agronomy

MDPI

Article

Tomato Recognition and Localization Method Based on Improved YOLOv5n-seg Model and Binocular Stereo Vision

Shuhe Zheng [1,2,†], Yang Liu [1,2,†], Wuxiong Weng [1,2,†], Xuexin Jia [1,2], Shilong Yu [1,2] and Zuoxun Wu [1,2,*]

[1] College of Mechanical and Electrical Engineering, Fujian Agriculture and Forestry University, Fuzhou 350002, China; zsh@fafu.edu.cn (S.Z.); wwx@fafu.edu.cn (W.W.)
[2] Fujian University Engineering Research Center for Modern Agricultural Equipment, Fujian Agriculture and Forestry University, Fuzhou 350002, China
* Correspondence: wuzx@fafu.edu.cn; Tel.: +86-0591-8375-6227
† These authors contributed equally to this work.

Abstract: Recognition and localization of fruits are key components to achieve automated fruit picking. However, current neural-network-based fruit recognition algorithms have disadvantages such as high complexity. Traditional stereo matching algorithms also have low accuracy. To solve these problems, this study targeting greenhouse tomatoes proposed an algorithm framework based on YOLO-TomatoSeg, a lightweight tomato instance segmentation model improved from YOLOv5n-seg, and an accurate tomato localization approach using RAFT-Stereo disparity estimation and least squares point cloud fitting. First, binocular tomato images were captured using a binocular camera system. The left image was processed by YOLO-TomatoSeg to segment tomato instances and generate masks. Concurrently, RAFT-Stereo estimated image disparity for computing the original depth point cloud. Then, the point cloud was clipped by tomato masks to isolate tomato point clouds, which were further preprocessed. Finally, a least squares sphere fitting method estimated the 3D centroid co-ordinates and radii of tomatoes by fitting the tomato point clouds to spherical models. The experimental results showed that, in the tomato instance segmentation stage, the YOLO-TomatoSeg model replaced the Backbone network of YOLOv5n-seg with the building blocks of ShuffleNetV2 and incorporated an SE attention module, which reduced model complexity while improving model segmentation accuracy. Ultimately, the YOLO-TomatoSeg model achieved an AP of 99.01% with a size of only 2.52 MB, significantly outperforming mainstream instance segmentation models such as Mask R-CNN (98.30% AP) and YOLACT (96.49% AP). The model size was reduced by 68.3% compared to the original YOLOv5n-seg model. In the tomato localization stage, at the range of 280 mm to 480 mm, the average error of the tomato centroid localization was affected by occlusion and sunlight conditions. The maximum average localization error was ±5.0 mm, meeting the localization accuracy requirements of the tomato-picking robots. This study developed a lightweight tomato instance segmentation model and achieved accurate localization of tomato, which can facilitate research, development, and application of fruit-picking robots.

Keywords: YOLOv5n-seg; instance segmentation; binocular camera; RAFT-Stereo; depth point cloud; least squares fitting

Citation: Zheng, S.; Liu, Y.; Weng, W.; Jia, X.; Yu, S.; Wu, Z. Tomato Recognition and Localization Method Based on Improved YOLOv5n-seg Model and Binocular Stereo Vision. *Agronomy* **2023**, *13*, 2339. https://doi.org/10.3390/agronomy13092339

Academic Editor: Paul Kwan

Received: 13 August 2023
Revised: 31 August 2023
Accepted: 3 September 2023
Published: 8 September 2023

1. Introduction

Tomato is one of the major economically important vegetable crops worldwide and is highly favored by consumers for its abundant nutritional value [1]. However, tomato picking is mainly performed manually, leading to low efficiency, high costs, and inadequate quality control issues. To address these challenges, researching fruit-picking robots for automated tomato picking is necessary so as to achieve higher efficiency and lower costs, replacing the traditional yet labor-intensive manual harvesting process [2]. Fruit recognition and localization in the visual system of picking robots is one of the key links to

realizing automatic picking. Therefore, it is of great research significance to achieve efficient recognition and accurate localization of tomatoes during the picking process.

In the early stage, fruit recognition was mainly based on digital image processing techniques [3–5] and machine learning [6–8], relying on manually designed features. Extracting representative semantic information was challenging with complex backgrounds. Meeting situations with serious fruit occlusion, uneven illumination, and other factors was also difficult [9–13]. In recent years, fruit recognition methods based on deep learning have become mainstream [14]. Deep convolutional neural networks can automatically learn features from training data and show stronger fruit target recognition capabilities in complex scenes. Among numerous detection models based on deep convolutional neural networks, YOLO-based detection models have been favored by many researchers for their high accuracy and fast detection speed [15]. For example, Sozzi et al. [16] used six versions of the YOLO model (YOLOv3, YOLOV3-Tiny, YOLOv4, YOLOV4-Tiny, YOLOv5x, and YOLOv5s) to detect and count white grapes. The comparison of model performance showed that YOLOv4-tiny had the best combination of accuracy and speed. Wang et al. [17] used transfer learning to build a fine-tuned YOLOv5s model for apple fruit detection, which solved the problem that small fruits were difficult to detect before fruit thinning. Cardellicchio et al. [18] used the YOLOv5-based one-stage detectors to detect tomato plant phenotyping traits and achieved high scores in detecting nodes, fruits, and flowers. Tian et al. [19] proposed an improved YOLOv3 apple recognition model combined with DenseNet, which can effectively detect overlapping apples and apples under occlusion conditions in orchards. However, these methods can only detect the area where the target is located and cannot realize the segmentation of the target at the pixel level. In contrast, Rong et al. [20] accurately recognized tomatoes through an improved Swin Transformer V2-based semantic segmentation model, which effectively separated the background environment and extracted fruit pixels. Nonetheless, semantic segmentation cannot segment overlapping and sticking fruit edges. On the other hand, Afonso et al. [21] applied the Mask R-CNN model to recognize tomatoes in greenhouse and achieved pixel-level segmentation of each fruit, but the model network structure was not improved to reduce model complexity and size. Jia et al. [22] proposed an improved Mask R-CNN model for green fruit segmentation in complex orchards. The model uses MobileNetV3 as a lightweight backbone network and incorporates a BPR module to refine segmentation masks. However, the overall complexity remains high as a two-stage segmentation model. Liu et al. [23] used an improved YOLACT to identify tomato main stems. The improved YOLACT reduces model size by 16.8% to 165.52 MB using Depthwise separable convolution in the last BottleBlock group. Even so, at 165.52 MB, the model size remains large, leaving substantial room for reducing model complexity.

The common fruit localization methods can be divided into two categories: active ranging and passive ranging. Active ranging relies on a specific light source to illuminate the target fruit and measures depth information by receiving reflected light with a photoelectric sensor, so it is susceptible to outdoor sunlight interference [24–26]. Passive ranging captures data under natural light conditions without the need for specific light sources and utilizes a localization algorithm to calculate depth information [27–29]. Binocular camera stereo vision localization is a mainstream passive ranging method that is widely used in the visual systems of picking robots due to its advantages of low cost, high localization accuracy, and no additional power consumption compared to other localization methods. Binocular camera achieves three-dimensional localization of fruits based on stereo matching, which currently commonly relies on feature-based and region-based methods. Tang et al. [30] adopted the detection box of the same target fruit in the left and right images of binocular cameras as the matching area to reduce the amount of matching calculation and improve the matching effect. The experimental results showed that, under sunlight and shading conditions, the errors measured for fruits on the tree were 23.568 ± 7.420 mm and 23.524 ± 7.428 mm, respectively. Liu et al. [31] calculated the disparity and average depth value between the pineapple feature points in the binocular image when locating

pineapples. The maximum absolute error and average absolute error in the locating results of 20 groups of pineapple images were 42.910 mm and 24.414 mm, respectively.

The aforementioned research has made great progress in fruit recognition and localization. However, the complexity and size of recognition models remain high, resulting in limited detection speed and unfavorable deployment on end devices. Traditional disparity matching methods have insufficient accuracy for tomato localization and are susceptible to interference, lacking computation of tomato size to provide adequate information for robotic picking. Therefore, this study aimed to develop a lightweight tomato recognition model and enhance the accuracy of tomato localization based on binocular camera. The YOLO-TomatoSeg instance segmentation model was proposed, which is an efficient and lightweight tomato instance segmentation model for obtaining masks of ripe tomatoes improved on the basis of YOLOv5n-seg by replacing its backbone network with ShuffleNetV2 building blocks and incorporating a Squeeze-and-Excitation (SE) attention module. High-quality disparity generated by RAFT-Stereo [32], which is a deep learning model for disparity estimation, and calibrated binocular camera parameters were used to compute depth point clouds of masked ripe tomato regions. After downsampling and outlier removal preprocessing, least squares fitting of the point clouds was performed to derive tomato centroid co-ordinates and radius size, achieving precise localization.

