Method for the real-time detection of tomato ripeness using a phenotype robot and RP-YolactEdge

Yuanqiao Wang[1,2,3], Wenbo Gou[2,3], Chuanyu Wang[2,3], Jiangchuan Fan[2,4], Weiliang Wen[2,3], Xianju Lu[2,3], Xinyu Guo[3*], Chunjiang Zhao[1,2*]

(1. *College of Information Engineering, Northwest A & F University, Yangling 712100, Shaanxi, China*;
2. *China National Engineering Research Center for Information Technology in Agriculture (NERCITA), Beijing 100097, China*;
3. *Beijing Key Laboratory of Digital Plant, Beijing Research Center for Information Technology in Agriculture, Beijing 100097, China*;
4. *Beijing PAIDE Science and Technology Development Co., Ltd., Beijing 100097, China*)

Abstract: In order to address the challenge of non-destructive detection of tomato fruit ripeness in controlled environments, this study proposed a real-time instance segmentation method based on the edge device. This method combined the principles of phenotype robots and machine vision based on deep learning. A compact and remotely controllable phenotype detection robot was employed to acquire precise data on tomato ripeness. The video data were then processed by using an efficient backbone and the FeatFlowNet structure for feature extraction and analysis of key-frame to non-key-frame mapping from video data. To enhance the diversity of training datasets and the generalization of the model, an innovative approach was chosen by using random enhancement techniques. Besides, the PolyLoss optimization technique was applied to further improve the accuracy of the ripeness multi-class detection tasks. Through validation, the method of this study achieved real-time processing speeds of 90.1 fps (RTX 3070Ti) and 65.5 fps (RTX 2060 S), with an average detection accuracy of 97% compared to manually measured results. This is more accurate and efficient than other instance segmentation models according to actual testing in a greenhouse. Therefore, the results of this research can be deployed in edge devices and provide technical support for unmanned greenhouse monitoring devices or fruit-picking robots in facility environments.

Keywords: instance segmentation, phenotype robot, tomato, greenhouse-based plant phenotyping, ripeness detection
DOI: 10.25165/j.ijabe.20241702.8403

Citation: Wang Y Q, Gou W B, Wang C Y, Fan J C, Wen W L, Lu X J, et al. Method for the real-time detection of tomato ripeness using a phenotype robot and RP-YolactEdge. Int J Agric & Biol Eng, 2024; 17(2): 200–210.

1 Introduction

Tomato, as a significant vegetable crop, relies on an accurate assessment of fruit ripeness, which is crucial for determining tomato quality and optimizing harvest yields[1-5]. This research indicates that non-destructive methods for evaluating fruit ripeness have become a prominent area of study in precision agriculture within controlled environments. The adoption of non-destructive data acquisition techniques offers advantages such as reduced experimental costs and shortened research cycles[6-8]. However, traditional non-destructive approaches for tomato ripeness detection often rely on labor-intensive manual selection and judgment[9-11]. These methods are susceptible to subjective factors and reduced efficiency, resulting in lower detection speed and accuracy. Besides, inefficient operational practices lead to challenges in harvesting tomato fruits in a timely manner based on market demands, causing economic losses for growers and researchers[12].

Presently, common non-destructive detection methods in greenhouse environments can be categorized as static or real-time dynamic, based on the continuity of data acquisition[13]. In terms of data acquisition equipment, they can be classified as fixed devices, such as gantry phenotype platforms, enabling overhead image data acquisition, or single-scale devices like dark boxes or pipeline platforms. Furthermore, these methods can be classified based on the scale of data acquisition, encompassing population-scale detection and individual plant-scale data detection. These methods leverage streamlined procedures to analyze the acquired phenotype information, facilitating subsequent qualitative or quantitative analysis[14-18].

In the current stage, fruit ripeness detection predominantly occurs in static environments, where monitoring devices are deployed in a fixed manner to initially assess the growth of tomato plants in greenhouses. Researchers utilize sensors such as RGB cameras to capture images of the vegetation at specific intervals and subsequently perform identification. For example, in 2021, Sigit Widiyanto presented a different approach in their paper, where they adopted a fixed-point image acquisition strategy to periodically gather information on tomato plants at a population scale[19]. They used an enhanced Mask R-CNN for real-time detection of fruit. Experimental results demonstrated the method's accuracy in dynamically segmenting tomato fruits, providing valuable support for subsequent qualitative analysis such as shape, color, and growth assessment. The study also highlighted the limitations of traditional

