*horticulturae*

MDPI

*Article*

# Design and Optimization of Target Detection and 3D Localization Models for Intelligent Muskmelon Pollination Robots

**Huamin Zhao [1,*], Shengpeng Xu [1], Weiqi Yan [2], Defang Xu [3,*], Yongzhuo Zhang [1], Linjun Jiang [1], Yabo Zheng [1], Erkang Zeng [1] and Rui Ren [1]**

[1] College of Agricultural Engineering, Shanxi Agricultural University, Jinzhong 030801, China; 20233756@stu.sxau.edu.cn (S.X.); 202430795@stu.sxau.edu.cn (Y.Z.); 202430041@stu.sxau.edu.cn (L.J.); 202430037@stu.sxau.edu.cn (E.Z.); b20221003@stu.sxau.edu.cn (R.R.)
[2] School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand; weiqi.yan@aut.ac.nz
[3] Department of Mathematics and Artificial Intelligence, Lvliang University, Lvliang 033001, China
* Correspondence: zhaohuamin@sxau.edu.cn (H.Z.); 20211018@llu.edu.cn (D.X.)

## Abstract

With the expansion of muskmelon cultivation, manual pollination is increasingly inadequate for sustaining industry development. Therefore, the development of automatic pollination robots holds significant importance in improving pollination efficiency and reducing labor dependency. Accurate flower detection and localization is a key technology for enabling automated pollination robots. In this study, the YOLO-MDL model was developed as an enhancement of YOLOv7 to achieve real-time detection and localization of muskmelon flowers. This approach adds a Coordinate Attention (CA) module to focus on relevant channel information and a Contextual Transformer (CoT) module to leverage contextual relationships among input tokens, enhancing the model's visual representation. The pollination robot converts the 2D coordinates into 3D coordinates using a depth camera and conducts experiments on real-time detection and localization of muskmelon flowers in a greenhouse. The YOLO-MDL model was deployed in ROS to control a robotic arm for automatic pollination, verifying the accuracy of flower detection and measurement localization errors. The results indicate that the YOLO-MDL model enhances AP and F1 scores by 3.3% and 1.8%, respectively, compared to the original model. It achieves AP and F1 scores of 91.2% and 85.1%, demonstrating a clear advantage in accuracy over other models. In the localization experiments, smaller errors were revealed in all three directions. The RMSE values were 0.36 mm for the X-axis, 1.26 mm for the Y-axis, and 3.87 mm for the Z-axis. The YOLO-MDL model proposed in this study demonstrates strong performance in detecting and localizing muskmelon flowers. Based on this model, the robot can execute more precise automatic pollination and provide technical support for the future deployment of automatic pollination robots in muskmelon cultivation.

**Keywords:** muskmelon; YOLO-MDL; three-dimensional localization; muskmelon automatic pollination robot



Academic Editor: Weikuan Jia

Received: 3 July 2025
Revised: 23 July 2025
Accepted: 29 July 2025
Published: 4 August 2025

**Citation:** Zhao, H.; Xu, S.; Yan, W.; Xu, D.; Zhang, Y.; Jiang, L.; Zheng, Y.; Zeng, E.; Ren, R. Design and Optimization of Target Detection and 3D Localization Models for Intelligent Muskmelon Pollination Robots. *Horticulturae* **2025**, *11*, 905. https://doi.org/10.3390/horticulturae11080905



## 1. Introduction

Muskmelon (*Cucumis melo* L.), commonly known as cantaloupe, is an annual creeping or climbing herb belonging to the genus Cucumis in the family Cucurbitaceae [1].

Muskmelon possesses a distinctive aromatic scent and is rich in nutrients, including water, soluble solids, and vitamins [2,3].

Currently, muskmelon cultivation primarily occurs in greenhouses due to space constraints. While small agricultural machinery can be used for plowing, subsequent tasks such as transplanting, spraying, pollination, and harvesting must still be performed manually [4]. Pollination is a crucial aspect of muskmelon cultivation, significantly influencing fruit set rates and yield. This process requires precise and meticulous execution [5]. The expansion of muskmelon cultivation and the growing demand for labor have rendered artificial pollination insufficient to support industry development. With advancements in smart agriculture, enhancing pollination efficiency and reducing labor dependence through automated pollination technology are of significant importance.

Among these technologies, precise detection and localization algorithms are crucial for enabling automated pollination robots, enhancing pollination efficiency, and providing a technical foundation for their implementation. In recent years, significant progress has been made in target and flower detection using deep learning techniques [6,7].

Target detection algorithms can generally be classified into two major categories. The first category comprises region-based convolutional neural networks, including R-CNN (Region-CNN) [8], Fast R-CNN [9], Faster R-CNN [10], and Mask R-CNN [11]. Lin et al. [12] integrated Faster R-CNN with VGG19 to detect strawberry flowers under varying camera viewpoints, distances, and complex backgrounds. The second category approaches target localization as a regression problem and includes single-point multi-frame detectors [13] and single-shot detectors (YOLO) [14–17]. Tian et al. [18] employed SSD algorithms for flower recognition and detection on the Oxford University Flower Dataset. The experimental results indicated that the average precision (AP) for the VOC2007 and VOC2012 datasets was 83.64% and 87.4%, respectively, with a processing time of 0.13 s per image. Li et al. [19] leveraged transfer learning with YOLOv4 to enable fast and accurate detection of kiwifruit flowers and buds in an orchard. The AP for kiwifruit flower and bud detection reached 96.66% and 98.57%, respectively. Chen et al. [20] employed SSD algorithms for flower recognition and detection on the Oxford University Flower Dataset to estimate orchard flowering levels and determine the peak flowering period. Apple flowers were identified using a YOLOv5s model incorporating a coordinated attention layer and a small-target detection layer, achieving an average accuracy of 77.5%. This approach provided critical decision-making data for apple flower detection and optimal flower thinning timing.

In the field of agriculture, recent studies on YOLO structure upgrades also offer valuable insights. Akdoğan et al. [21] investigated the application of an advanced YOLO-based model for detecting cherry and apple trees in agricultural environments using UAV-captured imagery. They proposed a PP-YOLO model that integrates a spatial attention mechanism with image preprocessing techniques to enhance feature representation and reduce noise. Experimental results demonstrated that, compared to the baseline model, PP-YOLO achieved superior performance, with a 1.5% improvement in the F1 score and a 1.4% increase in mAP50. This study highlights the effectiveness of attention modules and preprocessing techniques in improving detection accuracy for crops in complex agricultural scenes. Nguyen et al. [22] proposed a novel and lightweight object detection model tailored for banana-harvesting robots. Based on the YOLOv8n architecture, their model incorporates a group-shuffle convolution module, a C2f-fast channel attention block, and BiFPN. Experimental results showed that the model significantly outperformed the baseline in terms of precision, recall, and mAP. Moreover, the model was highly optimized, with a 43.3% reduction in parameter count and a 40.3% reduction in model size, making it well-

suited for deployment on embedded systems in agricultural robots. These improvements enable a better balance between detection accuracy and computational efficiency.

With advancements in this technology, flower target detection is increasingly integrated into sophisticated agricultural robotic systems, enabling automated operations such as precision pollination and flower thinning. William et al. [23] developed a kiwifruit pollination robot by integrating a deep learning model with a Faster R-CNN algorithm trained on a kiwifruit flower dataset, achieving a model mAP of 85.3%. The integration of an air-assisted sprayer with a flower-specific precision pollination system effectively enhances kiwifruit pollination efficiency. Gao et al. [24] developed a pollination robot for precision pollination of kiwifruit flowers, incorporating a multi-class flower detection method based on YOLOv5l in the vision system. The robot is equipped with a three-degree-of-freedom robotic arm and an air-liquid injection system. Field experiments conducted in the orchard demonstrated an average pollination success rate of 99.3% and an average fruit set rate of 88.5%. This approach enhances kiwifruit pollen utilization and reduces pollen waste. Wen et al. [25] employed 3D vision for tomato flower identification and localization, utilizing YOLOv4 for target detection, achieving a mAP of 97.67%. Subsequently, active alignment was integrated with PCL to acquire high-precision spatial point cloud data and determine the pollination center of mass coordinates within the 3D bounding box of the flower bouquet using a bi-directional averaging method. Yu et al. [26] enhanced YOLOv5s for tomato flower detection by incorporating CBAM into the network and employing weighted box fusion (WBF), achieving a model mAP@0.5 of 96.8%, which significantly reduces target leakage and misdetection. The improved algorithm was deployed on a self-developed pollination robot for experimental validation. Through multiple repeated experiments, the tomato flower detection system utilized coordinate transformation to rapidly locate targets, ultimately enabling precise robotic arm actuation for accurate pollination. Ahmad et al. [27] introduced a novel method for automated watermelon pollination leveraging visually intelligent guided servo control. The deep learning inference mechanism was utilized to estimate the flower's size and orientation. Across 50 experiments, the results demonstrated a high detection rate with a mAP of 90.9%. The average depth error of pollination target localization was only 1.028 cm, and the pollination speed was 8 s per flower on average, which is feasible in practical applications.

In summary, the deep learning-based flower target detection method offers high efficiency and excellent reliability in the complex environment of flowers, but there has not been a breakthrough in the research on automatic pollination of muskmelon flowers, which makes it difficult to achieve low-cost and high-precision automatic pollination. To enable accurate pollination by an automatic muskmelon pollination robot, this study constructs a muskmelon flower image dataset and proposes a method that combines an improved YOLOv7-based muskmelon flower detection model (YOLO-MDL) with a depth camera-based three-dimensional localization technique to achieve accurate pollination of muskmelon flowers.

(1) To enhance the model's focus on relevant channel information, the CA (Coordinate Attention) attention mechanism is introduced, and its optimal integration location is examined. Additionally, the effects of different attention mechanisms on model performance are compared.
(2) The CoT (Contextual Transformer) module is incorporated into the model, enabling full utilization of contextual information between input keys to guide the learning of the dynamic attention matrix, thereby enhancing the model's visual representation. Furthermore, the impact of its integration location on accuracy is examined.
(3) The optimal integration of both the CA attention mechanism and the CoT module is explored to develop the final YOLO-MDL model. Comparative experiments are

conducted to evaluate the improved model against other YOLO-series detection models, verifying its effectiveness.