2. Materials and Methods

2.1. Image Acquisition and Dataset Construction

The recognition and localization method proposed in this study is mainly applied to tomatoes grown in greenhouses. The tomato images used in the tomato recognition experiments in this study were taken at the plastic film greenhouse located in the Yinong Agricultural Base, Changle District, Fuzhou City, Fujian Province, China, which is situated at longitude 119.47° E and latitude 25.91° N with an altitude of 4 m. The tomato variety is Syngenta Sibede, cultivated in seasonal planting, with two plantings per year in spring and winter. The fruits turn orange–red when ripe. The image acquisition equipment consisted of two MV-SUA134GC industrial cameras (Shenzhen Mindvision Technology Co., Ltd., Shenzhen, China). Following the guidance of agronomists, a certain number of ripe tomatoes were photographed, which included ripe tomatoes under different degrees of tomato plant foliage occlusion (with no occlusion, slight occlusion with less than 20% covered by leaves, moderate occlusion with 20–60% covered, and heavy occlusion with more than 60% covered), with varying levels of fruit overlapping (with minimal overlap covering less than 10%, moderate overlap covering 10–50%, and severe overlap covering over 50%), and under different lighting conditions (in shade, indirect sunlight, and direct sunlight). After removing the unclear images, the collected image dataset contains a total of 1734 images, saved as 1024 × 1024 RGB images.

To increase the sample diversity by including as many types of complex scenes in the greenhouse as possible, so as to improve the robustness and generalization ability of the model trained on the dataset samples [33], offline data augmentation was performed using 8 methods of OpenCV, including scaling, blurring, brightening, darkening, rotating, flipping, warping, and adding noise. Examples are shown in Figure 1. Brightening and darkening simulate scenes with varying light conditions in the greenhouse; rotating, flipping, and scaling simulate different shooting positions and angles; and warping and adding noise simulate artifacts that could be introduced during image capture and processing. Finally, a dataset containing 3468 augmented images was constructed. The dataset was divided into a ratio of 8:2 for training and testing. A total of 2714 images were used as the training set and 754 images as the test set. LabelMe was utilized to manually annotate the ripe tomatoes in the constructed dataset, marking a total of 4791 ripe tomato annotations. The annotation results were checked by several agronomists to ensure the accuracy of ripe tomato annotations.

Figure 1. Examples of image augmentation. (**a**) Original image; (**b**) scaling; (**c**) blurring; (**d**) brightening; (**e**) darkening; (**f**) rotating; (**g**) flipping; (**h**) warping; (**i**) adding noise.

2.2. System Framework of Tomato Recognition and Localization Algorithm

The system framework consists of two modules: tomato instance segmentation and tomato fitting localization. For tomato instance segmentation, the YOLO-TomatoSeg model segmented tomatoes and generated instance masks. It was generated by replacing the backbone of the YOLOv5n-seg model with ShuffleNetV2 building blocks and incorporating an SE attention module, thus making the segmentation model more lightweight; as for tomato fitting localization, RAFT-Stereo estimated the disparity between rectified binocular images. The original depth point cloud was then computed based on the estimated disparity and the parameters obtained after binocular camera calibration. Next, it was clipped by tomato mask regions from instance segmentation to acquire tomato depth point clouds. Finally, after preprocessing, tomato centroid co-ordinates and radii were calculated via least squares sphere fitting. The system framework is shown in Figure 2.

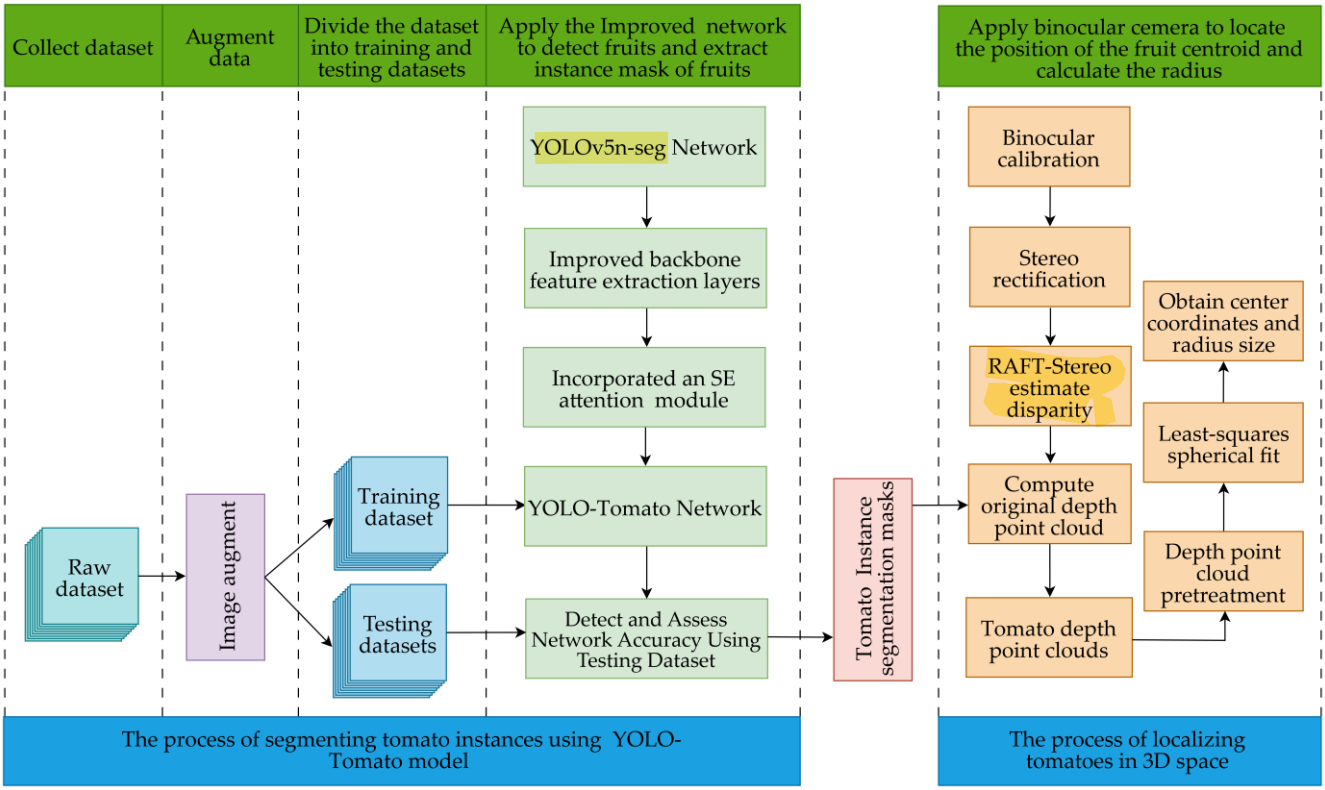


Figure 2. System framework diagram.

2.3. Tomato Recognition Model

2.3.1. Principle of Yolov5n-seg Model

YOLOv5n-seg is a one-stage instance segmentation model released by Ultralytics in 2022. As shown in Figure 3, the network architecture consists of four components: Input, Backbone, Neck, and Head. The Input uses Mosaic data augmentation to increase data complexity; adapts input images of different sizes to a fixed size through adaptive image scaling; and calculates preset fixed Anchors for subsequent training based on adaptive anchor box calculation. The Backbone, which is mainly used for feature extraction in instance segmentation, consists of the convolutional module CBS, the C3 modules CSP1_X and SP2_X, where CSP2_X contains residual modules compared to CSP1_X, and the SPPF module which is generated on the basis of Spatial Pyramid Pooling (SPP) [34] structure by changing the three parallel max pooling layers in SPP to a serial connection to reduce the computational cost in multi-feature fusion. It changes the three parallel max pooling layers in SPP to a serial connection to reduce the computational cost in multi-feature fusion. The Head contains two parallel branches, one of which uses the Prototype Network to generate a series of prototype masks and the other adds an additional output containing the mask coefficients of each detection frame based on the object detection output, so that high-quality masks corresponding to the target can be obtained by matrix multiplication of the outputs of the two branches.