Received date: 2023-07-04 **Accepted date:** 2024-02-22
Biographies: Yuanqiao Wang, Under Postgraduate, research interest: machine vision for agricultural robot, Email: yuanqiao_wang@163.com; **Wenbo Gou**, Engineer, research interest: phenotype platform, Email: gouwb@nercita.org.cn; **Chuanyu Wang**, PhD, research interest: plant phenotypes, Email: wangcy@nercita.org.cn; **Jiangchuan Fan**, PhD candidate, research interest: plant phenotypes, Email: fanjc@nercita.org.cn; **Weiliang Wen**, PhD, research interest: plant phenotypes, Email: wenwl@nercita.org.cn; **Xianju Lu**, PhD, research interest: plant phenotypes, Email: luxj@nercita.org.cn.
***Corresponding author: Xinyu Guo**, Researcher, research interest: plant phenotypes. Beijing Key Laboratory of Digital Plant, Beijing Research Center for Information Technology in Agriculture, Beijing 100097, China. Tel: +86-13021083241, Email: guoxy@nercita.org.cn; **Chunjiang Zhao**, Professor, research interest: smart agriculture. China National Engineering Research Center for Information Technology in Agriculture (NERCITA), Beijing 100097, China. Tel: +86-13801308848, Email: zhaocj@nercita.org.cn.

machine learning methods such as K-means and SVM in real-time fruit segmentation tasks, emphasizing the necessity of utilizing deep neural networks for data processing[13,20,21].

With the continuous advancement and modernization of agricultural production, the cultivated areas for crops have progressively expanded and become more intricate. The aforementioned static detection methods evidently fall short of meeting the escalating demands of agricultural automation. Consequently, Jan et al.[22] from the University of Bonn in Germany devised a solution by using a large phenotype robot. This system exhibits the capability to traverse rows in the field and is equipped with an overhead RGB camera for capturing images of sugar beet plants at the seedling stage. Subsequently, the research team employed these images to conduct instance detection of sugar beet plants, enabling differentiation from weeds. Additionally, they utilized a keypoint-based deep learning algorithm to accurately count the leaves of sugar beet plants. Nevertheless, methods like this are processed on the server side, which means that these models need strong computing power and require more power consumption and larger devices. This also limits its deployment on high-performance computing platforms.

Tomato, being the quintessential annual Solanaceae plant, presents a distinctive vine-growing pattern which means the vine needs to be hung up. It poses challenges to most large-scale overhead phenotype platforms. Consequently, our primary focus lies in the development of compact, agile, and side-view-capable phenotyping robots as the optimal solution[5,23-25]. In order to better reduce the size of the robot and reduce the cost of experiment time, we should also choose efficient computer vision models to match smaller edge devices.

In line with the research conducted by Linlu Zu in 2021, they successfully employed a small-scale phenotyping robot to achieve real-time detection of mature green tomatoes within a greenhouse environment[26]. The underlying system employed computer vision-based line-following techniques for efficient path planning, while a surveillance camera mounted on the robot facilitated the acquisition of video stream data on the tomatoes. Subsequently, the obtained data was seamlessly transmitted to a server via a 4G network SIM card slot. Leveraging the power of a Mask R-CNN instance segmentation model, the researchers accomplished accurate detection of the green tomatoes, demonstrating commendable segmentation precision. This solution aptly addressed the demanding requirement of acquiring high-frequency data in compact cultivation environments and showcased the viability of employing instance segmentation models for effective detection and classification. Nonetheless, the transmission of high frame rate video stream data over a 4G network poses inherent challenges, potentially leading to reduced operational speed of the phenotyping robot and consequent compromise in overall efficiency[27].

Furthermore, tomato fruits exhibit multiple distinct ripening stages, each associated with different harvesting times and product values. The accurate detection and classification of these diverse ripening stages place high demands on the precision of computer vision models, necessitating robustness in multi-class classification tasks.

In recent years, the continuous development of computer vision technology has provided novel approaches to address the aforementioned challenges[28-30]. Leveraging real-time instance segmentation algorithms based on lightweight neural networks enables automated detection and swift analysis of agricultural products. Instance segmentation techniques merge object detection and semantic segmentation into a cohesive framework. This integration can be achieved through two-stage methodologies, such as the Mask R-CNN model, which was introduced by He et al.[31] in 2018 using a top-down paradigm. This approach initially employs object detection to identify the regions of instances (bounding boxes) and subsequently performs semantic segmentation within each delineated region, resulting in distinct segmentation outputs for individual instances. While this model exhibits higher parameter complexity and slower inference speed, it ensures relatively precise detection outcomes. Nevertheless, akin to the methods proposed by Zu et al.[26], Widiyanto et al.[19] achieved real-time detection on low-computational edge devices remains a significant challenge.