(4) The YOLO-MDL model is deployed in ROS (Robot Operating System) to facilitate robot hardware implementation. The depth camera-based positioning method is integrated, enabling real-time detection and localization of muskmelon flowers. RMSE is employed as the error analysis metric to assess the positioning accuracy of robotic arm pollination.

## 2. Materials and Methods

### *2.1. Image Acquisition and Dataset Construction*

The muskmelon flower image dataset used in this study was collected from a muskmelon cultivation greenhouse in Taigu District, Jinzhong City, Shanxi Province, China. Images were captured at various times between 8:00 and 18:00 to account for different lighting conditions. To enhance dataset diversity, sample images were taken under varying illumination and viewing angles, including downlighting, backlighting, as well as near and far distances, as illustrated in Figure 1.


**Figure 1.** Sample data collected. (**a**) with light; (**b**) against light; (**c**) long distance.

A total of 836 images were manually labeled using LabelImg (version 1.8.6), with the targets enclosed by the smallest external rectangular bounding boxes. Only fully bloomed flowers suitable for pollination were labeled, while buds that were not fully open, as well as wilted flowers, were excluded from labeling. In this study, a randomized partitioning strategy was employed to divide the dataset into three subsets with an 8:1:1 ratio: the training set (668 images); the validation set (84 images); and the test set (84 images). The training set was used for model training, the validation set for hyperparameter tuning and initial evaluation of model performance, and the test set for assessing detection accuracy and generalization capability.

### *2.2. Construction of the YOLO-MDL Model*

#### 2.2.1. Construction of CA Module

Attention mechanism is one of the core techniques of deep learning, and its basic idea is to divide the input data into multiple parts and calculate how much each part contributes to the output of the model. At each time step, the model will weight different input parts according to their contributions, and then use the weighted results to obtain the output results, which ensures that the model can focus on the most relevant parts and ignore the irrelevant parts.

CA attention mechanism [28] is an attention mechanism that can distinguish spatial directions (i.e., coordinates) and generate coordinate-aware feature maps. By embedding location information into the channel attention, it allows lightweight networks to focus on information over a larger region while avoiding incurring a large computational overhead. To mitigate the loss of location information due to 2D global pooling, the channel attention is decomposed into two parallel 1D feature encoding processes, which effectively integrate spatial coordinate information into the generated attention map. As illustrated in Figure 2, by using the Squeeze-and-Excitation (SE) attention mechanism, which employs 2D pooling, as a comparative example against the CA attention mechanism, the structured diagram clearly shows the differences between the two approaches.

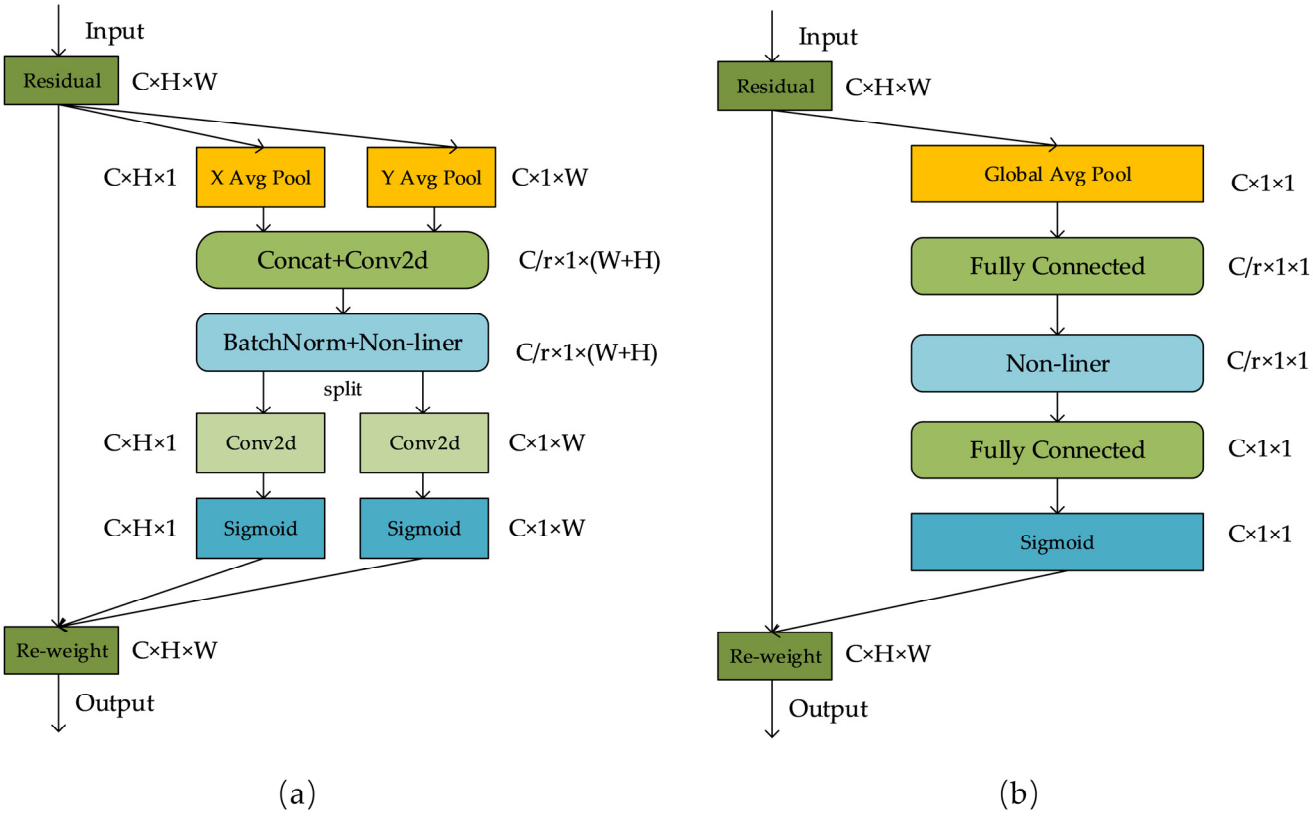


**Figure 2.** The structure of CA and SE. (**a**) CA attention mechanism using two parallel 1D feature encodings; (**b**) SE attention mechanism using 2D pooling.

The CA attention mechanism operates through two key stages: Coordinate Information Embedding and Coordinate Attention Generation. In the first stage, instead of using conventional global pooling, a decomposition strategy is applied. As shown in Equation (1), the global pooling operation is broken down into two one-dimensional feature encoding steps, allowing the model to retain precise location information while capturing broader spatial interactions.

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} x_{c(i,j)} \quad (1)$$

Specifically, given the input X, each channel is first encoded along the horizontal and vertical coordinates using a pooling kernel of dimensions (H,1) or (1,W), respectively. As a

result, the outputs of the cth channel with height h and the cth channel with width w are obtained by Equation (2) and Equation (3), respectively.

$$z_c^h(h) = \frac{1}{W} \sum_{0 \le i \le W} x_c(h, j) \tag{2}$$

$$z_c^w(w) = \frac{1}{H} \sum_{0 \le j < H} x_c(j, w) \tag{3}$$

After the Coordinate Information Embedding step, the transformed features are concatenated and passed through a convolutional transformation function. First, spatial information along the vertical and horizontal directions is encoded through batch normalization and nonlinear activation functions. Subsequently, the encoded spatial dimensions are decomposed into two separate feature maps corresponding to the horizontal and vertical directions. These feature maps are then processed using a $1 \times 1$ convolutional transformation function, generating two tensors with identical channel numbers. The final output, denoted as f, is computed as shown in Equation (4).

$$f = \delta\left(F_1\left(\left[z^h, z^w\right]\right)\right) \tag{4}$$

where $\delta$ denotes a nonlinear activation function and $f \in R^{C \times H \times W}$ is the intermediate feature map to encode spatial information in the horizontal and vertical directions. By splitting f along the spatial dimension into two independent tensors $f^h \in R^{C/r \times H}$ and $f^w \in R^{C/r \times W}$, two $1 \times 1$ convolutional transforms, $F_h$ and $F_w$, transform $f^h$ and $f^w$ into tensors $g^h$ and $g^w$, which have the same number of channels as the input X. The computation process is shown in Equation (5) and Equation (6), respectively.

$$g^h = \sigma\left(F_h\left(f^h\right)\right) \tag{5}$$

$$g^w = \sigma(F_w(f^w)) \tag{6}$$

where $\sigma$ is the sigmoid activation function and the output $g^h$ and $g^w$ are augmented as attention weights. Finally, the output of Coordinate Attention Block can be expressed as Equation (7).

$$y_c(i, j) = x_c(i, j) \times g_c^h(i) \times g_c^w(j) \tag{7}$$

2.2.2. Construction of CoT Module

Transformers based on self-attention mechanisms have led to significant advancements in Natural Language Processing (NLP) and have influenced the development of Transformer-inspired architectures in computer vision. These models have demonstrated strong performance across various visual tasks.

However, most current approaches apply self-attention directly to 2D feature maps, generating attention matrices based on independent query and key-value pairs at each spatial position, without fully exploiting the contextual relationships among neighboring keys. In contrast, the Contextual Transformer (CoT) module [29] leverages the context between input keys to dynamically generate attention matrices, thereby improving visual feature representation. The architecture of the CoT module is illustrated in Figure 3.

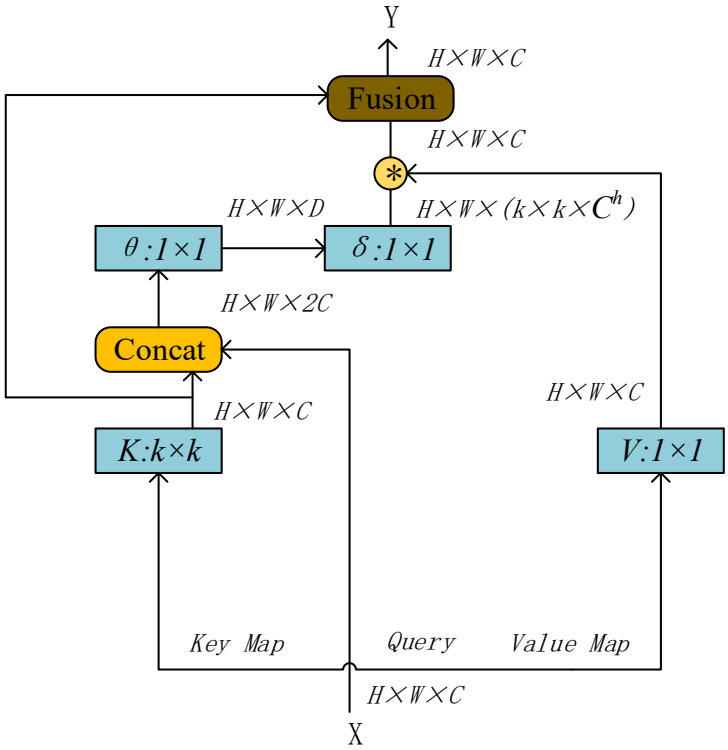


**Figure 3.** Schematic diagram of CoT structure.

The CoT module first encodes the input keys by applying convolution within a contextual framework, generating a static contextual representation of the input. The encoded keys are then concatenated with the input query, and the learned attention matrix is applied to the input values through two successive convolutional operations to obtain a dynamic contextual representation. Finally, the fusion of both static and dynamic contextual representations serves as the output.