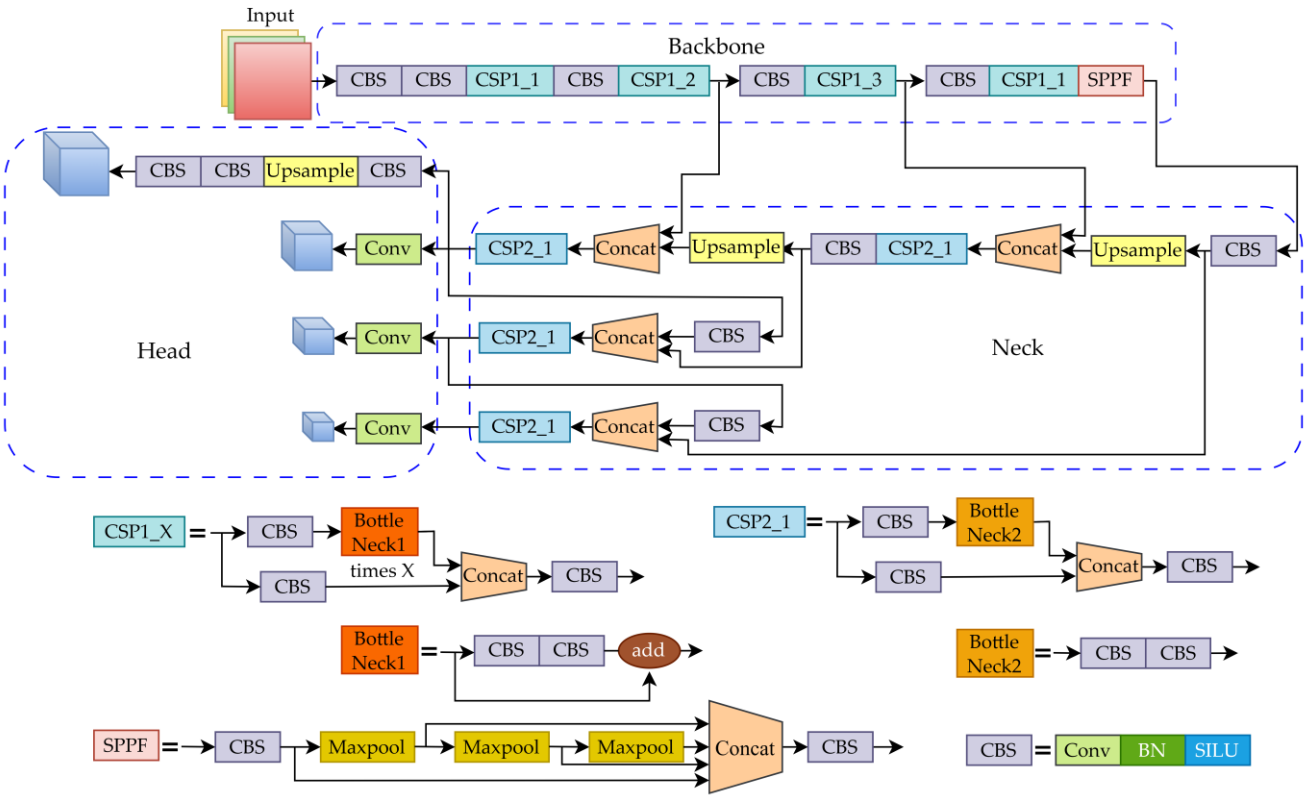


Figure 3. YOLOv5n-seg structure diagram.

2.3.2. Improvement of the Backbone Network

The backbone network in Yolov5n-seg contains multiple deep convolutional modules, which can enhance the model's feature extraction capabilities but also lead to larger model size, increased model complexity, and difficulties in end device deployment. Therefore, the lightweight ShuffleNetV2 building blocks were used to replace the backbone network to lightweight the model.

The ShuffleNetV2 building block consists of basic units and downsample units, as shown in Figure 4. The basic unit introduces the channel split operation to split the input feature map into two branches along the channel dimension: the number of channels is C′ and C-C′, respectively. The left branch performs equal mapping, while the right branch goes through three continuous convolutions with stride 1, keeping the output channels the same as the input channels. Among them, two 1×1 convolutions are standard convolutions and one 3×3 convolution is a depthwise convolution. Depthwise convolution, commonly as a key component in lightweight networks [35–38], together with the subsequent 1×1 convolution, constitutes the depthwise separable convolution, which significantly reduces parameters and computational cost in the convolutional layer while adding a feature interaction process. The outputs of the two branches are concatenated and then shuffled to enhance the fusion of information between channels. Compared to the basic unit, the downsample unit's left branch serially adds a 3×3 depthwise convolution and 1×1 standard convolution, and the depthwise convolutions in both branches have a stride of 2. Channel splitting is omitted, which directly increases the number of network channels and width, thereby enhancing the network's feature extraction capabilities.

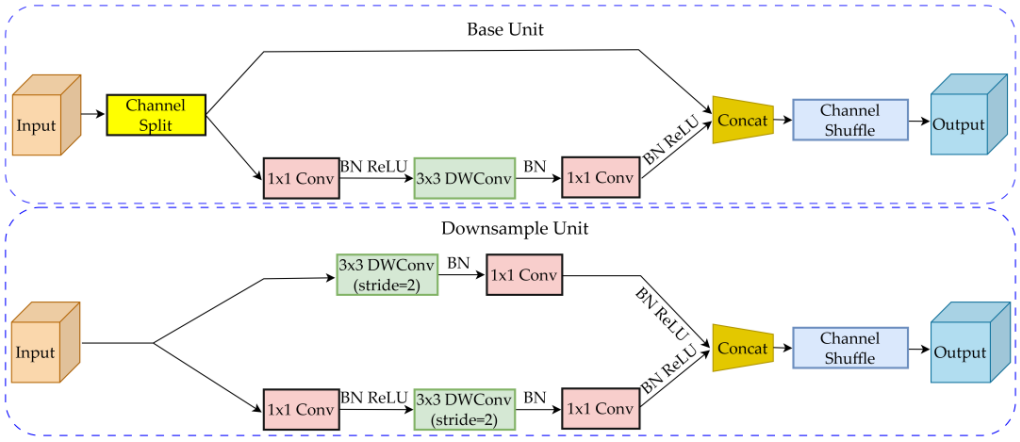


Figure 4. ShuffleNetV2 building block.

2.3.3. Incorporation of Attention Module

The dataset used in this study was captured inside tomato greenhouses, containing a large amount of environmental background interference and other noncritical features. After increasing the number of channels through the convolutional layers, the feature importance across channels becomes uneven. Simply treating features from different channels with the same level of importance would lead to the suppression of important features, causing some negative effects on the model. Therefore, the SE attention module was incorporated into the backbone network of the model to selectively emphasize important channel features and suppress less useful ones in order to handle the problem of imbalanced channel features.

The main operations of the SE module are squeeze and excitation. The squeeze operation squeezes the entire feature map into a vector Z through global average pooling and generates a channel descriptor for each channel, thereby introducing global information. The excitation operation is used to obtain the correlation between channels and preserve the channels with informative features while suppressing the ones with less informative features. As shown in Figure 5, X is the original input feature map and H', W', and C' are the spatial height, spatial width, and number of channels of the original input feature map, respectively. After the F_{tr} (transformation) operation, the output feature map is U, with spatial height H, spatial width W, and spatial channels C. Then, the F_{sq} (squeeze) operation squeezes the feature map, F_{ex} (excitation) excites the feature map, and F_{scale} (scale) performs channel-wise multiplication of the excitation value vector and the feature map U to obtain the calibrated feature map $\tilde{X}$.

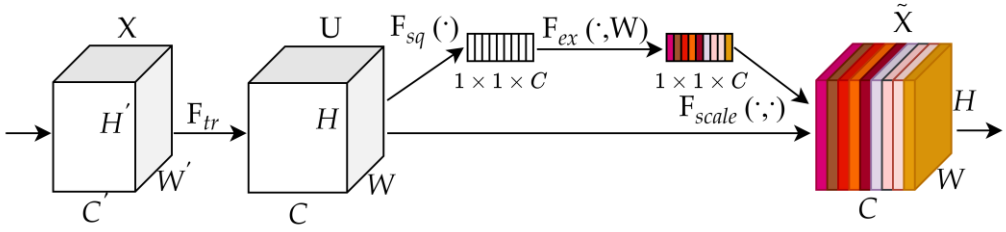


Figure 5. SE attention module structure.

2.3.4. YOLO-TomatoSeg Instance Segmentation Model

YOLO-TomatoSeg was generated after improving the backbone network of YOLOv5n-seg and incorporating an SE attention. The structure of the YOLO-TomatoSeg is shown in Figure 6. CBRM represents a convolutional module composed of convolutional, batch normalization, Rectified Linear Unit (ReLU), and max pooling; SN_Block_X represents a net-

work structure with one downsample unit and X repeated base units from ShuffleNetV2 building block connected in series.

