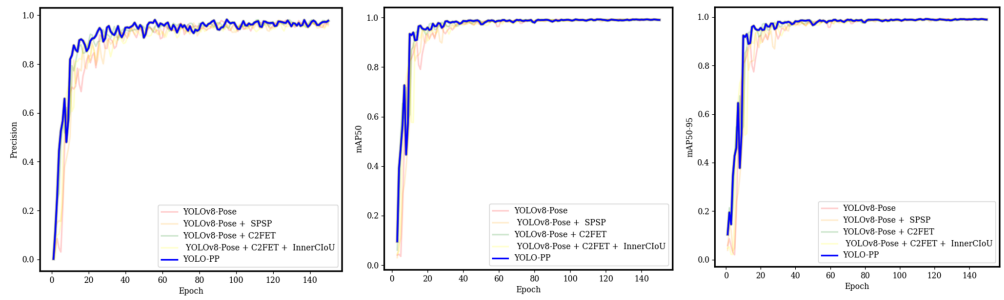


Figure 13. Comparison of ablation results from precision mAP@0.5 and mAP@0.5–0.95: Precision curve; mAP50 curve; mAP50-95 curve.

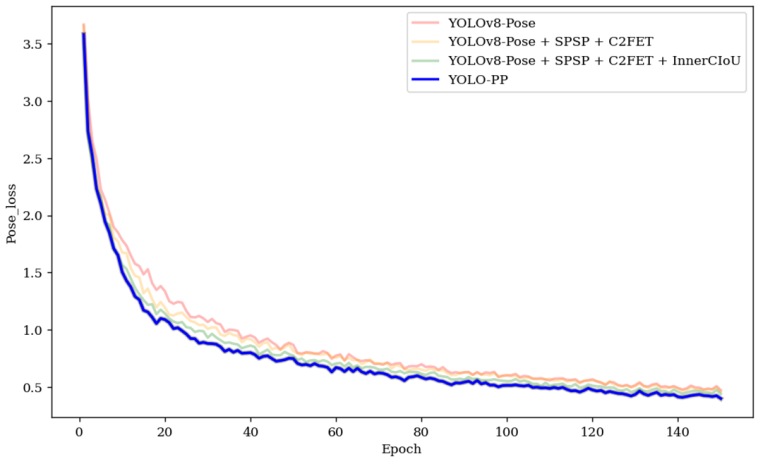


Figure 14. Variation curves of loss function for ablation experiments.

Figure 15 illustrates the curves of the loss function values during the training process, where the horizontal axis represents the number of iterations and the vertical axis represents the loss values. This graph visualises the changes in loss values over the course of training, providing a clear representation of the model's convergence process and performance improvements. After completing 200 training batches, the bounding box loss and pose estimation loss curves were obtained, as shown in Figure 15a for YOLOv8-Pose and Figure 15b for YOLO-PP. It can be observed that on the training dataset, the loss values for YOLOv8-Pose and YOLO-PP show similar fluctuations. However, on the validation dataset, the bounding box loss values for YOLOv8-Pose show irregular and significant fluctuations, whereas the improved bounding box loss values decrease significantly and remain stable, with the final loss values outperforming those of YOLOv8-Pose.

As shown in Table 5, we replaced the original CIoU loss function with the proposed Inner-CIoU loss function while retaining the SPSP and C2FET modules. We conducted comparative experiments with several mainstream IoU metric methods, such as EIoU, SIoU, DIoU, and Shape-IoU. The experimental results demonstrate that Inner-CIoU outperforms these methods in perceptual capability, effectively utilizing the width and height information of bounding boxes, significantly improving the network's performance in detecting small keypoints under monochromatic backgrounds. Unlike methods such as DIoU or EIoU, which improve detection performance by adding new geometric constraints, Inner-CIoU addresses the inherent limitations of IoU loss itself. It accelerates regression by introducing auxiliary bounding boxes without adding new loss terms and can be integrated with existing IoU-based loss functions to achieve more precise bounding box fitting, thereby enhancing detection performance. Specifically, Inner CIoU not only accelerates the model's convergence process but also achieves improvements in key performance metrics,

including mAP50 and mAP50:95. This enhancement further validates the superiority of Inner CIoU in handling complex scenes and target detection tasks.