Specifically, when the input feature map is defined as K = $XW_k$, Q = XWq, and V = XWv for keys, queries, and values, respectively, the CoT module first applies convolution to all neighboring keys within a k × k grid, grouping them into k × k sets to achieve a contextual representation for each key. This process generates the contextual key $K^1 \in R^{H\times W\times C}$, which captures the contextual information among the nearest neighbors. $K^1$ serves as the static contextual representation of the input. Subsequently, $K^1$ is concatenated with Q, and the attention matrix A is computed through two 1 × 1 convolutional operations using $W_\theta$ and $W_\delta$. The computation process is given in Equation (8).

$$A = \left[K^1, Q\right] W_\theta W_\delta \tag{8}$$

In this process, $W_\theta$ employs ReLU as its activation function, while $W_\delta$ does not utilize any activation function. At each spatial location, the attention matrix A is learned from query features and contextual key features, rather than being computed from independent query-key pairs. The contextual key $K_1$ captures static contextual information, thereby enhancing the self-attention mechanism's learning capability. Finally, the computed attention matrix is multiplied by the values to generate the attention feature map $K_2 \in R^{H\times W\times C}$, referred to as the dynamic contextual representation. Since $K_2$ facilitates dynamic feature interactions based on the input, it is designated as a dynamic contextual representation. The final output Y of the CoT module is obtained by fusing the local static contextual representation ($K_1$) and the global dynamic contextual representation ($K_2$) within the attention mechanism.

2.2.3. Construction of YOLO-MDL Detection Model

In the ELAN module within the Backbone, the final CBS layer preceding CatConv is first replaced with the CA attention mechanism module, enabling the model to focus more effectively on relevant channel information. Similarly, in the ELAN module within the Neck—specifically, the CBS layer following CatConv—the CoT module is introduced to fully exploit contextual relationships between input keys, thereby guiding the learning of the dynamic attention matrix and enhancing the model's visual representation.

To differentiate between the ELAN modules in the Backbone and Neck, the ELAN module in the Neck is designated as ELAN-W. Since each of these components contains four ELAN modules and four ELAN-W modules, four positional scenarios were examined for each module. The experimental results indicated that positioning the CA attention mechanism module at A1 yielded the best performance when adding only the CA module, while placing the CoT module at T3 resulted in optimal performance when integrating only the CoT module.

Building upon these findings, the two modules were incorporated simultaneously based on their individual optimal placements. However, this configuration did not yield the best overall performance. Therefore, multiple combinations of module placements were systematically evaluated. The final results demonstrated that positioning the CA attention mechanism module at A3 and the CoT module at T1 provided the most effective performance when both modules were introduced concurrently. The schematic diagram of the YOLO-MDL network architecture is illustrated in Figure 4.

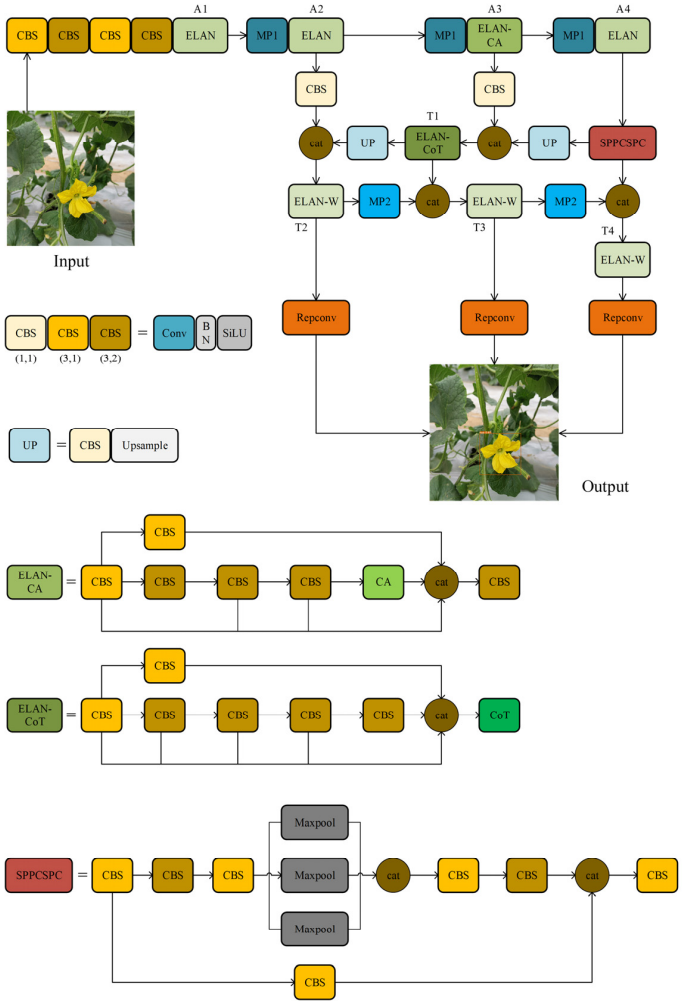


**Figure 4.** Structure of YOLO-MDL.

### *2.3. Experimental Platform*

The model training in this study was conducted on a Windows 10 operating system with the following hardware configuration: Intel Core i5-13400 CPU (2.50 GHz), 32 GB RAM, and NVIDIA Tesla P40 GPU with 24 GB of video memory. The software environment primarily consisted of Python 3.8.19, PyTorch 1.13.0, and CUDA 11.7. For robot control, a separate computer was configured with ROS Noetic, running on the Ubuntu 20.04 operating system.

The initial input image size for the network was 640 × 640 × 3 pixels, and Stochastic Gradient Descent (SGD) Optimization was employed for parameter updates. The parameters were optimized using SGD with a momentum of 0.937, an initial learning rate of 0.01, and a weight decay of 0.0005. The batch size was set to 16, and the model was trained for 200 epochs. To ensure consistency, the same dataset and training parameters were applied across all models during the training process.

The robotic arm used in this study was the Realman RM65-B (manufactured by Realman Corporation, headquartered in Beijing, China), a lightweight, six-degree-of-freedom robotic arm with a body weight of 7.2 kg and a payload capacity of 5.0 kg. It operated at a rated voltage of DC 24 V, with a total power consumption of ≤100 W, a working radius of 610 mm, and a repetitive positioning accuracy of ±0.05 mm. Figure 5 illustrates the muskmelon-pollinating robot used in this study, where the robotic arm, equipped with a depth camera, was mounted on a crawler-type mobile chassis. A small brush head was attached to the actuator at the end of the arm, enabling it to dip into the pollination solution and pollinate muskmelon flowers.

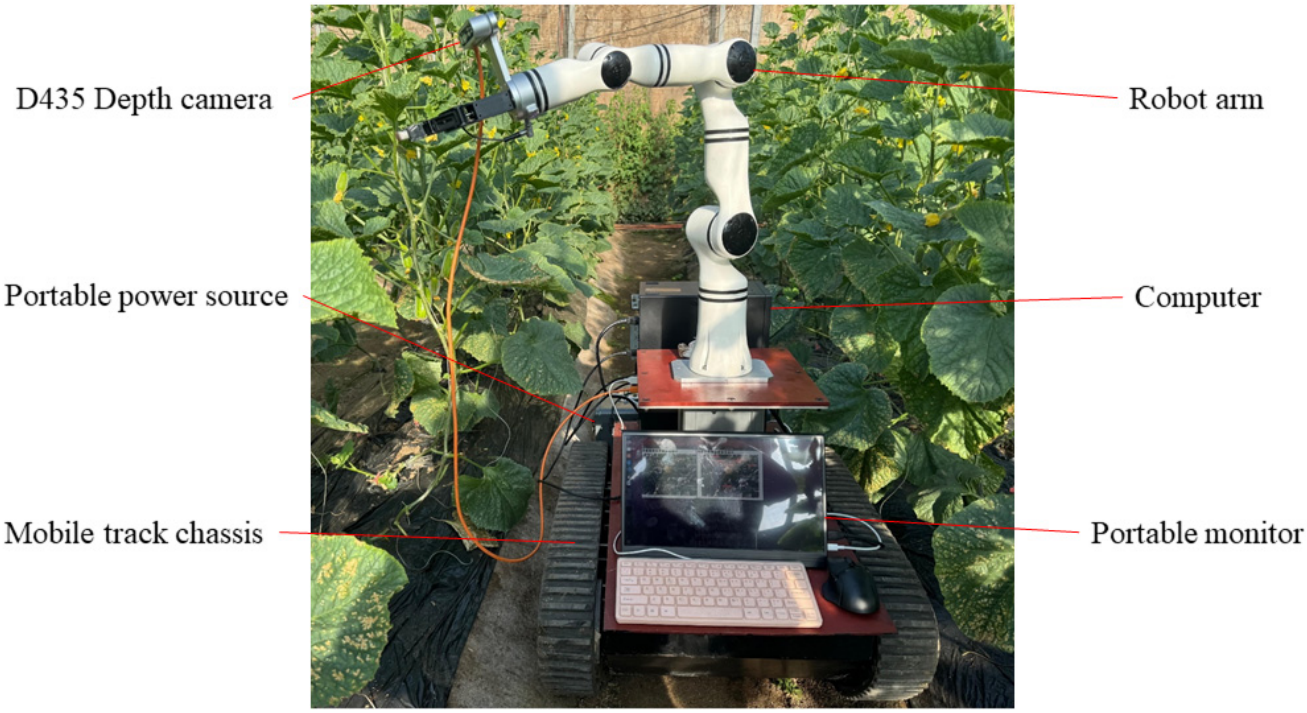


**Figure 5.** Photographs of the automatic pollination robot experiment.

### *2.4. Evaluation Metrics*

To accurately evaluate the performance of the proposed model, this study employs commonly used performance metrics in target detection algorithms, namely average precision (AP) and the F1 score. The F1 score is calculated as the harmonic mean of precision and recall, providing a balanced measure of both evaluation criteria. Both the F1 score and

AP are derived from Precision (P) and Recall (R). The computation process for each metric is presented in Equations (9)–(13).

$$\text{Precision} = \frac{\text{correctly classified actual positives}}{\text{everything classified as positive}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{9}$$

$$\text{Recall} = \frac{\text{correctly classified actual positives}}{\text{all actual positives}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

$$\text{AP} = \int_0^1 \text{Precision}(\text{Recall})\text{d}(\text{Recall}) \tag{11}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

$$\text{Parameters} = [\text{r} \times (\text{f} \times \text{f}) \times \text{o}] + \text{o} \tag{13}$$

In this context, the ground truth class of the samples is compared with the model's predicted class, categorizing the samples into four types: True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN). The parameters r, f, and o in the formula represent the input size, convolution kernel size, and output size, respectively.