In order to achieve faster speeds while maintaining overall accuracy, researchers have proposed single-stage methods. Inspired by single-stage object detection methods, such as the Yolact model introduced by Bolya et al.[32], these methods aimed to improve instance segmentation tasks. Specifically, the proposed approach utilizes two parallel subtasks: 1) generating a set of prototype masks; 2) predicting mask coefficients for each instance. The instance masks are then generated by linearly combining the prototypes with the mask coefficients. Additionally, a Fast NMS method is employed to consolidate bounding boxes, reducing training and inference time while ensuring high segmentation and detection accuracy.

In practical application scenarios, small-sized and low-power (low computational capability) edge devices emerge as the optimal choice for phenotype detection robots and other agricultural robots. The data processing speed of the aforementioned models is evidently insufficient to achieve real-time execution at high frame rates on edge devices. Therefore, this study takes note of the more efficient YolactEdge proposed by Liu et al.[33]. This model reduced feature redundancy by distinguishing between keyframes and non-keyframes during prediction. Additionally, the model was optimized using TensorRT, enabling it to achieve a performance of up to 30.8 fps on Jetson AGX Xavier (and 172.7 fps on RTX 2080 Ti).

Therefore, in order to achieve real-time, non-destructive, high-precision, unmanned, high-throughput, and multi-class ripeness detection of tomatoes, an edge device-based approach was proposed using real-time instance segmentation models, which the method can be deployed on phenotype robots. This method harnesses the flexible and intelligent nature of phenotype detection robots to dynamically acquire continuous and comprehensive tomato images from multiple perspectives. It enables efficient and non-invasive ripeness detection of tomato fruits, which belong to the Solanaceae family. Specifically, a phenotype robot equipped with a remote control module, optical sensors, and edge devices were utilized[34]. Tomato images were acquired in constrained environments using remote control techniques. Subsequently, RP-YolactEdge (YolactEdge with Random Enhancement & PolyLoss) was employed to perform automated and accurate classification and counting of tomato ripeness[35,36]. This study aimed to explore the application of phenotype robots in controlled environments for unmanned and non-invasive fruit ripeness detection, providing valuable insights for future research on phenotype robots and other agricultural robots in controlled environments[37-40].

2 Materials and methods

2.1 Experimental platform

To achieve non-destructive detection of tomato fruit ripeness in controlled environments, this experiment considers various aspects including phenotype platforms, control methods, data acquisition

and preprocessing, model development and deployment, and practical testing (Figure 1, including the selection of phenotype devices and control methods, data acquisition and preprocessing, model deployment, and testing). In the preparation phase, first, the phenotype devices, sensors, power supply equipment, and network communication devices deployed were used. Next, tomato plant images were captured from the hanging vines according to the plan, and the data were selected and processed accordingly. The processed data were then used to train a deep learning model, and the model's details were dynamically adjusted based on the training results to improve accuracy and prediction efficiency. Finally, the model was deployed on edge devices, and field testing and validation were conducted.

The small-scale plant phenotyping robot utilized for real-time detection of tomato fruit ripeness incorporates the Autolabor PM1 robot (manufactured by Autolabor, Beijing, China) as its underlying mobile platform. This device employs a three-wheel differential drive system with dimensions measuring 750×520×1150 mm^3 and weighing approximately 45 kg. It offers a ground clearance of about 50 mm and provides three distinct control modes: handle control, SLAM mapping and navigation, and remote control. The equipment configuration includes an upper computer, a front-to-back single-line LiDAR, a 4G/5G signal transmitter positioned at the top, and a monocular camera mounted on each side.