Table 5. Results and comparative experiments of the Inner-IoU method.

Methods	mAP50	mAP50-95
baseline (CIoU)	98.75	97.5
baseline + EIoU	97.86	95.76
baseline + SIoU	97.42	96.35
baseline + DIoU	98.52	97.76
baseline + Shape-IoU	98.43	98.45
baseline + Inner CIoU	99.18	98.87

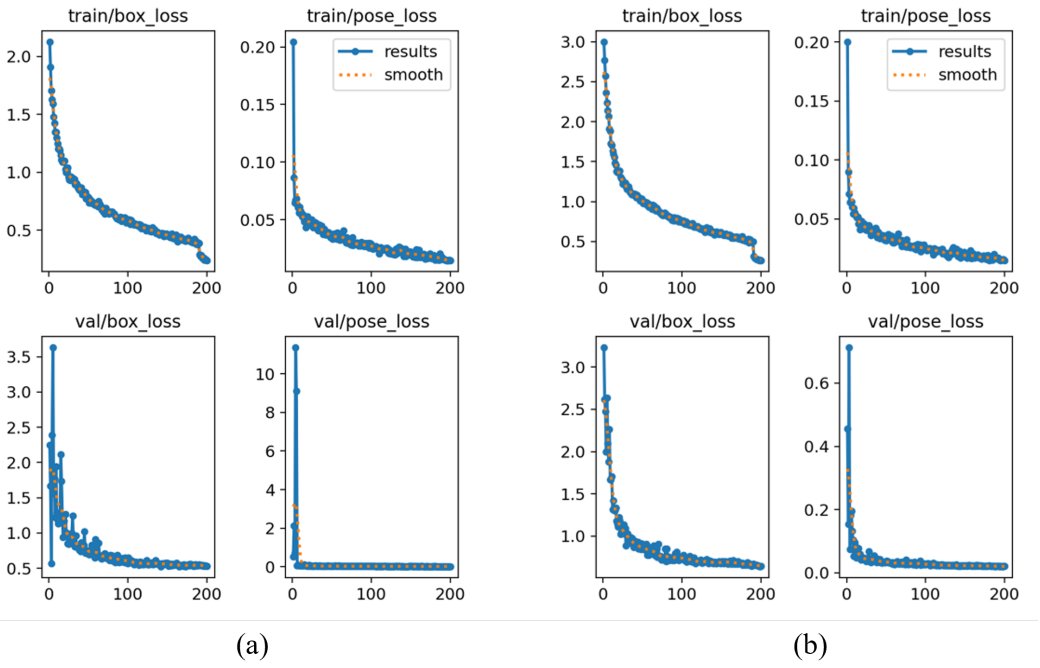


Figure 15. Training loss function value curves: (**a**) YOLOv8-Pose loss function curve; (**b**) YOLO-PP loss function curve Abscissa, iteration times, and ordinate, loss value.

6. Device Deployment and Interactive Interface

This study aims to provide key technological support for cherry tomato-picking robots. To achieve this, we designed YOLO-PP and plan to deploy it on embedded devices equipped with an intuitive interactive interface, enabling non-professionals to use it with ease.

In our previous work, we developed harvesting equipment for single tomato picking. We aim to explore algorithm and software development based on these existing harvesting robot hardware platforms and operational scenarios, which will then guide the specific design and improvements of the robot. Specifically, we set up a software environment for running YOLO-PP on the NVIDIA Jetson Xavier NX device and tested the detection speed of the single-stage keypoint model on the device using images from the test set. We evaluated DEKR, YOLO-Pose, YOLO-PP, and the TensorRT FP16-quantized version of YOLO-PP. Table 6 presents a comparison of the inference speeds of these models on the Jetson Xavier NX for the same task. The inference time for a single image is calculated as the average over 30 images. After FP16 quantization with TensorRT, YOLO-PP achieves an inference time of 31.64 milliseconds per image and an FPS of 31.24, representing a 3.6-fold improvement in inference speed compared to the non-quantized version. Furthermore, the inference

time is reduced by 84.50% and 83.95% compared to the non-quantized YOLO-Pose and YOLO-PP, respectively, meeting the requirements for real-time detection. Figure 16 shows screenshots of the actual tests, with the left side displaying the test device data and the right side showing the predicted picking points on random images.