In this experiment, the Root Mean Square Error (RMSE) is used as an evaluation metric for error analysis, and is calculated as shown in Equation (14).

$$\text{RMSE} = \sqrt{\frac{1}{\text{n}}\sum_{\text{i}=1}^{\text{n}}(\text{Y}_\text{i} - \text{y}_\text{i})^2} \tag{14}$$

In the equation, n represents the number of observation values, $\text{Y}_\text{i}$ is the true value of the i-th observation, and $\text{y}_\text{i}$ is the predicted value of the i-th observation.

### *2.5. Depth Camera-Based Target Localization Methods*

#### 2.5.1. Transformation Between Pixel Coordinate System and Image Coordinate System

Depth cameras represent a significant advancement over traditional cameras, as they are capable of measuring the distance to a target. The depth camera employed in this study is the Intel RealSense D435 (referred to as D435, manufactured by Intel Corporation, headquartered in Santa Clara, CA, USA). After the object detection model identifies the pixel coordinates o(u,v) of the bounding box, the 3D coordinates can be ultimately obtained by incorporating the depth information provided by the depth camera. This process relies on a clear understanding of the relationship between the pixel coordinate system and the image coordinate system.

As illustrated in Figure 6, $p$ denotes the 2D coordinates of the target center identified by the model in the pixel coordinate system $O_{uv}$, while $O_{xy}$ represents the image coordinate system. The transformation between pixel coordinate system and image coordinate system is defined as shown in Equations (15)–(17), where Equation (17) represents the matrix form of the transformations described in Equations (15) and (16).

$$\text{u} = \frac{\text{x}}{\text{dx}} + \text{u}_0 \tag{15}$$

$$\text{v} = \frac{\text{y}}{\text{dy}} + \text{v}_0 \tag{16}$$

$$\begin{bmatrix} \text{u} \\ \text{v} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{\text{dx}} & 0 & \text{u}_0 \\ 0 & \frac{1}{\text{dy}} & \text{v}_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \text{x} \\ \text{y} \\ 1 \end{bmatrix} \tag{17}$$

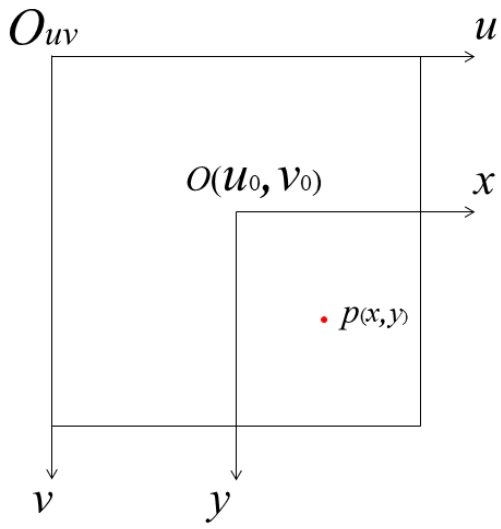


**Figure 6.** Pixel coordinate system and image coordinate system.

2.5.2. Transformation Between Image Coordinate System and Camera Coordinate System

As shown in Figure 7, point *p* still represents the coordinates (u,v) detected by the model in the pixel coordinate system, which correspond to (x,y) in the image coordinate system. Based on the principles of perspective mapping, the projection in the camera coordinate system of point *p* in the image coordinate system is denoted as $P(X_t,Y_t,Z_t)$. The coordinate transformation between these systems is expressed in Equations (18) and (19), where Equation (19) represents the homogeneous matrix form of the transformation described in Equation (18).

$$\frac{BC}{oA} = \frac{BO_t}{oC_t} = \frac{PC}{pA} = \frac{X_t}{x} = \frac{Y_t}{y} = \frac{Z_t}{z} \tag{18}$$

$$Z_t \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} z & 0 & 0 \\ 0 & z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_t \\ Y_t \\ Z_t \end{bmatrix} \tag{19}$$

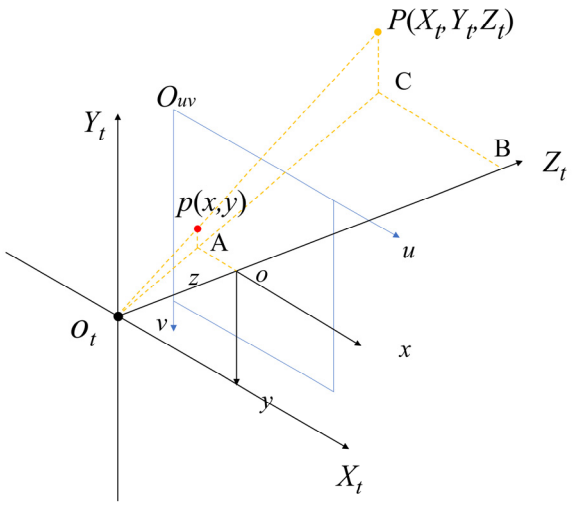


**Figure 7.** The relationship between the image coordinate system and the camera coordinate system.

To project 3D spatial points onto 2D image points and correct image distortions, it is necessary to perform camera intrinsic calibration to obtain the internal reference matrix.

This study calibrates the D435 camera using Zhang Zhengyou's [30] method to complete the internal parameter calibration. A calibration plate is fabricated, fixing the camera position, and capturing 28 images by rotating the calibration plate at various angels and attitudes. The calibration process is depicted in Figure 8.

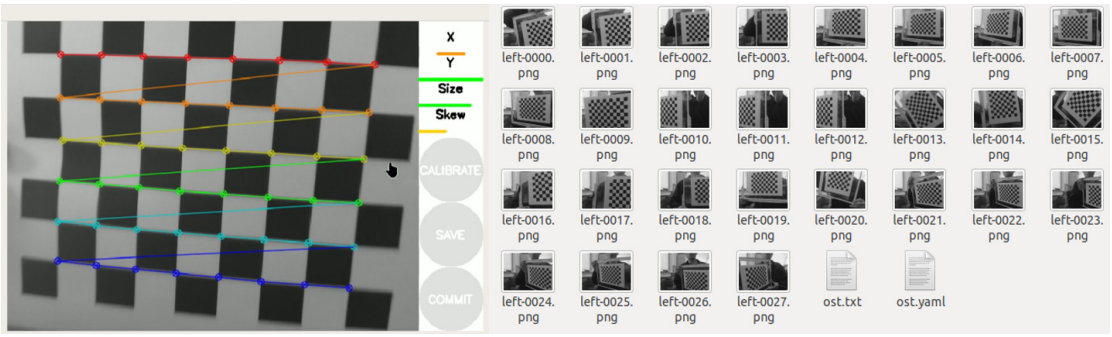


**Figure 8.** The process of camera intrinsic calibration.

### 2.5.3. Transformation Between Camera Coordinate System and Robot Coordinate System

The transformation between the camera coordinate system and the robot coordinate system relies on the extrinsic calibration of the camera, commonly referred to as hand-eye calibration. In this study, an "eye-on-hand" hand-eye calibration approach was adopted, enabling the coordinate transformation between the camera frame and the robot end-effector frame. The calibration procedure is illustrated in Figure 9. Specifically, the camera is mounted on the end-effector of the robotic arm and aligned with an Aruco Marker. During the calibration process, images are captured by the camera while the robot is moved to various poses. Simultaneously, the corresponding joint position data of the robotic arm is recorded. Based on this dataset, the hand-eye calibration algorithm computes the transformation matrix that describes the spatial relationship between the camera and the robotic arm.

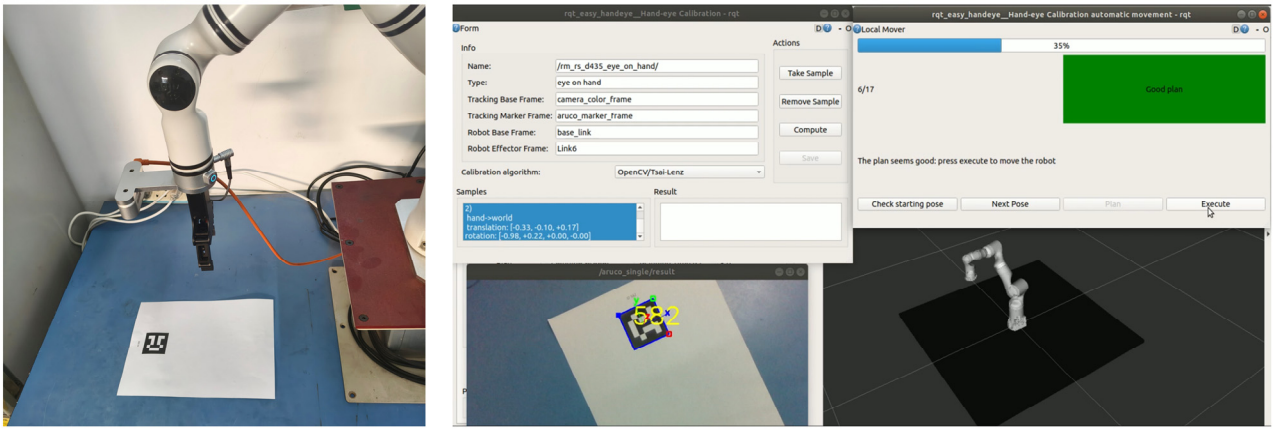


**Figure 9.** The process of camera hand-eye calibration.

After the aforementioned hand-eye calibration, the transformation between the camera coordinate system and the end-effector coordinate system of the robotic arm is established. However, this does not fully align the camera coordinate system with the robot's global coordinate system. The robot coordinate system is defined with respect to the base frame, and therefore, an additional transformation from the end-effector frame to the base frame is required. This transformation falls within the scope of robotic kinematics and relies on the D–H parameters of the robot. The robotic arm in this study employs the modified D–H

(MD-H) parameters, which represent an enhanced version of the standard D–H convention. This approach improves applicability by refining the definition of joint parameters. The modified D–H parameter table for the RM65-B robotic arm is presented in Table 1.