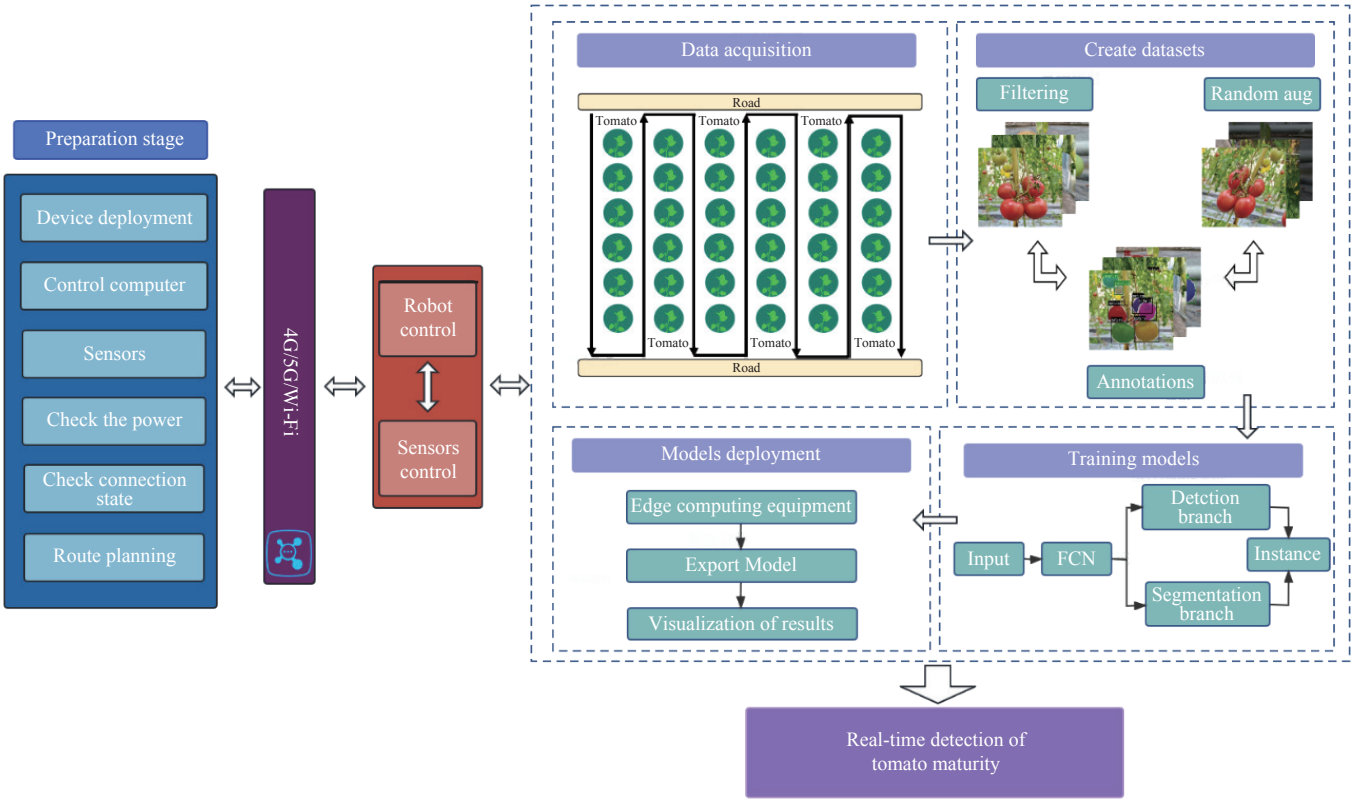


Figure 1 Overall flowchart of the experiment

Taking into consideration the requirements of the experimental site environment, autonomous mobility, obstacle avoidance, and data acquisition, the remote control mode for operation was opted for. During usage, connections with the robot's control equipment were established through 5G signals. Therefore, the client can receive real-time road condition data transmitted by the camera, enabling remote control of the device. The remote client utilizes a simulated driving system, which employs split screens or multiple displays to provide a 360° perception of the robot's operating environment. Moreover, the bottom-mounted LiDAR sensor was equipped with auxiliary obstacle avoidance capabilities, ensuring safe operations by preventing incidents such as chassis collisions.

To capture side-view images of tomato fruits, the phenotype robot was equipped with a sensor box (on the left) and a control device box (on the right), serving as the data acquisition platform. The sensor box features multiple fixed slots for sensors, allowing for the integration of various devices such as the depth camera sensor used in this experiment, as well as industrial cameras (RGB sensors), multispectral sensors, thermal infrared sensors, and LiDAR sensors. These sensors were integrated within the right box, which houses an edge computing device (industrial computer with RTX 2060S) and was connected to power adapters, enabling real-time data acquisition and processing[25].

For data acquisition, the Microsoft Azure Kinect DK depth camera (located in Redmond, Washington, USA) was employed in this experiment. With dimensions of approximately 12.5 cm in length and 10.3 cm in width and a weight of 440 g, this sensor operates at a maximum power consumption of only 5.9 W while achieving a frame rate of 30 fps for RGB image acquisition at a resolution of 5 million pixels. RGB channel data from the depth camera were utilized as the training and validation datasets for tomato fruit images, facilitating dynamic detection of fruit ripeness using real-time video stream data (Figure 2).