Table 6. Comparison of average inference time results.

Model	Inference Time	FPS
DEKR	283.17 ms	11.3
YOLO-Pose	204.20 ms	8.63
YOLO-PP (Unquantized)	197.13 ms	7.85
YOLO-PP (Quantified)	31.64 ms	31.24

We developed a human–machine interaction interface based on the PySide6 framework, specifically designed for detecting picking points, to enable non-professionals in the agricultural field to quickly and effectively utilize this detection model in scenarios such as Figure 17a. Since the complete device is still under development, the interface does not include actual picking actions. It has been functionally tested on the Jetson Xavier NX embedded device, and the detailed interface functions and layout are shown in Figure 17b. The left side of the interface is the detection result display panel, while the right side contains the operation and information panels. Specifically, the information panel includes the Equipment Information module and the detection result statistics module. The Equipment Information module dynamically checks whether the camera and battery are connected to the device, as well as the storage space usage level and the remaining battery level, with the battery-related controls being reserved. The detection result statistics module displays the number of categories, the number of targets, real-time FPS, and the currently selected model weight file. The operation panel consists of three modules: the model selection module, the detection module, and the control module. Currently, three presets are provided for model selection, including YOLO-Pose, YOLOv8-Pose, and YOLO-PP, with built-in weight files provided for each model. The detection module is divided into photo detection based on local storage and video detection after connecting the camera. The video detection button is only clickable when the camera is shown as connected in the Equipment Information module. After loading an image or video, the next steps are performed using the four buttons in the control module: start, pause, setting, and quit. In the setting module, parameters such as confidence threshold and IOU threshold for model inference can be configured. Most importantly, all results are displayed in real-time on the detection result display panel on the left side of the interface, allowing humans to visually monitor the process in real-time for subsequent development work and to prevent unexpected issues. From the perspective of a non-professional user, I will simulate the usage scenario of this detection software in an actual agricultural facility environment: Step 1: Confirm the device connection status in the Equipment Information panel. Step 2: Select the model for picking point detection and load the pre-trained weights to detect the picking points of cherry tomatoes. Step 3: Choose an image, video, or input from a depth sensor as the detection target, then click the start button. Step 4: Display the detection statistics in the information panel and save the detection results to the local hard drive.