**Table 1.** MD-H parameters of the RM65-B robot.

| Joint_id (i) | $a_{i-1}$ (mm) | $\alpha_{i-1}$ (°) | $d_i$ (mm) | $\theta_i$ (°) |
|---|---|---|---|---|
| 1 | 0 | 00 | 240.5 | 00 |
| 2 | 0 | 90 | 00 | 99 |
| 3 | 256 | 0 | 0 | 90 |
| 4 | 0 | 90 | 210 | 0 |
| 5 | 0 | −90 | 0 | 0 |
| 6 | 0 | 90 | 144 | 0 |

In the table, $a_{i-1}$ denotes the link length, $\alpha_{i-1}$ denotes the link twist angle, $d_i$ represents the link offset, and $\theta_i$ indicates the joint rotation angle.

### *2.6. ROS-Based Robotic Arm Control*

ROS is an open-source middleware framework designed for robotic development. In this study, the enhanced YOLO-MDL detection model is integrated with a localization algorithm to obtain the 3D coordinates of the target. Both the model and the algorithm are deployed on the robot control PC running Ubuntu, with ROS facilitating the control of the robotic arm. This setup enables path planning, ensuring that the brush reaches the 3D coordinates of the detected flowers to perform robot-assisted pollination. As outlined in Figure 10, ROS is employed to execute the robot's automatic pollination task, utilizing the target detection and localization algorithm.

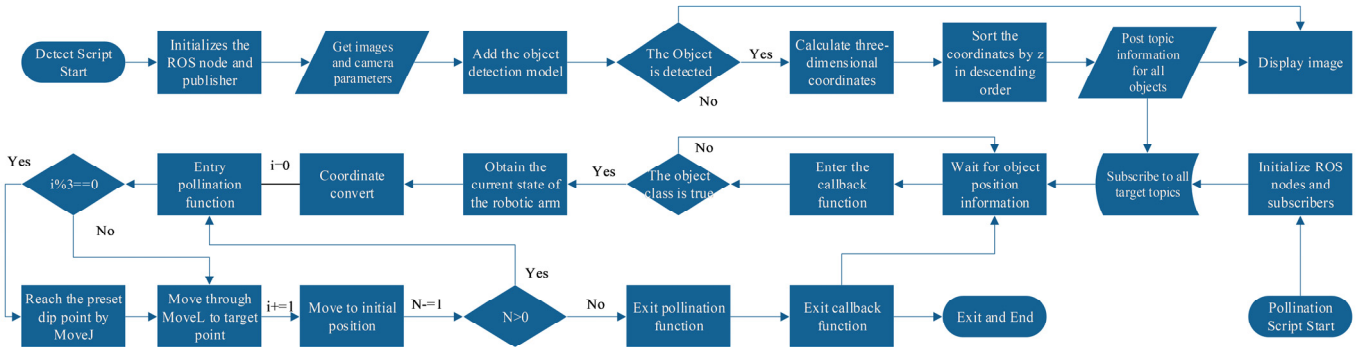


**Figure 10.** Flowchart of ROS-controlled robotic arm to complete the pollination process.

## 3. Experiments and Results

### *3.1. Effect of Different Attention Mechanisms on Model Performance*

The CBAM (Convolutional Block Attention Module) attention mechanism [31], SE (Squeeze-and-Excitation) attention mechanism [32], and ECA (Efficient Channel Attention) [33] attention mechanism were incorporated into the original network model and compared with the YOLOv7-CA model, which includes the CA attention mechanism. Like the CA attention mechanism, the other attention mechanisms were also evaluated at four different positions to determine the optimal position. The experimental results show that the other mechanisms achieved the best performance when placed at position A2. The contrast experimental results are presented in Table 2.

**Table 2.** Comparison of the effects of different attention mechanisms.

| Model | P (%) | R (%) | AP (%) | F1 (%) |
|---|---|---|---|---|
| YOLOv7 | 85.0 | 81.7 | 87.9 | 83.3 |
| YOLOv7-CBAM | 91.2 | 77.0 | 88.5 | 83.5 |
| YOLOv7-SE | 87.9 | 81.7 | 89.0 | 84.7 |
| YOLOv7-ECA | 85.1 | 80.2 | 88.3 | 82.6 |
| YOLOv7-CA | 89 | 81.7 | 89.1 | 85.2 |

P Refers to Precision; R Refers to Recall; AP Refers to Average Precision; F1 Refers to F1 score.

A comparison of the results demonstrates that the performance of the model improves after incorporating any of the aforementioned attention mechanisms. Notably, the highest average precision (AP) was achieved with the CA attention mechanism, which resulted in a 2.4% increase over the original model. Additionally, the F1 score reached 85.2%, reflecting a 1.9% improvement compared to the original model. Therefore, the integration of the CA attention mechanism yields the best performance, enhancing the model's overall effectiveness.

### *3.2. Effect of Applying Different Positions of CA Module and CoT Module on Model Performance*

To determine the optimal placement of the CA and CoT modules within the original model, comparative experiments were conducted by integrating the modules at different positions. After integrating the CA modules into positions A1, A2, A3, and A4 of the network, the model's detection performance improved over the original model, with the exception of position A3. Specifically, the average precision (AP) increased by 1.2%, 0.2%, and 0.4% at positions A1, A2, and A4, respectively. The model's detection performance was optimal when the CA module was added to position A1, as detailed in Table 3. Following the introduction of the CoT module, a significant decline in detection performance was observed at position T4 compared to the original model. However, at positions T1, T2, and T3, the model's performance improved, with the AP increasing by 0.6%, 1.3%, and 2.1%, respectively. The best detection performance was achieved when the CoT module was added at position T3, as presented in Table 4.

**Table 3.** Comparison of the effect of adding CA module at different positions.

| Model | P (%) | R (%) | AP (%) | F1 (%) |
|---|---|---|---|---|
| YOLOv7 | 85.0 | 81.7 | 87.9 | 83.3 |
| YOLOv7-CA-A1 | 89.0 | 81.7 | 89.1 | 85.2 |
| YOLOv7-CA-A2 | 90.1 | 74.3 | 88.0 | 81.4 |
| YOLOv7-CA-A3 | 85.4 | 75.5 | 85.5 | 80.1 |
| YOLOv7-CA-A4 | 86.1 | 81.7 | 88.4 | 83.8 |

**Table 4.** Comparison of the effect of adding CoT module at different positions.

| Model | P (%) | R (%) | AP (%) | F1 (%) |
|---|---|---|---|---|
| YOLOv7 | 85.0 | 81.7 | 87.9 | 83.3 |
| YOLOv7-CoT-T1 | 90.7 | 75.9 | 88.5 | 82.6 |
| YOLOv7-CoT-T2 | 86.0 | 81.3 | 89.2 | 83.6 |
| YOLOv7-CoT-T3 | 88.7 | 82.9 | 90.0 | 85.7 |
| YOLOv7-CoT-T4 | 83.3 | 68.5 | 79.7 | 75.2 |

### *3.3. Ablation Test*

Ablation tests are used to explore the effects of specific substructures of the network or training strategies on the model. In this study, we compare the performance of different

improved models through an ablation test based on the YOLOv7 model. The ablation test results are presented in Table 5. Where YOLOv7-CA-A1 represents adding the CA at position A1 of the model, YOLOv7-CoT-T3 represents adding the CoT at position T3 of the model, and YOLO-MDL is the final improved model.

**Table 5.** Ablation test results.

| Model | P (%) | R (%) | AP (%) | F1 (%) |
|---|---|---|---|---|
| YOLOv7 | 85.0 | 81.7 | 87.9 | 83.3 |
| YOLOv7-CA-A1 | 89.0 | 81.7 | 89.1 | 85.2 |
| YOLOv7-CoT-T3 | 88.7 | 82.9 | 90.0 | 85.7 |
| YOLO-MDL | 92.7 | 78.6 | 91.2 | 85.1 |

The experimental results demonstrate that each improved model outperforms the original one. Notably, the performance enhancement achieved by incorporating the CoT module alone is greater than that obtained by adding the CA module individually. The YOLO-MDL model, which integrates both modules and benefits from their combined effect, achieves the best overall performance.

In order to further verify the effectiveness of the added module, the AP and Loss curves of the validation set of the YOLO-MDL, original model, and models after adding a single module are shown in Figure 11. The four models basically converge and stabilize after 75 epochs; it can be seen in the AP curve that the AP of YOLO-MDL is higher than that of the other models. As shown in the Loss curve, the loss of the original model without any added modules is the highest, and YOLO-MDL has the lowest loss compared with the other models, indicating that the model fitting performance is good and stable with the best results. In summary, the improvements in this study are effective for the model. Furthermore, although the AP curve graph indicates that the models basically stabilize after 75 epochs, the loss curve reveals that the loss value does not reach a stable state at this point. Instead, stabilization of the loss is observed only after around 175 epochs. Based on this observation, a total of 200 training epochs were adopted in this study to ensure full convergence of the model.