2.2 Image acquisition and processing

The experiment was conducted from November 5, 2022, to December 2, 2022, at the Beijing Academy of Agriculture and Forestry Sciences greenhouse, located at 39°56′N, 116°16′E. The tomato cultivation area consisted of two plots measuring 10 m× 40 m. The tomatoes were grown using soil cultivation techniques, and the plants were trained using string trellises. The selected tomato variety was "Fen Yan No. 1" (GPD Tomato (2017) 110007), and standard irrigation and fertilization practices were followed.

Note: a and b: The front and rear views of the robot's structure, respectively; c and d: The sensor gimbal on the left side and the control box on the right side of the robot; e: The camera and signal transmission device used for remote control; f: The bottom-mounted LiDAR sensor employed for obstacle avoidance; g: The actual workflow of remote control operations.

Figure 2 Structure and operational images of the employed phenotyping robot

To collect images of the tomato fruits in the horizontal direction, the robot's sensors, industrial computer, and mobile power supply were installed before the experiment. The robot was placed in the experimental area (field roads) and connected to a power source. After establishing the connection with the remote control computer, the robot was operated remotely at a constant speed of 0.5 m/s along the tomato experimental field. The camera captured images at a rate of 1 photo per 2 s to ensure high-quality data acquisition.

An image acquisition plan was developed based on the actual conditions of the experimental field (Figure 1): the robot was remotely driven at a constant speed of 0.5 m/s (equivalent to a travel speed of 2 km/h) between the rows of tomato plants. A total of over 3700 images were captured during the experiment.

According to research, tomato fruits exhibit four stages of maturity: green stage, turning stage, mature stage, and fully ripe stage. The turning stage and firm stage are considered the semi-ripe stage, characterized by differences in internal biochemical components while displaying minimal external variations in terms of shape, color, and texture. The specific characteristics of each stage are as follows:

1) Green stage: The fruit has reached its full size, but the skin remains entirely green, and the flesh maintains firmness;

2) Turning stage: The top of the fruit begins transitioning from green to yellow-white, accompanied by a softening of the flesh and an increase in sugar content. This stage is ideal for harvesting when the fruit requires long-distance transportation or storage;

3) Mature stage: Approximately three-fourths of the fruit's surface turns red or yellow, indicating the highest nutritional value. This stage is optimal for immediate consumption or when the fruit needs to be transported over shorter distances;

4) Fully ripe stage: The entire surface of the fruit turns red, and the flesh reaches a soft consistency while attaining its maximum sugar content. This stage is also suitable for harvesting when the fruit requires short-distance transportation or immediate use.

Therefore, the tomato images were first annotated using three different labels: "Mature", "Semimature", and "GreenRipening". Next, the normalization and cropping operations were performed on the images used for training. In this experiment, 332 images were selected as the original training set and 32 images as the original validation set. Additionally, 10 segments of 160 frames of video data were obtained for model testing and accuracy validation.

To enhance the classification accuracy and generalization of the model, random data augmentation techniques were employed to increase the diversity and quantity of the training data. Specifically, the following data augmentation operations were applied to the original image data:

1) Random perspective: The images were randomly rotated along the *X*-axis, *Y*-axis, or *Z*-axis to introduce diversity into the dataset. Additionally, random translation operations were performed on the images. This approach makes the dataset more representative of real-world scenarios where fruit ripeness detection occurs from different perspectives;

2) Salt and pepper noise: Salt and pepper noise (impulse noise) was randomly added to the images.

Salt and pepper noise refers to the presence of random black and white pixels scattered throughout an image, resulting from factors such as image sensor artifacts, transmission channel distortions, or decoding and processing errors. In Equation (1), N represents the modified color values of the three channels after pixel manipulation. This method emulates the interference caused by sensor device anomalies, ultimately contributing to enhancing the model's resilience and robustness in handling noisy input data.

$$N = \begin{cases} 0, & \text{pepper} \\ 255, & \text{salt} \end{cases} \tag{1}$$

1) Gaussian noise: Gaussian noise was added to images. Gaussian noise, denoted as $f(x)$, refers to a type of noise characterized by a probability density function that follows a Gaussian distribution (also known as a normal distribution). The Gaussian distribution is symmetric about $x=\mu$, with an amplitude of $\sqrt{2\pi}\sigma$ and e represent the natural constant. By applying this statistical property, each pixel in the image is modified to introduce noise. This method effectively simulates the interference caused by various lighting conditions and environmental factors, thus enhancing the model's generalization capabilities.