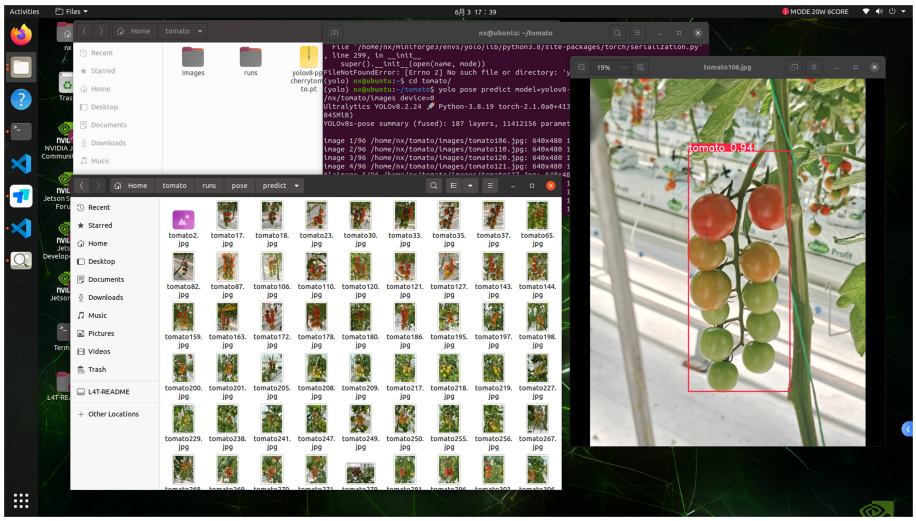
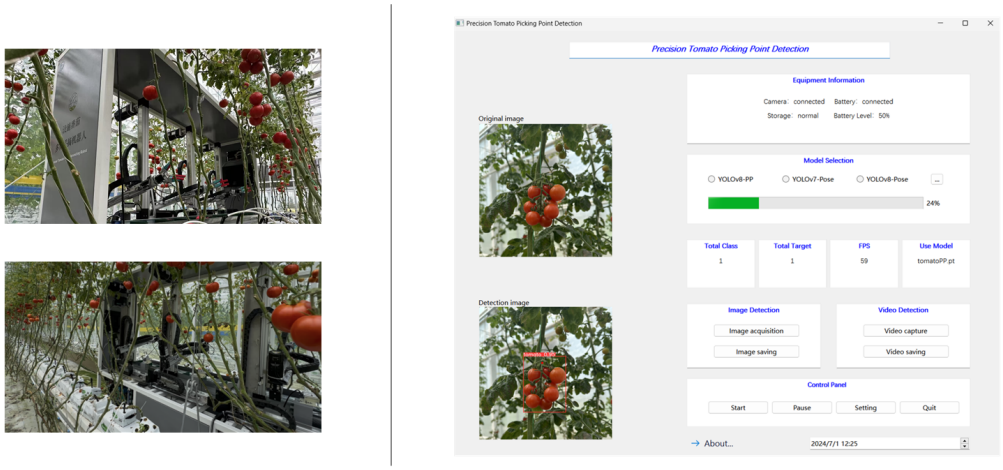


Figure 16. Actual screen of device deployment.



(a) The harvesting robot hardware platform and operational scenarios.

(b) Human-Machine Interactive Interface for Automated Tomato Harvesting Robots

Figure 17. Hardware platform and software implementation of the automated tomato harvesting robot.

7. Conclusions

In recent years, there has been a significant focus on the large-scale development of protected agriculture systems with the aim of automating the production process of fruits and vegetables as much as possible, with cherry tomatoes representing a key area of interest. The high planting density of cherry tomatoes, intertwined branches, and difficulty in identifying tomato clusters and stems have made it essential to develop a robust computer vision method to enhance the efficiency of cherry tomato stem detection. To address this issue, we have developed a novel cherry tomato picking point detection method, YOLO-PP, which has been designed for use with shear-type picking actuators. This method employs a top-down pose estimation approach, integrating object and keypoint detection. This innovative solution enables the simultaneous detection of cherry tomatoes and picking points in complex environments, even under limited computational resources.

The model incorporates the C2FET module to combine local and global perceptual capabilities and employs the innovative SPSP interactive neck structure to effectively diffuse and fuse multi-scale features, reconstructing fine-grained hierarchical semantic information. This innovative approach allows the model to process data at multiple levels of detail, providing a comprehensive understanding of the input data. Extensive ablation and

comparative experiments have demonstrated that our method achieves higher detection accuracy than traditional keypoint detection models. The model has been tested on the NVIDIA Jetson Xavier NX, demonstrating high accuracy and efficiency, making it an ideal choice for use in picking robots. In addition, based on existing hardware and extensive real-world automated harvesting experience, we have developed an intuitive and user-friendly interface, enabling non-expert users to easily deploy the model. Our follow-up work primarily revolves around three themes: a vision positioning system based on YOLO-PP, a cutting-style end-effector, and research on harvesting other types of clustered fruits. Firstly, our vision positioning system includes a wide-angle main camera configured as "eye-to-hand" and multiple sub-cameras configured as "eye-in-hand". The main camera is used to detect pre-selected targets for harvesting within a large range and to guide the robotic arm to approach the target. The sub-cameras, equipped with Realsense D435i depth cameras in an "eye-in-hand" configuration, capture more accurate and detailed images and utilize our designed YOLO-PP model to ultimately output picking points. Secondly, we will focus on developing a customized harvesting end-effector capable of gripping the fruit while cutting and accurately placing it in the designated location. Finally, for other cluster-grown crops similar to cherry tomatoes, such as grapes and olives, our design approach and detection model are equally applicable, requiring only sufficient targeted samples and labels. The vision positioning system and end-effector requirements are also similar, but new structures need to be designed according to specific environments. We plan to explore a modular main framework to enhance the system's reusability and adaptability. These efforts will help address existing challenges more effectively and offer more dependable solutions for real-world applications.