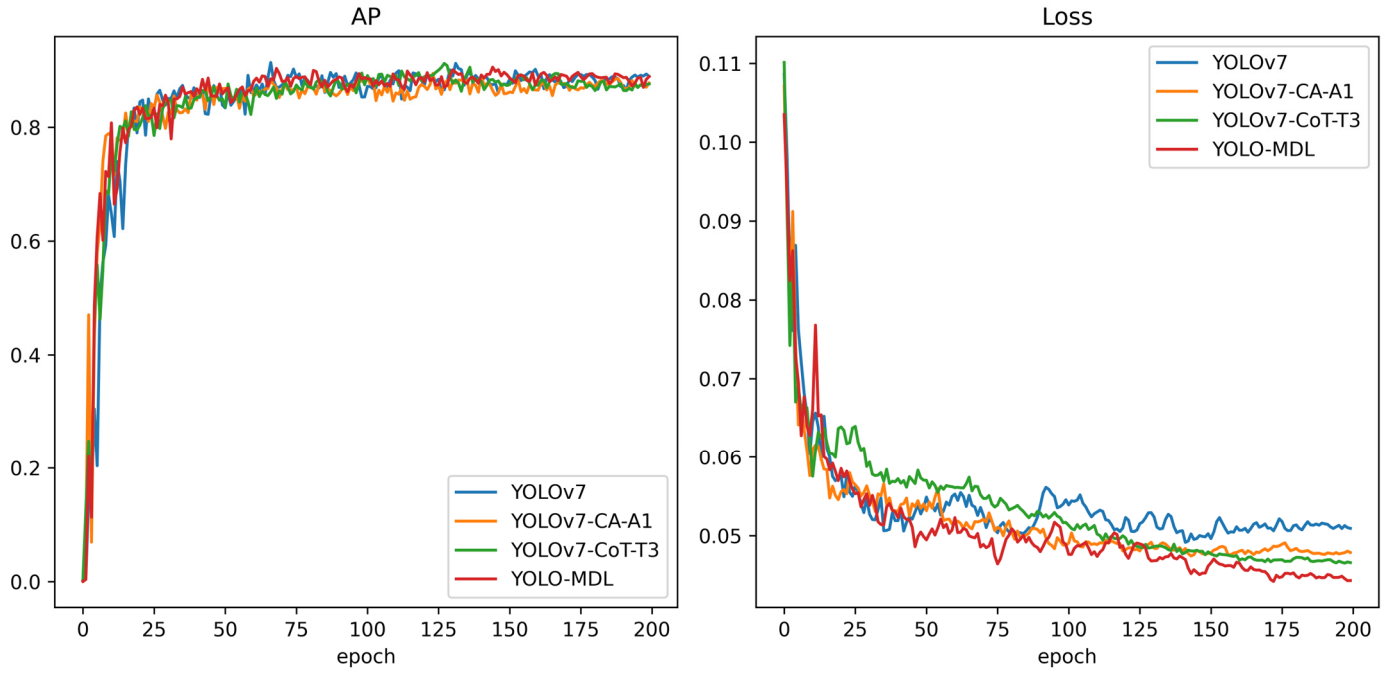


**Figure 11.** AP and Loss curves of original model and improved models.

### *3.4. Comparison of Detection Effectiveness of Different Models*

In this study, after deriving the highest-performing YOLO-MDL model, a comparison was conducted with other YOLO models, including the latest YOLOv12 model. The results demonstrate that the improved YOLO-MDL model outperforms the other models, with significantly higher values for both average precision (AP) and F1 score. The experimental results are presented in Table 6.

**Table 6.** Comparison of the effect of different models.

| Model | P (%) | R (%) | AP (%) | F1 (%) | Model Size (MB) |
|---|---|---|---|---|---|
| YOLOv7 | 85.0 | 81.7 | 87.9 | 83.3 | 73.0 |
| YOLOv7-tiny | 84.7 | 75.9 | 84.6 | 80.1 | 12.0 |
| YOLOv7x | 84.3 | 75.1 | 83.6 | 79.4 | 138.7 |
| YOLOv5m | 82.8 | 71.1 | 79.4 | 76.4 | 41.2 |
| YOLOv8m | 79.1 | 71.3 | 78.9 | 75.2 | 50.8 |
| YOLOv9m | 80.9 | 82.3 | 87.1 | 81.6 | 64.7 |
| YOLOv10m | 76.0 | 73.5 | 77.9 | 74.7 | 32.7 |
| YOLOv11m | 82.1 | 68.8 | 78.0 | 74.9 | 39.5 |
| YOLOv12m | 80.8 | 66.6 | 75.9 | 73.0 | 39.8 |
| YOLO-MDL | 92.7 | 78.6 | 91.2 | 85.1 | 72.4 |

To provide a more intuitive comparison of the model performance, each model parameter is visualized in a radar chart and a 3D histogram. As shown in Figure 12a, the radar chart clearly illustrates that the improved YOLO-MDL model outperforms the other models, particularly in terms of AP and F1 scores, when comparing models of similar sizes. As shown in Figure 12b, the advantage of YOLO-MDL in detection accuracy can be more clearly seen by the 3D histogram after normalization, which has the highest Precision (P), AP, and F1 scores, except for the Recall (R).

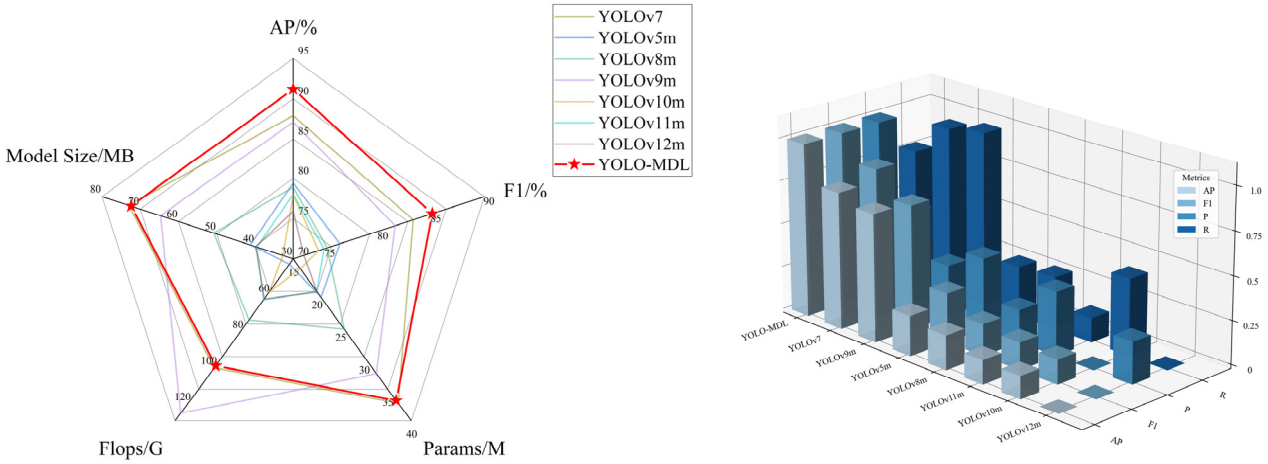


**Figure 12.** (**a**) Radar diagram for model comparison; (**b**) 3D histogram diagram for model comparison.

To confirm the benefits of the YOLO-MDL model in detecting muskmelon flowers, the detection results of several photos of muskmelon flowers in the test set were compared with those of YOLOv7. Figure 13 displays the results.


**Figure 13.** Detection effect between YOLO-MDL and YOLOv7.

In the figure, a blue cross represents false detection, while a blue circle indicates false detection. Compared to the original YOLOv7, YOLO-MDL not only achieves higher detection confidence but also significantly reduces both false detection and false detection. This qualitative comparison clearly demonstrates the improved detection performance of the enhanced YOLO-MDL model.

### *3.5. Results of Camera Calibration*

The specific intrinsic parameter matrix after calibration is as follows:

$$\begin{bmatrix} 621.55 & 0 & 308.19 \\ 0 & 612.07 & 239.92 \\ 0 & 0 & 1 \end{bmatrix} \tag{20}$$

The hand-eye calibration results of the camera, which include the translational and rotational components, are as follows: the translation vector t is [−0.109617, 0.049763, −0.026437] and the rotation vector q, represented as a quaternion, is [0.007165, 0.021498, −0.671826, 0.740362]. The quaternion q can be converted into a 3 × 3 rotation matrix R, as expressed in Equation (21).

$$\mathrm{R} = \begin{bmatrix} 0.096375 & 0.995297 & 0.022205 \\ -0.994481 & 0.097197 & -0.039495 \\ -0.041460 & -0.018276 & 0.998973 \end{bmatrix} \tag{21}$$

The resulting homogeneous transformation matrix T used to describe the spatial relationship between the camera and the robotic end-effector is expressed in Equation (22):

$$T = \begin{bmatrix} R & t^T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.096375 & 0.995097 & 0.022205 & -0.103617 \\ -0.994481 & 0.097197 & -0.039495 & 0.049763 \\ -0.041460 & -0.018276 & 0.998973 & -0.026437 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (22)$$

### *3.6. Positioning Experiment Results and Analysis*

After calibrating the camera, the final improved model, integrated with the depth camera, was used to perform positioning experiments to assess the accuracy of the model's detection and positioning algorithms. The depth camera is mounted at the end of the robotic arm, which is controlled via ROS to adjust the distance between the camera and the detection target. The procedure is as follows:

Step 1: Once the robot moves to the designated operation position, the camera is adjusted to an optimal angle and fixed in place using ROS.

Step 2: The detection and positioning process is then initiated, and the 3D coordinates of the target's center are recorded and stored along with the original image. The detection results are shown in Figure 14.

(a) (b)

**Figure 14.** Detection and localization results. (**a**) YOLO-MDL detection results; (**b**) 3D coordinates calculated by combining depth camera.

Step 3: A tape measure is used to measure the distance from the camera to the target surface, and the measurement is recorded, as shown in Figure 15.

Step 4: This process is repeated 20 times, with all data being recorded for further analysis.

The ground truth values for the pixel coordinates of the center point of the target detection frame in the image, obtained in step 2, are recorded using LabelImg. Additionally, the ground truth depth distance is obtained from step 3. These ground truth coordinates are then compared with the predicted coordinates from the test, and the results are presented in Table 7.

As shown in the table, the maximum errors in the true 3D coordinate values are 0.6 mm, 2.9 mm, and 7 mm along the X, Y, and Z axes, respectively, when compared to the predicted values identified and localized by the model. The relatively large errors in the depth values can be attributed to measurement inaccuracies as well as the instability of the camera. In this experiment, the RMSE values are as follows: 3.87 mm for the Z-axis,

0.36 mm for the X-axis, and 1.26 mm for the Y-axis. The use of the ROS-controlled robotic arm for automatic pollination experiments also achieves precise positioning results, as shown in Figure 15.

(a) (b)

**Figure 15.** Experimental process. (**a**) Measuring the actual distance from the camera to the target; (**b**) ROS controlling the robotic arm to complete pollination.