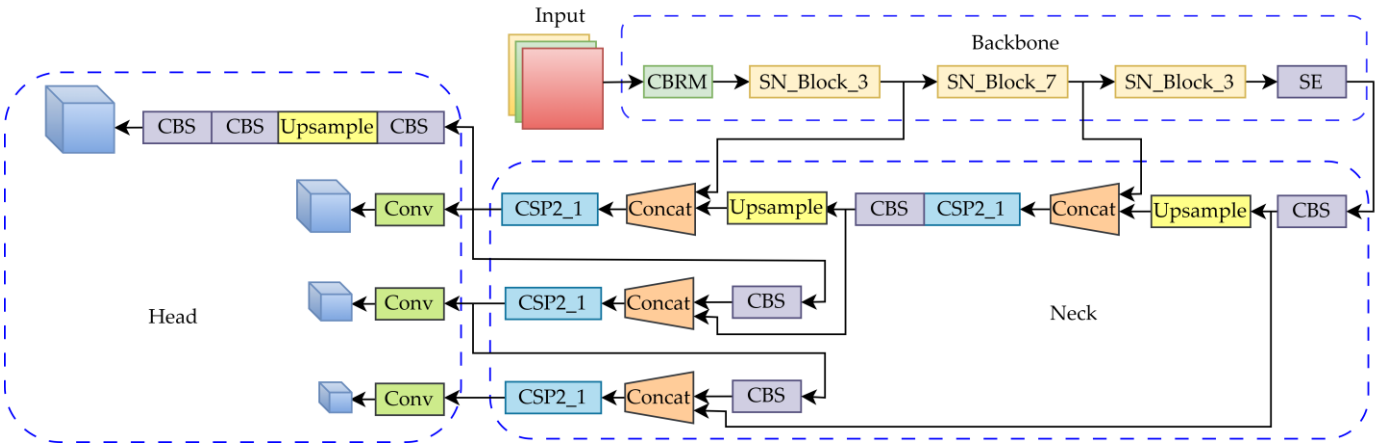


Figure 6. YOLO-TomatoSeg structure diagram.

2.4. Tomato Depth Point Cloud Acquisition and Fitting

2.4.1. Binocular Camera Calibration

To establish an effective imaging model, camera calibration was performed using the Zhang's calibration method [39]. The binocular camera consists of two MV-SUA134GC industrial cameras connected to the computer via USB cables, as shown in Figure 7. The calibration chessboard has 12 × 9 corner points, with each grid of 15 mm length and accuracy error within 0.001 mm. The calibration software employed the Stereo Camera Calibrator toolbox in MATLAB R2016b. In total, 20 groups of 1024 × 1024 resolution stereo image pairs were captured from different angles of the calibration board, with the left and right images imported into the Stereo Camera Calibrator toolbox for detection and calibration. Table 1 shows the calibration parameters obtained for the binocular camera after calibration, which can be used for stereo rectification and depth computation of binocular images.

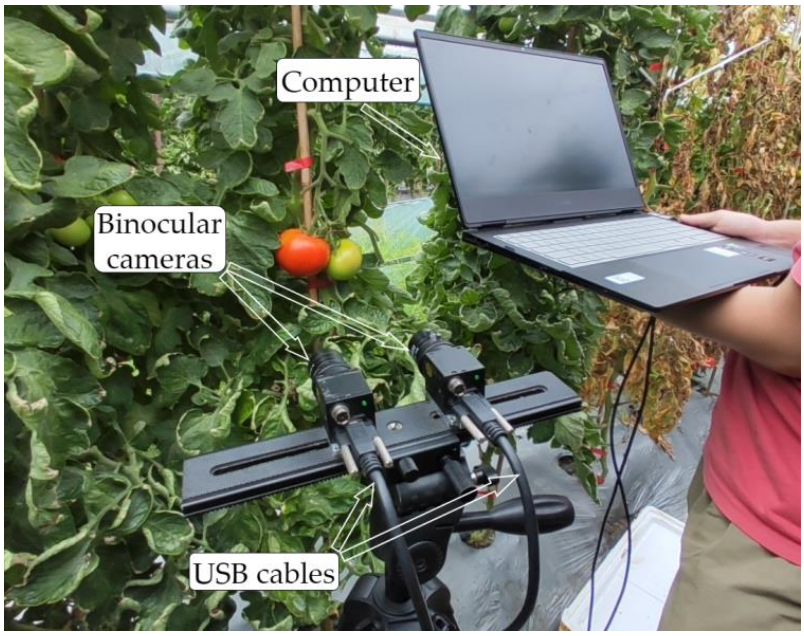


Figure 7. The connection status of the binocular camera.

Table 1. Binocular camera calibration parameters.

Stereo Parameters	Left Camera	Right Camera
f	$[1122.700 \quad 1122.500]$	$[1130.6 \quad 1130.4]$
Intrinsic matrix	$\begin{bmatrix} 1122.700 & 0 & 0 \\ 0 & 1122.500 & 0 \\ 325.913 & 234.467 & 1 \end{bmatrix}$	$\begin{bmatrix} 1130.600 & 0 & 0 \\ 0 & 1130.400 & 0 \\ 330.586 & 259.772 & 1 \end{bmatrix}$
Translation	$[-74.676 \quad -0.168 \quad 0.490]$	
Baseline length	74.676	

2.4.2. Disparity Estimation Using RAFT-Stereo

RAFT-Stereo is a deep learning model based on the optical flow estimation network RAFT [40]. In this study, it was utilized to estimate the disparity between rectified binocular image pairs. Compared to traditional stereo matching algorithms, RAFT-Stereo demonstrates stronger adaptability and robustness in handling challenges such as lack of texture and occlusion. The disparity calculation structure of RAFT-Stereo involves three main components: feature extractor, correlation pyramid, and GRU-based update operator. The calculation process is shown in Figure 8. Relevant features are extracted from the rectified left and right images to construct the correlation pyramid, while context image features and initial hidden states are extracted from the context encoder and the disparity field is initialized to zero. Then, at each iteration, the GRU module samples from the correlation pyramid based on the current disparity estimate to obtain correlation features. The initial image features, current hidden state, and correlation features are combined and input to the GRU to produce an updated hidden state and an updated disparity. After multiple iterations, the final smooth disparity map is generated.

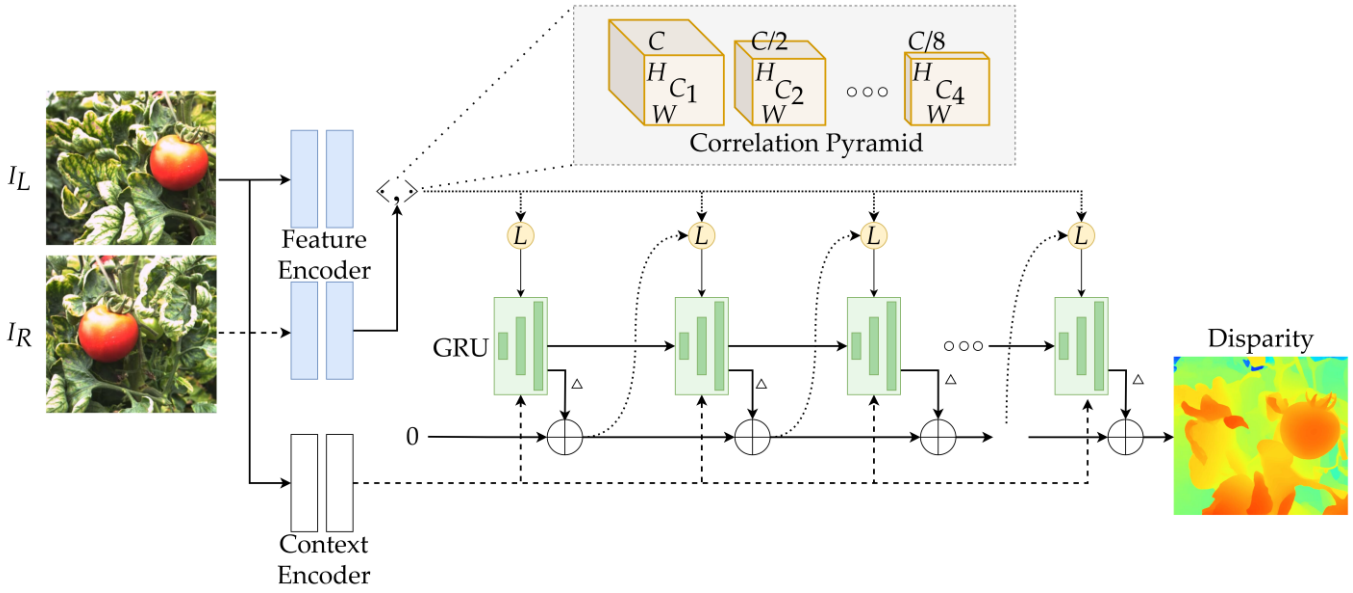


Figure 8. Disparity estimation process of RAFT-Stereo.