Author Contributions: X.Q. and J.C. conducted the data analysis and wrote the manuscript. Y.Z. contributed to the conception of the study. T.D. made significant contributions to the analysis and manuscript preparation. H.C. completed the revision and editing of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (Grants: 42175157 and 42475151), the Science and Technology Development Fund Project of Wuxi (Grant: N20221002), and the Postgraduate Research & Practice Innovation Program of Jiangsu Province (Grant: SJCX24_0466).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The second author, Jingxing Cao, is affiliated with Wuxi Siasun Robot and Automation Company Limited, which could be considered as a potential conflicts of interest. However, the authors assert that this affiliation has not influenced the objectivity or integrity of the research. Furthermore, they confirm that the study was carried out without any commercial or financial relationships that might be perceived as a conflicts of interest.

References

1. Mohamed, A.; Shaw-Sutton, J.; Green, B.; Andrews, W.; Rolley-Parnell, E.; Zhou, Y.; Zhou, P.; Mao, X.; Fuller, M.; Stoelen, M. Soft manipulator robot for selective tomato harvesting. In *Precision Agriculture'19*; Wageningen Academic: Wageningen, The Netherlands, 2019; pp. 799–805.
2. Maureira, F.; Rajagopalan, K.; Stöckle, C.O. Evaluating tomato production in open-field and high-tech greenhouse systems. *J. Clean. Prod.* **2022**, 337, *130459. [CrossRef]*
3. Wang, Z.; Xun, Y.; Wang, Y.; Yang, Q. Review of smart robots for fruit and vegetable picking in agriculture. *Int. J. Agric. Biol. Eng.* **2022**, *15*, 33–54.
4. Zhou, H.; Wang, X.; Au, W.; Kang, H.; Chen, C. Intelligent robots for fruit harvesting: Recent developments and future challenges. *Precis. Agric.* **2022**, *23*, 1856–1907. [CrossRef]
5. Lu, S.; Xiao, X. Neuromorphic Computing for Smart Agriculture. *Agriculture* **2024**, *14*, 1977. [CrossRef]