**Table 7.** Results of Muskmelon flower 3D coordinate positioning.

| | Actual Value (mm) | | | Predicted Value (mm) | | | Inaccuracies (mm) | | |
|---|---|---|---|---|---|---|---|---|---|
| **No.** | $x$ | $y$ | $z$ | $x'$ | $y'$ | $z'$ | $\|\Delta x\|$ | $\|\Delta y\|$ | $\|\Delta z\|$ |
| 1 | −5.1 | 14.2 | 357 | −5.2 | 14.4 | 358 | 0.1 | 0.2 | 1 |
| 2 | 66.3 | 37.8 | 432 | 66.1 | 37.2 | 434 | 0.2 | 0.6 | 2 |
| 3 | 56.9 | 10.9 | 480 | 57.1 | 10.4 | 486 | 0.2 | 0.5 | 6 |
| 4 | −14.1 | 7.2 | 277 | −14.2 | 6.8 | 282 | 0.1 | 0.4 | 5 |
| 5 | 46.3 | 4.2 | 221 | 46.1 | 4.3 | 226 | 0.2 | 0.1 | 5 |
| 6 | −1.4 | 24.8 | 551 | −1.2 | 24.3 | 556 | 0.2 | 0.5 | 5 |
| 7 | −15.2 | 44.9 | 618 | −15.3 | 43.4 | 625 | 0.1 | 1.5 | 7 |
| 8 | −1.4 | 48.1 | 248 | −1.1 | 47.4 | 254 | 0.3 | 0.7 | 6 |
| 9 | 12.4 | 23.2 | 167 | 12.7 | 21.3 | 173 | 0.3 | 1.9 | 6 |
| 10 | 17.1 | 51.3 | 308 | 17.7 | 52.1 | 312 | 0.6 | 0.8 | 4 |
| 11 | 17.2 | 50.8 | 300 | 16.8 | 51.4 | 302 | 0.4 | 0.6 | 2 |
| 12 | −30.9 | 73.3 | 446 | −30.4 | 72.4 | 450 | 0.5 | 0.9 | 4 |
| 13 | −17.1 | 132.4 | 500 | −17.4 | 131.4 | 501 | 0.3 | 1.0 | 1 |
| 14 | 9.2 | 57.2 | 592 | 9.3 | 56.3 | 594 | 0.1 | 0.9 | 2 |
| 15 | −74.6 | 123.1 | 673 | −75.1 | 121.4 | 675 | 0.5 | 1.7 | 2 |
| 16 | −87.2 | 83.3 | 716 | −86.9 | 82.4 | 717 | 0.3 | 0.9 | 1 |
| 17 | −29.5 | −18.2 | 613 | −28.9 | −19.6 | 615 | 0.6 | 1.4 | 2 |
| 18 | −31.8 | −15.5 | 721 | −31.2 | −17.8 | 723 | 0.6 | 2.3 | 2 |
| 19 | −63.1 | 41.3 | 844 | −63.2 | 38.4 | 846 | 0.1 | 2.9 | 2 |
| 20 | −116.8 | 29.3 | 952 | −117.3 | 28.2 | 954 | 0.5 | 1.1 | 2 |

The time consumption of each stage in the entire pollination process is illustrated in Figure 16, including model detection and localization, as well as robotic arm movement.

By presenting the time breakdown of each step, the feasibility of deploying the proposed method on edge devices is better demonstrated.

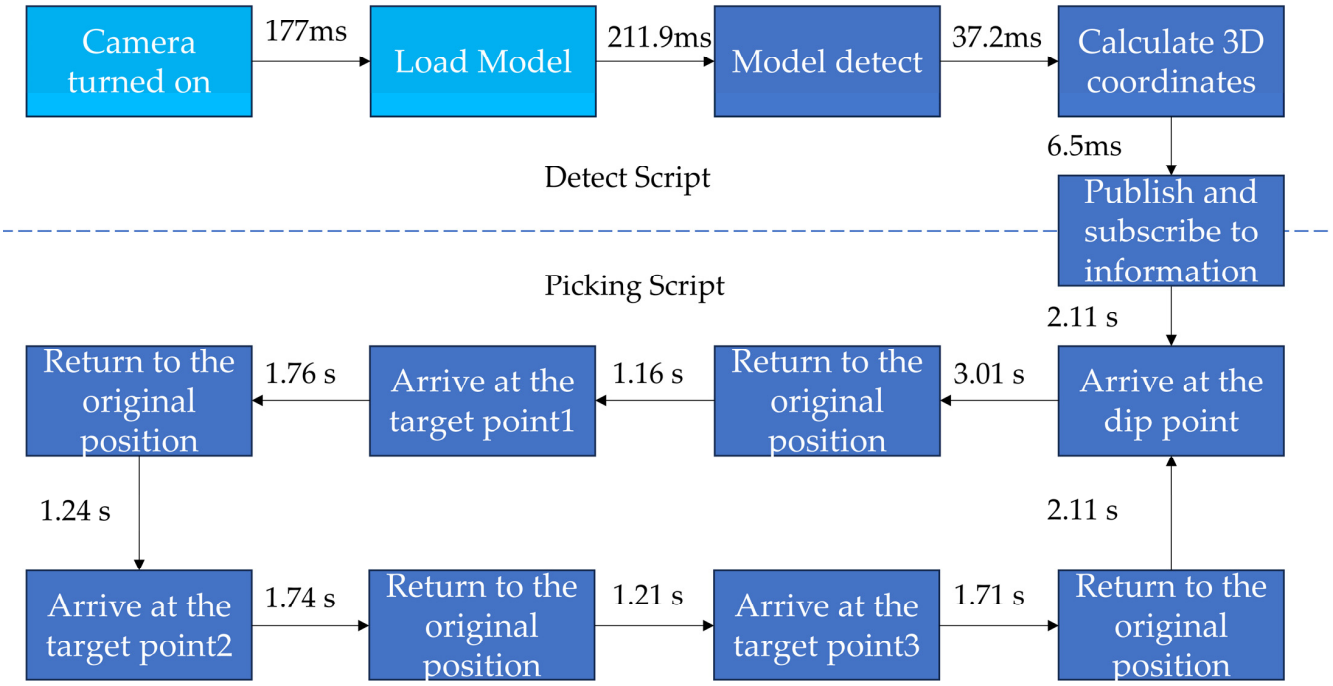


**Figure 16.** Time consumption details of the complete pollination process.

It should be noted, however, that the camera configuration and model loading steps are only performed once at the beginning of the first pollination task. These two components are not included in subsequent consecutive pollination operations. Moreover, during the robotic arm's pollination process, the total execution time is not simply the sum of the individual step durations shown in the figure. Additional node sleep times, introduced intentionally between actions, allow for error tolerance and accommodate unexpected situations. Nevertheless, even with these safety margins, the system is still able to complete a single pollination task for one flower within 8 s and continuous pollination tasks for three flowers within 18 s, which aligns well with the expectations of the current research phase.

## 4. Discussion

The YOLO-MDL model proposed in this study performed exceptionally well in detecting muskmelon flowers and when combined with a three-dimensional positioning method, providing reliable technical support for the automatic pollination robot of muskmelons. The experimental results and multi-dimensional comparisons show that YOLO-MDL offers high detection accuracy. In addition, this model can effectively combine with the three-dimensional positioning algorithm and be deployed in the robotic arm to achieve automatic pollination of muskmelons.

Firstly, the study compared the performance of the model after adding different attention mechanisms, showing that the model with the CA attention mechanism achieved the highest AP and F1 scores. Then, the study investigated the effect of positioning the CA and CoT modules at various locations in the model, determining the optimal locations for each module individually, as well as the optimal configuration for adding both modules simultaneously. The final improved YOLO-MDL model, after incorporating both modules, achieved an AP of 91.2% and an F1 score of 85.1%, which represents a 3.3% and 1.8% increase in AP and F1, respectively. Additionally, YOLO-MDL was also compared with

other common YOLO models, with the results indicating that YOLO-MDL outperforms other models in detection performance. In comparison with the apple flower detection approach proposed by Chen et al. [20], which is based on an improved version of YOLOv5, our enhanced model achieves a significantly higher AP of 91.2%, compared to their reported mAP 77.5%. Additionally, our model demonstrates a greater improvement in AP (3.3%), outperforming their reported gain of 1.7%, which further highlights the effectiveness of our proposed method.

For the localization experiments, the D435 camera was calibrated using the internal reference matrix. The improved YOLO-MDL model, combined with a localization algorithm, was used for 3D localization experiments. The predicted 3D coordinates were compared with the true values, and RMSE was used for error analysis. The results show minimal errors in all three axes: the RMSE for the Z-axis was 3.87 mm, for the X-axis 0.36 mm, and for the Y-axis 1.26 mm, meeting the high-precision localization requirements for pollinating robots in natural scenarios. The system is able to complete a single pollination task for one flower within 8 s. In the watermelon pollination method proposed by Ahmad et al. [27], the average depth error related to pollination target localization is 1.028 cm, with an average pollination time of 8 s per flower. When compared to the method presented in this study, both the depth error and the time required for pollination are higher, indicating that our approach offers improved accuracy and efficiency in robotic pollination tasks.

However, some noteworthy limitations were identified in this study. Although the detection accuracy of YOLO-MDL is very high, the deployment of the model on hardware devices requires a high degree of lightweighting, which cannot be met by existing models. Secondly, a relatively small dataset may limit the model's generalization capability, especially under greenhouse conditions, where flowers may suffer from occlusion, blurring, large viewpoint variations, or background interference. Thirdly, because in the actual scene the muskmelon flower in the camera captured image will appear in a variety of shapes and poses, such as flowering and non-flowering, with stamens facing the camera or back to the camera, and so on, this situation will affect not only the detection accuracy but also the pollination operation of the robot. In addition to this, compared to the errors in the other two dimensions, the larger errors in the depth values in the localization experimental results can be attributed to the perceived inaccuracy of the measurements as well as the inherent errors in the calibration of the camera parameters. Furthermore, it is important to note that both measurement accuracy and repeatability can be influenced by external disturbances, such as mechanical vibration. Therefore, our future research will focus on the following aspects:

(1) Lightweighting research: In order for the model to meet the requirement of easy deployment on hardware devices to the extent that the speed of detection can be increased to improve pollination efficiency, subsequent research will prioritize lightweight module design, optimization of the model structure, and pruning to achieve lightweight.
(2) Dataset expansion: To avoid the possibility of reduced model generalization capability due to small dataset, we are planning to collect more images across different growth stages and environmental conditions or use a variety of data enhancement methods to enhance the dataset's comprehensiveness. Additionally, in our future research, we intend to carry out systematic experiments to analyze how varying dataset sizes affect model performance.
(3) Diversification of data processing and model design: Since in real-life scenarios, flowers can take on different shapes and poses, and the current treatment of the dataset only makes it possible to label open flowers, which does not take into account all scenarios, in the future, the dataset can be labeled according to multi-class of

"buds", "fully bloomed flowers", and "wilted flowers", including open and unopened flowers. This method can better reflect real-world variability and improve the model's ability to distinguish pollinatable flowers from non-targets in complex environments. In addition to this, pose estimation of flowers can also be studied to optimize the target selection of robotic pollination through the combination of attitude estimation and target detection.