2.4.3. Tomato Depth Point Cloud Generation

The ideal binocular stereo vision model after stereo rectification using binocular camera calibration parameters is shown in Figure 9. In the figure, the focal lengths of both cameras are f, the optical centers are O_L and O_R, respectively, and the distance between the two optical centers is called the baseline B. $O_L O_l$ and $O_R O_r$ are two lines perpendicular to the imaging planes of the cameras, called the optical axes of the left and right cameras. The feature point $P(X, Y, Z)$ in the 3D space is captured by both cameras at the same time and imaged at $Q_l(x_l, y_l)$ on the left camera image plane and $Q_r(x_r, y_r)$ on the right camera

image plane. y_l equals y_r. Setting the world co-ordinate origin at the left camera optical center O_L, O_L space co-ordinates $(0,0,0)$, O_R space co-ordinates $(B,0,0)$. The following Equation (1) can be obtained through similar triangle geometry:

$$\begin{cases} \chi_\iota = f\dfrac{X}{Z} \\ \dfrac{B-(\chi_l-\chi_r)}{B} = \dfrac{Z-f}{Z} \\ y = f\dfrac{Y}{Z} \end{cases} \tag{1}$$

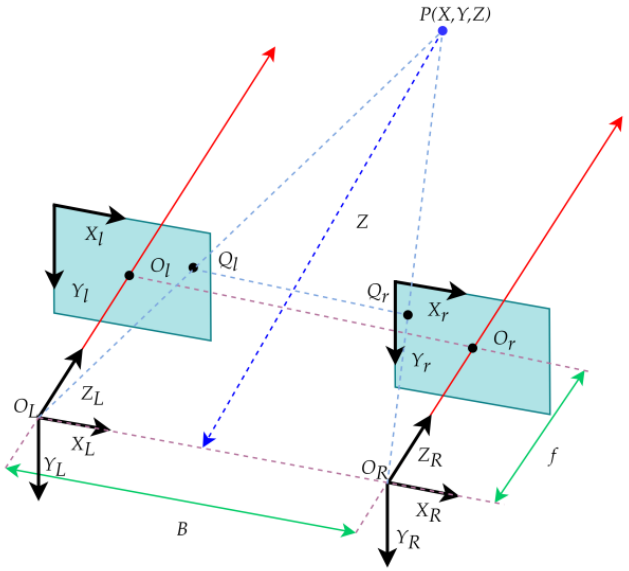


Figure 9. Ideal binocular stereo vision model.

The disparity is $d = |x_l - x_r|$. By transforming the above equations, the 3D space co-ordinates of point P in the left camera co-ordinate system can be solved:

$$\begin{cases} X = \dfrac{Bx_l}{d} \\ Y = \dfrac{By}{d} \\ Z = \dfrac{Bf}{d} \end{cases} \tag{2}$$

Using the disparity calculated by RAFT-Stereo and the focal length f and baseline length B from the binocular camera calibration parameters, the 3D space co-ordinates of the target point can be reconstructed through Equation (2) to generate the original depth point cloud.

After obtaining the mask of each ripe tomato through the YOLO-TomatoSeg instance segmentation model, the clipping operation was performed on the original depth point cloud using the masks to remove the background points, leaving the ripe tomato depth point cloud regions. Figure 10a shows the original depth point cloud diagram, while Figure 10b shows the clipped ripe tomato depth point cloud diagram. Both diagrams were plotted using the Open3D library.

(**a**) (**b**)

Figure 10. Depth point cloud visualization results. (**a**) Original depth point cloud; (**b**) ripe tomato depth point cloud.

2.4.4. Tomato Depth Point Cloud Preprocessing

To reduce subsequent fitting computation and improve accuracy, preprocessing on ripe tomato depth point clouds was required, including downsampling and removing outliers. The voxel downsampling method was used for downsampling, which divided the tomato depth point cloud data into a series of voxels, calculated the centroid of the non-empty voxels to replace all the points in the voxel to reduce the number of points, and the voxel centroid calculation equation is as follows:

$$\begin{cases} X_{centeroid} = \dfrac{\sum_{i=1}^{m} X_i}{m} \\ Y_{centeroid} = \dfrac{\sum_{i=1}^{m} Y_i}{m} \\ Z_{centeroid} = \dfrac{\sum_{i=1}^{m} Z_i}{m} \end{cases} \tag{3}$$

where $(X_{centeroid}, Y_{centeroid}, Z_{centeroid})$ is the calculated centroid co-ordinates of each voxel, (X_i, Y_i, Z_i) is the co-ordinates of each point in the voxel, and m is the number of points contained in each voxel.

The Local Outlier Factor (LOF) algorithm was used as the method of removing outliers in the depth point cloud [41], which is a density-based outlier detection method. By assigning an outlier factor that depends on local density to each point in the depth point cloud and comparing it with a given threshold to determine whether the data point is an outlier. The outlier factor of the object point is calculated as:

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \dfrac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} \tag{4}$$

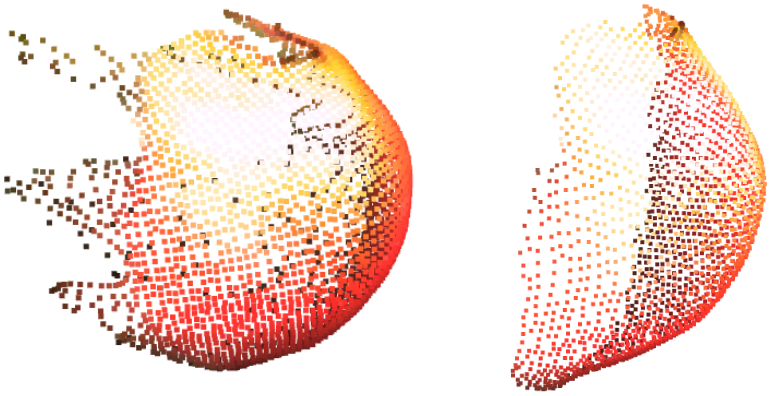
where $N_k(p)$ is the k-distance neighborhood of the object point p containing k nearest neighbor points and $lrd_k(p)$ and $lrd_k(o)$ are the local reachability densities of the object points p and o within the k-distance neighborhood of p, respectively. The local reachability density is an indicator used to measure the density around a point, which is determined by calculating the reachability distance between the object point and its k nearest neighbors.

Equation (5) is the calculation method of the local reachability density of any point in the depth point cloud:

$$lrd_k(p) = 1/\left[\frac{\sum_{o \in N_k(p)} rd_k(p,o)}{|N_k(p)|}\right] \tag{5}$$

where $rd_k(p,o)$ represents the k-reachability distance of point p with respect to point o, defined as the maximum of the distance from point p to its kth nearest neighbor and the distance from point p to point o.

After applying voxel downsampling and LOF algorithm sequentially on each ripe tomato depth point cloud, the number of points was reduced while retaining the key feature points. Figure 11a shows the tomato depth point cloud after downsampling. Figure 11b shows the tomato depth point cloud after removing outliers.



(**a**) (**b**)

Figure 11. Depth point cloud visualization after preprocessing. (**a**) Downsampled depth point cloud; (**b**) outliers removed depth point cloud.

2.4.5. Spherical Fitting of Tomato Depth Point Clouds

The fitted tomato shape from the depth point cloud of the tomato plays an important role in the actual tomato-picking operations conducted by tomato-picking robots. The picking robot can adjust the grasping strategy based on the size of the tomato shape, such as controlling the grasping position and grasping force, thereby reducing the possibility of damage to the tomatoes and improving the efficiency of picking. Since the shape of tomatoes is close to spherical and the preprocessed depth point cloud of the tomato surface is close to a hemispherical surface, spherical fitting via the least squares method was chosen to fit the tomato shape [42]. The basic principle is to minimize the sum of squares of distances from all points in the depth point cloud to the fitted spherical surface, with the constructed equation being:

$$E(x_c, y_c, z_c, r) = \sum_{i=0}^{n}\left((x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2 - r^2\right)^2 \tag{6}$$

where (x_c, y_c, z_c) and r are the center co-ordinates and radius of the sphere during the iterative optimization process, (x_i, y_i, z_i) is the coordinates of each point in the depth point cloud, and n is the number of points.