6. Li, Y.; Feng, Q.; Li, T.; Xie, F.; Liu, C.; Xiong, Z. Advance of target visual information acquisition technology for fresh fruit robotic harvesting: A review. *Agronomy* **2022**, *12*, 1336. [CrossRef]
7. Li, Z.; Yuan, X.; Wang, C. A review on structural development and recognition–localization methods for end-effector of fruit—Vegetable picking robots. *Int. J. Adv. Robot. Syst.* **2022**, *19*, 17298806221104906. [CrossRef]
8. Li, M.; Wu, F.; Wang, F.; Zou, T.; Li, M.; Xiao, X. CNN-MLP-Based Configurable Robotic Arm for Smart Agriculture. *Agriculture* **2024**, *14*, 1624. [CrossRef]
9. Tang, Y.; Chen, M.; Wang, C.; Luo, L.; Li, J.; Lian, G.; Zou, X. Recognition and localization methods for vision-based fruit picking robots: A review. *Front. Plant Sci.* **2020**, *11*, 510. [CrossRef]
10. Zhang, B.; Xie, Y.; Zhou, J.; Wang, K.; Zhang, Z. State-of-the-art robotic grippers, grasping and control strategies, as well as their applications in agricultural robots: A review. *Comput. Electron. Agric.* **2020**, *177*, 105694. [CrossRef]
11. Fu, L.; Gao, F.; Wu, J.; Li, R.; Karkee, M.; Zhang, Q. Application of consumer RGB-D cameras for fruit detection and localization in field: A critical review. *Comput. Electron. Agric.* **2020**, *177*, 105687. [CrossRef]
12. Saleem, M.H.; Potgieter, J.; Arif, K.M. Automation in agriculture by machine and deep learning techniques: A review of recent developments. *Precis. Agric.* **2021**, *22*, 2053–2091. [CrossRef]
13. Meshram, V.; Patil, K.; Meshram, V.; Hanchate, D.; Ramkteke, S. Machine learning in agriculture domain: A state-of-art survey. *Artif. Intell. Life Sci.* **2021**, *1*, 100010. [CrossRef]
14. Xiang, R.; Jiang, H.; Ying, Y. Recognition of clustered tomatoes based on binocular stereo vision. *Comput. Electron. Agric.* **2014**, *106*, 75–90. [CrossRef]
15. Luo, L.; Tang, Y.; Lu, Q.; Chen, X.; Zhang, P.; Zou, X. A vision methodology for harvesting robot to detect cutting points on peduncles of double overlapping grape clusters in a vineyard. *Comput. Ind.* **2018**, *99*, 130–139. [CrossRef]
16. Luo, L.; Tang, Y.; Zou, X.; Ye, M.; Feng, W.; Li, G. Vision-based extraction of spatial information in grape clusters for harvesting robots. *Biosyst. Eng.* **2016**, *151*, 90–104. [CrossRef]
17. Luo, L.; Liu, W.; Lu, Q.; Wang, J.; Wen, W.; Yan, D.; Tang, Y. Grape berry detection and size measurement based on edge image processing and geometric morphology. *Machines* **2021**, *9*, 233. [CrossRef]
18. Pérez-Zavala, R.; Torres-Torriti, M.; Cheein, F.A.; Troni, G. A pattern recognition strategy for visual grape bunch detection in vineyards. *Comput. Electron. Agric.* **2018**, *151*, 136–149. [CrossRef]
19. Behroozi-Khazaei, N.; Maleki, M.R. A robust algorithm based on color features for grape cluster segmentation. *Comput. Electron. Agric.* **2017**, *142*, 41–49. [CrossRef]
20. Bai, Y.; Mao, S.; Zhou, J.; Zhang, B. Clustered tomato detection and picking point location using machine learning-aided image analysis for automatic robotic harvesting. *Precis. Agric.* **2023**, *24*, 727–743. [CrossRef]
21. Jin, Y.; Yu, C.; Yin, J.; Yang, S.X. Detection method for table grape ears and stems based on a far-close-range combined vision system and hand-eye-coordinated picking test. *Comput. Electron. Agric.* **2022**, *202*, 107364. [CrossRef]
22. Tang, Y.; Qiu, J.; Zhang, Y.; Wu, D.; Cao, Y.; Zhao, K.; Zhu, L. Optimization strategies of fruit detection to overcome the challenge of unstructured background in field orchard environment: A review. *Precis. Agric.* **2023**, *24*, 1183–1219. [CrossRef]
23. Tian, Y.; Yang, G.; Wang, Z.; Li, E.; Liang, Z. Detection of apple lesions in orchards based on deep learning methods of CycleGAN and YOLOV3-dense. *J. Sens.* **2019**, *2019*, 7630926. [CrossRef]
24. Yan, C.; Chen, Z.; Li, Z.; Liu, R.; Li, Y.; Xiao, H.; Lu, P.; Xie, B. Tea sprout picking point identification based on improved DeepLabV3+. *Agriculture* **2022**, *12*, 1594. [CrossRef]
25. Sa, I.; Ge, Z.; Dayoub, F.; Upcroft, B.; Perez, T.; McCool, C. Deepfruits: A fruit detection system using deep neural networks. *Sensors* **2016**, *16*, 1222. [CrossRef]
26. Wu, F.; Duan, J.; Chen, S.; Ye, Y.; Ai, P.; Yang, Z. Multi-target recognition of bananas and automatic positioning for the inflorescence axis cutting point. *Front. Plant Sci.* **2021**, *12*, 705021. [CrossRef] [PubMed]
27. Zhang, Q.; Chen, J.; Li, B.; Xu, C. Method for recognizing and locating tomato cluster picking points based on RGB-D information fusion and target detection. *Trans. Chin. Soc. Agric. Eng. (Trans. CSAE)* **2021**, *37*, 143–152.
28. Li, D.; Sun, X.; Lv, S.; Elkhouchlaa, H.; Jia, Y.; Yao, Z.; Lin, P.; Zhou, H.; Zhou, Z.; Shen, J.; et al. A novel approach for the 3D localization of branch picking points based on deep learning applied to longan harvesting UAVs. *Comput. Electron. Agric.* **2022**, *199*, 107191. [CrossRef]
29. Qi, X.; Dong, J.; Lan, Y.; Zhu, H. Method for identifying litchi picking position based on YOLOv5 and PSPNet. *Remote Sens.* **2022**, *14*, 2004. [CrossRef]
30. Zhang, T.; Wu, F.; Wang, M.; Chen, Z.; Li, L.; Zou, X. Grape-bunch identification and location of picking points on occluded fruit axis based on YOLOv5-GAP. *Horticulturae* **2023**, *9*, 498. [CrossRef]
31. Ding, J.; Niu, S.; Nie, Z.; Zhu, W. Research on Human Posture Estimation Algorithm Based on YOLO-Pose. *Sensors* **2024**, *24*, 3036. [CrossRef]
32. Pavlov, M.; Marakhtanov, A.; Korzun, D. Detection of Key Points for a Rainbow Trout in Underwater Video Surveillance System. In Proceedings of the 33rd Conference of FRUCT Association, Zilina, Slovakia, 24–26 May 2023.