(4) Improvement of measurement means and experimental design: Since the inaccuracy of human measurement will largely affect the size of the error in the depth value, the use of high-precision laser rangefinders to measure the size of the actual depth value will be considered in subsequent experiments. In response to the influence of external factors on accuracy measurements, this consideration can be incorporated into future research to optimize the experimental design by validating the system's accuracy through real-world error measurements under dynamic conditions.

## 5. Conclusions

In order to improve pollination efficiency and alleviate labor demands, this study designs an automatic pollination robot and investigates automatic pollination technology, with a focus on the key aspect of accurate recognition models. The current YOLO detection method based on deep learning demonstrates high efficiency and excellent reliability in complex flower environments; however, it has not yet yielded effective results for muskmelon flower detection, presenting challenges in achieving low-cost, high-precision automatic pollination. To address these issues, this study constructed a muskmelon flower image dataset and proposed a method for detecting and locating muskmelon flowers using the YOLO-MDL model. After enhancing the model and integrating it with a depth camera for 3D positioning and error measurement, the ROS-controlled robotic arm completed the pollination task. The experimental results show that the AP and F1 of YOLO-MDL are 91.2% and 85.1% respectively, which are significantly higher than the original model and other target detection models. In the localization experiments, the predicted 3D coordinates are compared with the true 3D coordinates and the RMSE is used as the evaluation index for error analysis. The results show that the coordinate errors in all three directions are small, which meets the requirement of high-precision localization of automatic pollination robots in natural scenes.

In conclusion, the YOLO-MDL model proposed in this study, designed for real-time detection and localization of muskmelon flowers, satisfies the high-precision requirements for both detection and localization after experimental evaluation. This provides strong technical support for future automatic pollination robots. In future work, the model can be further optimized to reduce its size while maintaining detection accuracy, facilitating deployment on edge devices. Additionally, reinforcement learning can be integrated to enable the robot to autonomously adjust parameters based on environmental changes, further enhancing pollination efficiency.

**Author Contributions:** Conceptualization, D.X. and H.Z.; methodology, H.Z., Y.Z. (Yongzhuo Zhang), L.J., Y.Z. (Yabo Zheng) and S.X.; validation, D.X., H.Z. and W.Y.; formal analysis, H.Z.; resources, H.Z. and R.R.; data curation, S.X., R.R., Y.Z. (Yongzhuo Zhang), L.J., Y.Z. (Yabo Zheng) and E.Z.; writing—original draft preparation, S.X.; writing—review and editing, D.X., W.Y. and H.Z.; funding acquisition, D.X. and H.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Basic Research Project of Shanxi Province, grant number 202103021223145; the 2025 Shanxi Agricultural University Discipline Construction Special Project and Lvliang City introduces key research and development projects for high-level scientific and technological talents, under grant number 2024NY03.

**Data Availability Statement:** The raw data supporting the conclusions of this article will be made available by the authors on request.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References


1. Chi, X.; Zhao, Y.; Wang, W.; Chen, Y. Analysis and evaluation of genetic diversity of melon germplasm resources. *China Cucurbits Veg.* **2024**, *37*, 0468.
2. Németh, D.; Balázs, G.; Daood, H.G.; Kovács, Z.; Bodor, Z.; Zinia Zaukuu, J.-L.; Szentpéteri, V.; Kókai, Z.; Kappel, N. Standard Analytical Methods, Sensory Evaluation, NIRS and Electronic Tongue for Sensing Taste Attributes of Different Melon Varieties. *Sensors* **2019**, *19*, 5010. [CrossRef]
3. Cui, H.; Ding, Z.; Zhu, Q.; Wu, Y.; Qiu, B.; Gao, P. Population Structure, Genetic Diversity and Fruit-Related Traits of Wild and Cultivated Melons Based on Chloroplast Genome. *Genet. Resour. Crop Evol.* **2021**, *68*, 1011–1021. [CrossRef]
4. Zhao, S.; Wu, J.; He, Y.; Gao, Z.; Zhong, Y.; Han, W.; Zhao, D.; Gao, J.; Wang, S. Effects of different pollination methods on the quality of facility netted melons. *China Cucurbits Veg.* **2024**, *34*, 0315.
5. Fu, Q.; Zhang, X.; Zhu, H.; Wang, H. Effects of different pollination methods on fruit quality of facility thick-skinned melon. *China Veg.* **2014**, *11*, 31–36.
6. Rahim, U.F.; Utsumi, T.; Mineno, H. Deep Learning-Based Accurate Grapevine Inflorescence and Flower Quantification in Unstructured Vineyard Images Acquired Using a Mobile Sensing Platform. *Comput. Electron. Agric.* **2022**, *198*, 107088. [CrossRef]
7. Wang, X.; Tang, J.; Whitty, M. DeepPhenology: Estimation of Apple Flower Phenology Distributions Based on Deep Learning. *Comput. Electron. Agric.* **2021**, *185*, 106123. [CrossRef]
8. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
9. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision 2015, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
10. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: New York, NY, USA, 2015; Volume 28.
11. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In *Proceedings of the 16th IEEE Transactions on Pattern Analysis & Machine Intelligence*; IEEE Computer Society: Washington, DC, USA, 2017; pp. 2961–2969.
12. Lin, P.; Lee, W.S.; Chen, Y.M.; Peres, N.; Fraisse, C. A Deep-Level Region-Based Visual Representation Architecture for Detecting Strawberry Flowers in an Outdoor Field. *Precis. Agric.* **2020**, *21*, 387–402. [CrossRef]
13. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Proceedings of the Computer Vision—ECCV 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.
14. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
15. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
16. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
17. Zhou, Y.; Tang, Y.; Zou, X.; Wu, M.; Tang, W.; Meng, F.; Zhang, Y.; Kang, H. Adaptive Active Positioning of Camellia Oleifera Fruit Picking Points: Classical Image Processing and YOLOv7 Fusion Algorithm. *Appl. Sci.* **2022**, *12*, 12959. [CrossRef]
18. Tian, M.; Chen, H.; Wang, Q. Detection and Recognition of Flower Image Based on SSD Network in Video Stream. *J. Phys. Conf. Ser.* **2019**, *1237*, 032045. [CrossRef]
19. Li, G.; Suo, R.; Zhao, G.; Gao, C.; Fu, L.; Shi, F.; Dhupia, J.; Li, R.; Cui, Y. Real-Time Detection of Kiwifruit Flower and Bud Simultaneously in Orchard Using YOLOv4 for Robotic Pollination. *Comput. Electron. Agric.* **2022**, *193*, 106641. [CrossRef]
20. Chen, Z.; Su, R.; Wang, Y.; Chen, G.; Wang, Z.; Yin, P.; Wang, J. Automatic Estimation of Apple Orchard Blooming Levels Using the Improved YOLOv5. *Agronomy* **2022**, *12*, 2483. [CrossRef]
21. Akdoğan, C.; Özer, T.; Oğuz, Y. PP-YOLO: Deep Learning Based Detection Model to Detect Apple and Cherry Trees in Orchard Based on Histogram and Wavelet Preprocessing Techniques. *Comput. Electron. Agric.* **2025**, *232*, 110052. [CrossRef]
22. Nguyen, D.T.; Do, P.B.L.; Nguyen, D.D.K.; Lin, W.-C. A Lightweight and Optimized Deep Learning Model for Detecting Banana Bunches and Stalks in Autonomous Harvesting Vehicles. *Smart Agric. Technol.* **2025**, *11*, 101051. [CrossRef]
23. Williams, H.; Nejati, M.; Hussein, S.; Penhall, N.; Lim, J.; Jones, M.H.; Bell, J.; Ahn, H.; Bradley, S.; Schaare, P.; et al. Autonomous Pollination of Individual Kiwifruit Flowers: Toward a Robotic Kiwifruit Pollinator. *J. Field Robot.* **2019**, *37*, 246–262. [CrossRef]

24. Gao, C.; He, L.; Fang, W.; Wu, Z.; Jiang, H.; Li, R.; Fu, L. A Novel Pollination Robot for Kiwifruit Flower Based on Preferential Flowers Selection and Precisely Target. *Comput. Electron. Agric.* **2023**, *207*, 107762. [CrossRef]
25. Wen, C.; Long, J.; Zhang, Y.; Guo, W.; Lin, S.; Liang, X. Positioning Method of Tomato Pollination Flowers Based on 3D Vision. *Trans. Chin. Soc. Agric. Mach.* **2021**, *53*, 320–328.
26. Yu, X.; Kong, D.; Xie, X.; Wang, Q.; Bai, X. Deep learning-based target recognition and detection for tomato pollination robots. *Trans. Chin. Soc. Agric. Eng.* **2022**, *38*, 129–137. [CrossRef]
27. Ahmad, K.; Park, J.-E.; Ilyas, T.; Lee, J.-H.; Lee, J.-H.; Kim, S.; Kim, H. Accurate and Robust Pollinations for Watermelons Using Intelligence Guided Visual Servoing. *Comput. Electron. Agric.* **2024**, *219*, 108753. [CrossRef]
28. Hou, Q.; Zhou, D.; Feng, J. Coordinate Attention for Efficient Mobile Network Design. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 13713–13722.
29. Li, Y.; Yao, T.; Pan, Y.; Mei, T. Contextual Transformer Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 1489–1500. [CrossRef] [PubMed]
30. Zhang, Z. A Flexible New Technique for Camera Calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]
31. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the Computer Vision –ECCV 2018: 15th European Conference, Munich, Germany, 8–14 September 2018; pp. 3–19.
32. Hu, J.; Shen, L.; Sun, G. Squeeze-and-Excitation Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
33. Wang, Q.; Wu, B.; Zhu, P.; Li, P.; Zuo, W.; Hu, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11534–11542.