3. Results

3.1. Experimental Environment and Parameter Settings

The hardware used in these experiments was Intel i9-12900H CPU, Nvidia RTX 3060 6 GB GPU, and 16 GB RAM. The software environment was Windows 11, PyTorch 1.12.0, CUDA 11.6, Python 3.9, Open3D 0.17.0, and MATLAB R2016b. The training parameters of

the YOLO-TomatoSeg instance segmentation model were adjusted repeatedly according to the training situation with reference to the recommendations of previous related research. The final specific training parameter settings were as follows: initial learning rate was 0.01, momentum parameter was 0.937, optimizer weight decay was 0.0005, batch size was 6, number of training epochs was 300, and Stochastic Gradient Descent (SGD) optimizer was used to update parameters. The pretrained weights from the partial ShuffleNetV2 model provided by PyTorch were loaded into the training process of YOLO-TomatoSeg. Specifically, the ShuffleNetV2 building blocks used to replace the backbone of YOLOv5n-seg contain pretrained weights from ImageNet. By utilizing these pretrained weights for training, transfer learning was leveraged to accelerate convergence and boost model performance. With no special instructions, all instance segmentation models described below employ pretrained weights during training.

3.2. Evaluation Metrics

To evaluate the effects of the YOLO-TomatoSeg, this study mainly used Precision, Recall, Average Precision (AP), Floating Point Operations (FLOPs), Params (Parameters), Size, and Frames Per Second (FPS) as evaluation metrics for the model. Precision refers to the ratio of the number of correctly detected fruit pixels to the total number of detected fruit pixels. Recall refers to the ratio of the number of correctly detected fruit pixels to the total actual number of fruit pixels. AP is defined as the area obtained by integrating the smoothed Precision–Recall curve. The specific formulas for calculating these metrics are shown in Equations (7)–(9):

$$Recall = \frac{TP}{(TP + FN)} \tag{7}$$

$$Precision = \frac{TP}{(TP + FP)} \tag{8}$$

$$AP = \int_0^1 P(R)dR \tag{9}$$

where TP, FN, and FP represent the number of correctly detected tomato pixels, the number of undetected tomato pixels, and the number of falsely detected tomato pixels, respectively. $P(R)$ is the smoothed Precision–Recall curve.

3.3. Impact of Improved Backbone Network on Segmentation Results

To demonstrate the advantages of using the building blocks of ShuffleNetV2 to improve the backbone network compared with other lightweight methods, multiple comparative experiments were conducted. Table 2 shows the performance of the tomato instance segmentation models with different backbone networks. As can be seen from the data in the table, the lightweighting of the backbone network resulted in a reduced number of convolutions, which, in turn, led to a relatively shallower model depth. This reduction in depth then slightly weakened its feature extraction capability, which then caused a certain degree of decrease in AP for all the segmentation models with lightweighted backbones compared to the original YOLOv5n-seg model. However, the decreases in AP were small, with a maximum of only 0.4 percentage points. Among the lightweight methods, using ShuffleNetv2 building blocks to improve the backbone network achieved the best lightweight performance. Under the condition that the Precision, Recall, and AP did not differ much, compared with other lightweight methods, the number of parameters was lower, reduced by 38.9% from the original segmentation model, and the model size was smaller, only 31.6% of the original segmentation model size, meeting the lightweight requirements of the tomato instance segmentation model.

Table 2. Performance of instance segmentation models with different backbones.

Model	Precision/%	Recall/%	AP/%	FLOPs/G	Params/M	Size/MB
YOLOv5n-seg	97.1	96.3	98.87	6.9	1.8	7.94
YOLO_Mobile	96.5	96.3	98.79	5.4	1.9	4.08
YOLO_Efficient	97.2	96.8	98.52	4.5	9.2	9.11
YOLO_Shuffle	97.5	95.9	98.45	4.6	1.1	2.51

Note: YOLO_Mobile, YOLO_Efficient, and YOLO_Shuffle are enhanced versions of YOLOv5n-seg, utilizing MobileNetV3, EfficientNet-B0, and ShuffleNetV2 building blocks to improve the backbone network.

3.4. Impact of Incorporated Attention Module on Segmentation Results

To verify the effectiveness of incorporating the SE attention module, the YOLO_Shuffle model generated by replacing the backbone network of YOLOv5n-seg with the building blocks of ShuffleNetV2 was used as the baseline model. The SE attention module was incorporated into the last layer of the baseline model's backbone network. Then, the SE module was sequentially substituted with other lightweight attention modules for comparison. The comparison experimental results are shown in Table 3. As can be seen from the results, since the incorporated attention modules were all lightweight, the FLOPs, Params, and Size changed very little for models that incorporated different attention modules compared to YOLO_Shuffle. Additionally, the AP all increased slightly, with the model incorporating the SE attention module achieving the highest increase in AP of 0.56 percentage point to 99.01%, slightly surpassing the AP of YOLOv5n-seg in Table 2. However, compared to YOLOv5n-seg, the segmentation model incorporating the SE attention module still had significantly reduced FLOPs by 33.3%, Params by 38.8%, and Size by 68.3%, thus resulting in a 0.4 percentage point decrease in Precision.

Table 3. Performance of instance segmentation models incorporating different attention modules.

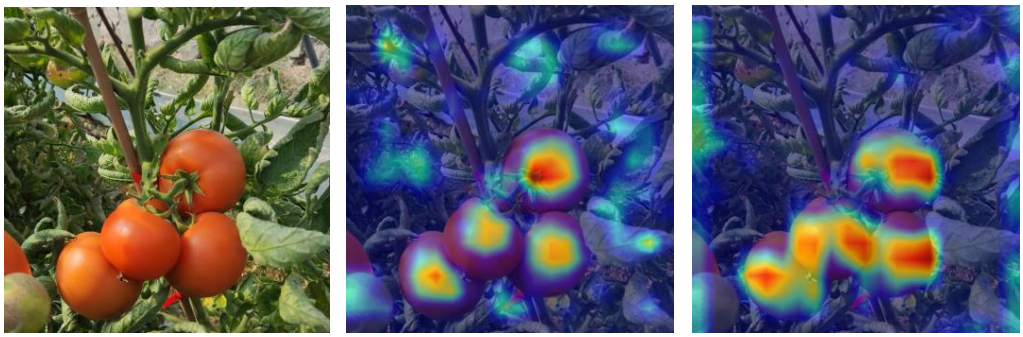
Model	Precision/%	Recall/%	AP/%	FLOPs/G	Params/M	Size/MB
YOLO_Shuffle	97.5	95.9	98.45	4.6	1.1	2.51
YOLO_Shuffle + ECA	96.9	96.0	98.74	4.6	1.1	2.52
YOLO_Shuffle + CBAM	97	95.8	98.73	4.6	1.1	2.52
YOLO-TomatoSeg	96.7	96.3	99.01	4.6	1.1	2.52

Note: YOLO_Shuffle + ECA, YOLO_Shuffle + CBAM, and YOLO-TomatoSeg refer to YOLO_Shuffle models improved by incorporating ECA, CBAM, and SE attention modules, respectively.

Incorporating the SE attention module can emphasize important channel features, thus reducing the influence of unimportant features. Grad-CAM [43] was used to visualize and analyze the features extracted by the backbone networks of the YOLOv5n-seg model and the YOLO-TomatoSeg model, as shown In Figure 12. The brighter regions in the heat map indicate that the features in this region are more important. As can be seen from the response levels of various regions in the heatmaps, compared to the YOLOv5n-seg model, the YOLO-TomatoSeg model with the SE attention module pays better attention to the feature regions corresponding to ripe tomatoes.

3.5. Comparative Experiments of Different Instance Segmentation Models

To evaluate the superiority of the YOLO-TomatoSeg instance segmentation model, the performance of several mainstream instance segmentation models on the constructed dataset was studied, including YOLACT, another one-stage instance segmentation model, as well as Mask R-CNN, a two-stage instance segmentation model. The comparison results are shown in Table 4.



(**a**) (**b**) (**c**)

Figure 12. Visualization analysis of Grad-CAM. (**a**) Input image; (**b**) YOLOv5n-seg; (**c**) YOLO-TomatoSeg.

Table 4. Performance comparison of different instance segmentation models.

Model	AP/%	Size/MB	FPS
YOLACT	96.49	189	11.1
Mask R-CNN	98.30	334	7.7
YOLO-TomatoSeg	99.01	2.27	70.0

As can be seen from Table 4, compared to the other two instance segmentation models, while increasing AP, the YOLO-TomatoSeg model significantly reduced the model size, reducing by 98.8% compared to the YOLACT model and 99.3% compared to the Mask R-CNN model. The segmentation speed was also remarkably improved, increasing by 530.6% compared to the YOLACT model and 809.1% compared to the Mask R-CNN model. This makes the YOLO-TomatoSeg model easier to deploy on end devices and better meets the needs of actual picking operations.

Figure 13 shows a comparison of actual instance segmentation results on tomatoes under different scenarios by different models in Table 4. The segmentation results reveal that YOLACT had difficulties detecting irregularly shaped tomatoes, missed heavily occluded tomatoes, and poorly segmented boundaries of overlapping tomatoes. In comparison, Mask R-CNN and YOLO-TomatoSeg performed better in detecting irregular shapes, occlusion, and overlap. However, YOLO-TomatoSeg generated finer mask contours along fruit boundaries than Mask R-CNN.