33. Tan, J.; Qin, H.; Chen, X.; Li, J.; Li, Y.; Li, B.; Leng, Y.; Fu, C. Point cloud segmentation of breast ultrasound regions to be scanned by fusing 2D image instance segmentation and keypoint detection. In Proceedings of the 2023 International Conference on Advanced Robotics and Mechatronics (ICARM), Sanya, China, 8–10 July 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 669–674.
34. Nguyen, T.D.T.; Nguyen, M.H.; Nguyen, T.H.; Pham, V.C. Deep Learning Based Pose Estimation and Action Prediction for Construction Machines. In Proceedings of the 2023 8th International Scientific Conference on Applying New Technology in Green Buildings (ATiGB), Danang, Vietnam, 10–11 November 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 268–273.
35. Wu, Z.; Xia, F.; Zhou, S.; Xu, D. A method for identifying grape stems using keypoints. *Comput. Electron. Agric.* **2023**, *209*, 107825. [CrossRef]
36. Chen, J.; Ma, A.; Huang, L.; Li, H.; Zhang, H.; Huang, Y.; Zhu, T. Efficient and lightweight grape and picking point synchronous detection model based on key point detection. *Comput. Electron. Agric.* **2024**, *217*, 108612. [CrossRef]
37. Tzutalin, D. tzutalin/labelimg: Labelimg is a Graphical Image Annotation Tool and Label Object Bounding Boxes in Images. 2018. Available online: https://github.com/wkentaro/labelme (accessed on 8 December 2024).
38. Maji, D.; Nagori, S.; Mathew, M.; Poddar, D. Yolo-pose: Enhancing yolo for multi person pose estimation using object keypoint similarity loss. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–20 June 2022; pp. 2637–2646.
39. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
40. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
41. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
42. Zhang, H.; Xu, C.; Zhang, S. Inner-IoU: More effective intersection over union loss with auxiliary bounding box. *arXiv* **2023**, arXiv:2311.02877.
43. Jocher, G.; Chaurasia, A.; Qiu, J. Ultralytics YOLO. 2023. Available online: https://github.com/ultralytics/ultralytics (accessed on 8 December 2024).
44. Liu, X.; Peng, H.; Zheng, N.; Yang, Y.; Hu, H.; Yuan, Y. Efficientvit: Memory efficient vision transformer with cascaded group attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 14420–14430.
45. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 658–666.
46. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 12993–13000. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.