3.6. Tomato Shape Fitting and Localization Experiments

To investigate the localization accuracy after fitting of tomatoes captured under varying occlusion and lighting conditions, 36 pairs of binocular images were acquired in a greenhouse with shooting distances ranging from 280 mm to 480 mm using a binocular camera. Half of the tomato images contained occlusions, while the other half did not. Within each of these two categories, half of the images were taken under sunlight and the other half under shading. The actual diameter of the tomato was measured using a digital caliper. The actual depth value of the tomato relative to the left camera optical center plane was measured by a laser rangefinder, and the depth value plus the radius was used as the ground truth distance from the tomato centroid to the left camera. The average errors of the fitted centroid localization and radius after experiments are shown in Table 5, and Figure 14 depicts the fitting results.

Figure 13. Effect comparison of different instance segmentation models. (**a**) Input image; (**b**) YOLACT; (**c**) Mask R-CNN; (**d**) YOLO-TomatoSeg.

Table 5. Performance of fitting algorithms under different occlusion and lighting conditions.

Occlusion	Lighting	Localization Average Error/mm	Radius Average Error/mm
Exist	Sunlight	5.0	3.9
	Shading	3.5	2.8
Not exist	Sunlight	4.2	3.3
	Shading	2.8	2.1

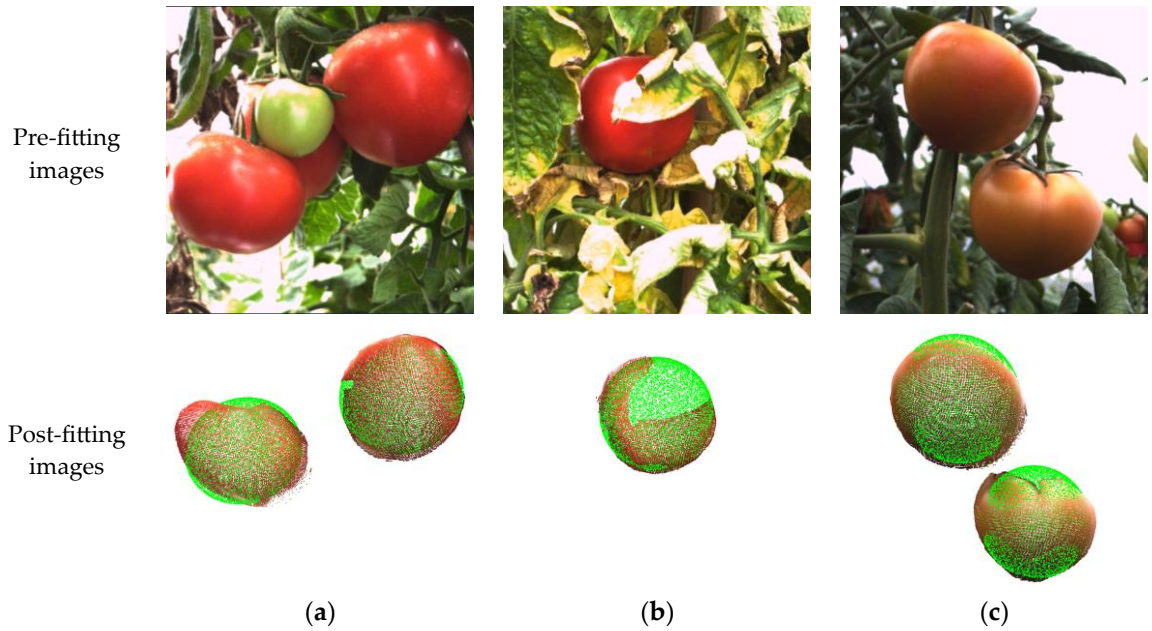


Figure 14. Fitting result figures. (**a**) Under sunlight; (**b**) with occlusion; (**c**) irregular shape.

The results in Table 5 show that both occlusion and sunlight conditions introduced additional errors to the located centroid distance and calculated radius size after tomato shape fitting, with sunlight having a greater impact than occlusion. As shown in Figure 14a, the

sunlight could create locally overexposed bright regions on the tomato surface, saturating pixel values and losing detailed information that affected the accuracy of disparity estimation, leading to large errors in the depth values of the depth point cloud and negatively impacting the fitting. Figure 14b shows that occlusion caused a reduction in the effective tomato surface area used to generate the depth point cloud, resulting in an insufficient point cloud density and reduced ability to represent the 3D shape of the tomato. Other sources of errors in the located centroid distance and calculated radius size after tomato shape fitting include inherent errors in the disparity estimation algorithm causing mismatching between left and right image pixels and anomalous depth point clouds; violations of the equal focal length assumption of the ideal stereo model due to focal length differences between the two cameras; and the irregular tomato surface shapes that could not be perfectly fitted by a regular spherical model, as shown in Figure 14c, and that also impeded accurate ground truth measurement of the tomato radius size and centroid distance.

4. Conclusions

In this study, a lightweight tomato instance segmentation model called YOLO-TomatoSeg and an accurate tomato fitting localization method were proposed, and multiple experiments were conducted to validate the efficiency of YOLO-TomatoSeg in segmenting tomato instances and the adaptability and accuracy of the tomato fitting localization method in localizing tomatoes under various conditions, with the following conclusions:

(1) In the tomato instance segmentation stage, this study improved upon the YOLOv5n-seg model by using ShuffleNetV2 building blocks to enhance the backbone network and incorporating an SE attention mechanism module. This achieved significant reductions in the model's FLOPs, Params, and Size while maintaining high segmentation accuracy. The Params were reduced by 38.9% and the Size was reduced by 68.3% of the original model. Therefore, the model was made lightweight, improving recognition efficiency.

(2) This study compared the performance of the YOLO-TomatoSeg model with other mainstream instance segmentation models, Mask R-CNN and YOLACT. The results demonstrated that the YOLO-TomatoSeg model had obvious advantages over the other models in terms of segmentation accuracy, speed, and model size. It also had lower miss and false detection probabilities, higher mask boundary refinement, a maximum segmentation speed increase of 809.1%, and a maximum model size reduction of 99.3%.

(3) In the tomato localization stage, the disparity map was generated using RAFT-Stereo and combined with binocular camera parameters to calculate the depth point cloud. After clipping and preprocessing, the point cloud was fitted to the tomato shape using least squares fitting. Fitting experiments under different occlusion conditions and lighting conditions at distances of 280–480 mm showed that overexposed points under sunlight had the greatest impact on localization accuracy. The maximum localization error was within $\pm$5.0 mm and the maximum radius error was within $\pm$3.9 mm, meeting the localization requirements for fruit-picking robots.

(4) Overall, the experimental results showed that the YOLO-TomatoSeg model could accurately perform tomato instance segmentation in complex greenhouse environments, and its lightweight level was high, making it more suitable for deployment on picking robots. Moreover, the RAFT-stereo deep learning model used in this study for binocular image stereo matching could obtain high-quality disparity maps, thereby achieving high localization accuracy under different conditions.

Author Contributions: Conceptualization, Y.L.; methodology, Y.L. and X.J.; software, Y.L.; validation, X.J.; formal analysis, Y.L.; investigation, S.Y., Z.W. and X.J.; resources, Y.L., X.J., S.Y. and Z.W.; data curation, X.J. and Y.L.; writing—original draft preparation, Y.L.; writing—review and editing, S.Z. and W.W.; visualization, Y.L.; project administration, Z.W., W.W. and S.Z.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Guiding Project of Fujian Provincial Department of Science and Technology, (grant number 2022N0009); Cross Disciplinary Project of Fujian Agriculture and Forestry University, (grant number 000-71202103B).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data are presented in this article in the form of figures and tables.

Acknowledgments: The authors would like to acknowledge the College of Mechanical Electronic Engineering, Fujian Agriculture and Forestry University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Simko, I.; Jia, M.; Venkatesh, J.; Kang, B.; Weng, Y.; Barcaccia, G.; Lanteri, S.; Bhattarai, G.; Foolad, M.R. Genomics and marker-assisted improvement of vegetable crops. *Crit. Rev. Plant Sci.* **2021**, *40*, 303–365. [CrossRef]
2. Li, T.; Sun, M.; He, Q.; Zhang, G.; Shi, G.; Ding, X.; Lin, S. Tomato recognition and location algorithm based on improved yolov5. *Comput. Electron. Agric.* **2023**, *208*, 107759. [CrossRef]
3. Rakun, J.; Stajnko, D.; Zazula, D. Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry. *Comput. Electron. Agric.* **2011**, *76*, 80–88. [CrossRef]
4. Payne, A.; Walsh, K.; Subedi, P.; Jarvis, D. Estimating mango crop yield using image analysis using fruit at 'stone hardening' stage and night time imaging. *Comput. Electron. Agric.* **2014**, *100*, 160–167. [CrossRef]
5. Chaivivatrakul, S.; Dailey, M.N. Texture-based fruit detection. *Precis. Agric.* **2014**, *15*, 662–683. [CrossRef]
6. Zhao, Y.; Gong, L.; Zhou, B.; Huang, Y.; Liu, C. Detecting tomatoes in greenhouse scenes by combining adaboost classifier and colour analysis. *Biosyst. Eng.* **2016**, *148*, 127–137. [CrossRef]
7. Qureshi, W.S.; Payne, A.; Walsh, K.B.; Linker, R.; Cohen, O.; Dailey, M.N. Machine vision for counting fruit on mango tree canopies. *Precis. Agric.* **2017**, *18*, 224–244. [CrossRef]
8. Gongal, A.; Amatya, S.; Karkee, M.; Zhang, Q.; Lewis, K. Sensors and systems for fruit detection and localization: A review. *Comput. Electron. Agric.* **2015**, *116*, 8–19. [CrossRef]
9. Liu, G.; Mao, S.; Kim, J.H. A mature-tomato detection algorithm using machine learning and color analysis. *Sensors* **2019**, *19*, 2013. [CrossRef]
10. Si, Y.; Liu, G.; Feng, J. Location of apples in trees using stereoscopic vision. *Comput. Electron. Agric.* **2015**, *112*, 68–74. [CrossRef]
11. Benavides, M.; Cantón-Garbín, M.; Sánchez-Molina, J.A.; Rodríguez, F. Automatic tomato and peduncle location system based on computer vision for use in robotized harvesting. *Appl. Sci.* **2020**, *10*, 5887. [CrossRef]
12. Wang, C.; Zou, X.; Tang, Y.; Luo, L.; Feng, W. Localisation of litchi in an unstructured environment using binocular stereo vision. *Biosyst. Eng.* **2016**, *145*, 39–51. [CrossRef]
13. 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]
14. Li, R.; Ji, Z.; Hu, S.; Huang, X.; Yang, J.; Li, W. Tomato maturity recognition model based on improved yolov5 in greenhouse. *Agronomy* **2023**, *13*, 603. [CrossRef]
15. Wang, X.; Liu, J. Tomato anomalies detection in greenhouse scenarios based on yolo-dense. *Front. Plant Sci.* **2021**, *12*, 634103. [CrossRef]
16. Sozzi, M.; Cantalamessa, S.; Cogato, A.; Kayad, A.; Marinello, F. Automatic bunch detection in white grape varieties using yolov3, yolov4, and yolov5 deep learning algorithms. *Agronomy* **2022**, *12*, 319. [CrossRef]
17. Wang, D.; He, D. Channel pruned yolo v5s-based deep learning approach for rapid and accurate apple fruitlet detection before fruit thinning. *Biosyst. Eng.* **2021**, *210*, 271–281. [CrossRef]
18. Cardellicchio, A.; Solimani, F.; Dimauro, G.; Petrozza, A.; Summerer, S.; Cellini, F.; Renò, V. Detection of tomato plant phenotyping traits using yolov5-based single stage detectors. *Comput. Electron. Agric.* **2023**, *207*, 107757. [CrossRef]
19. Tian, Y.; Yang, G.; Wang, Z.; Wang, H.; Li, E.; Liang, Z. Apple detection during different growth stages in orchards using the improved yolo-v3 model. *Comput. Electron. Agric.* **2019**, *157*, 417–426. [CrossRef]
20. Rong, Q.; Hu, C.; Hu, X.; Xu, M. Picking point recognition for ripe tomatoes using semantic segmentation and morphological processing. *Comput. Electron. Agric.* **2023**, *210*, 107923. [CrossRef]
21. Afonso, M.; Fonteijn, H.; Fiorentin, F.S.; Lensink, D.; Mooij, M.; Faber, N.; Polder, G.; Wehrens, R. Tomato fruit detection and counting in greenhouses using deep learning. *Front. Plant Sci.* **2020**, *11*, 571299. [CrossRef] [PubMed]
22. Jia, W.; Wei, J.; Zhang, Q.; Pan, N.; Niu, Y.; Yin, X.; Ding, Y.; Ge, X. Accurate segmentation of green fruit based on optimized mask rcnn application in complex orchard. *Front. Plant Sci.* **2022**, *13*, 955256. [CrossRef] [PubMed]
23. Liu, C.; Feng, Q.; Sun, Y.; Li, Y.; Ru, M.; Xu, L. Yolactfusion: An instance segmentation method for rgb-nir multimodal image fusion based on an attention mechanism. *Comput. Electron. Agric.* **2023**, *213*, 108186. [CrossRef]

24. Gené-Mola, J.; Llorens Calveras, J.; Rosell-Polo, J.; Gregorio Lopez, E.; Arnó, J.; Solanelles, F.; Martínez-Casasnovas, J.A.; Escolà, A. Assessing the performance of rgb-d sensors for 3d fruit crop canopy characterization under different operating and lighting conditions. *Sensors* **2020**, *20*, 7072. [CrossRef]
25. Vitzrabin, E.; Edan, Y. Changing task objectives for improved sweet pepper detection for robotic harvesting. *IEEE Robot. Autom. Lett.* **2016**, *1*, 578–584. [CrossRef]
26. Gongal, A.; Silwal, A.; Amatya, S.; Karkee, M.; Zhang, Q.; Lewis, K. Apple crop-load estimation with over-the-row machine vision system. *Comput. Electron. Agric.* **2016**, *120*, 26–35. [CrossRef]
27. Mehta, S.S.; Burks, T.F. Vision-based control of robotic manipulator for citrus harvesting. *Comput. Electron. Agric.* **2014**, *102*, 146–158. [CrossRef]
28. De-An, Z.; Jidong, L.; Wei, J.; Ying, Z.; Yu, C. Design and control of an apple harvesting robot. *Biosyst. Eng.* **2011**, *110*, 112–122. [CrossRef]
29. Zhang, H.; Tang, C.; Sun, X.; Fu, L. A refined apple binocular positioning method with segmentation-based deep learning for robotic picking. *Agronomy* **2023**, *13*, 1469. [CrossRef]
30. Tang, Y.C.; Zhou, H.; Wang, H.J.; Zhang, Y.Q. Fruit detection and positioning technology for a camellia oleifera c. Abel orchard based on improved yolov4-tiny model and binocular stereo vision. *Expert Syst. Appl.* **2023**, *211*, 118573. [CrossRef]
31. Liu, T.; Nie, X.; Wu, J.; Zhang, D.; Liu, W.; Cheng, Y.; Zheng, Y.; Qiu, J.; Qi, L. Pineapple (*Ananas comosus*) fruit detection and localization in natural environment based on binocular stereo vision and improved yolov3 model. *Precis. Agric.* **2023**, *24*, 139–160. [CrossRef]
32. Lipson, L.; Teed, Z.; Deng, J.; Ieee, C.S. RAFT-Stereo: Multilevel recurrent field transforms for stereo matching. In Proceedings of the 2021 International Conference on 3D Vision (3DV), London, UK, 1–3 December 2021; pp. 218–227.
33. Ji, W.; Liu, D.; Meng, Y.; Liao, Q. Exploring the solutions via retinex enhancements for fruit recognition impacts of outdoor sunlight: A case study of navel oranges. *Evol. Intell.* **2022**, *15*, 1875–1911. [CrossRef]
34. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef] [PubMed]
35. Sandler, M.; Howard, A.; Zhu, M.L.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
36. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
37. Chollet, F. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1800–1807.
38. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
39. Zhang, Z. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 1330–1334. [CrossRef]
40. Jia, D.; Wang, K.; Luo, S.; Liu, T.; Liu, Y. Braft: Recurrent all-pairs field transforms for optical flow based on correlation blocks. *IEEE Signal Process. Lett.* **2021**, *28*, 1575–1579. [CrossRef]
41. Breunig, M.M.; Kriegel, H.; Ng, R.T.; Sander, J. Lof: Identifying density-based local outliers. *Sigmod Rec.* **2000**, *29*, 93–104. [CrossRef]
42. Gené-Mola, J.; Sanz-Cortiella, R.; Rosell-Polo, J.R.; Escolà, A.; Gregorio, E. In-field apple size estimation using photogrammetry-derived 3d point clouds: Comparison of 4 different methods considering fruit occlusions. *Comput. Electron. Agric.* **2021**, *188*, 106343. [CrossRef]
43. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. *Int. J. Comput. Vis.* **2020**, *128*, 336–359. [CrossRef]