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2}$$

2) HSV augment: Adjustments were performed to the brightness and contrast of the images to simulate variations in data obtained under different times and lighting conditions, thereby

enhancing the overall accuracy of the method.

In order to enhance the variability of the samples and optimize the training effectiveness of the model, a single random method from the aforementioned techniques was applied to each individual image during the preprocessing stage. Additionally, the parameters used for the processing were randomly determined. This approach maximizes the diversity of the samples and improves the efficacy of the model training. Following the data augmentation process, the training set was expanded to include 664 images (approximately containing 2000 instances of tomato fruit), while the validation set comprised 64 images (approximately containing 200 instances of tomato fruit) (Figure 3). To test the limited frame rate of the model processing, the video frame rate was also adjusted to 160 fps, and the experimental results are illustrated in the next section.

Note: First column: The original images after screening; Second column: Randomly rotated images; Third column: Images with randomly removed pixels; Fourth column: Images subjected to random noise processing; Fifth column: Images with randomly adjusted brightness and contrast.

Figure 3 Experimental image data of tomatoes

2.3 RP-YolactEdge

RP-YolactEdge (YolactEdge with Random Enhancement & PolyLoss) is an advanced single-stage real-time instance segmentation model that builds upon the foundations of YolactEdge. This model introduces several targeted improvements to enhance its performance (Figure 4). It utilizes a lightweight backbone network along with more efficient neck and head structures, resulting in significantly improved prediction speed. Additionally, the proposed method incorporates the PolyLoss optimization technique to refine the cross-entropy loss function, reducing the overall parameter count and enhancing the accuracy of maturity classification. To further enhance its real-time capabilities, the model was accelerated by using TensorRT. The real-time detection results were presented in video format and can be accessed on the website at https://youtu.be/mY31sPLOreI.

2.3.1 Backbone

In RP-YolactEdge, MobileNetV2 was chosen as the backbone network for training. MobileNetV2 retains the depthwise separable convolution units from MobileNetV1, which decomposes the standard convolution operation into two smaller operations: depthwise convolution and pointwise convolution. The depthwise convolution performs lightweight convolutions on each input channel to extract spatial features, while the pointwise convolution integrates these features and generates the output by performing a set of 1×1 convolutions on each output channel[41].

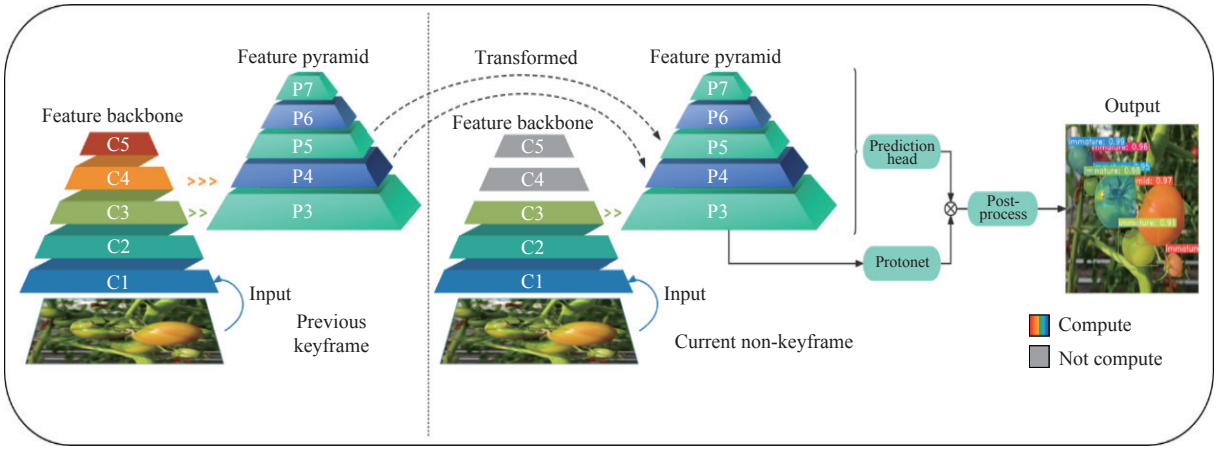


Figure 4 RP-YolactEdge reduces network computations by transforming a subset of features from keyframes (left) to non-keyframes (right)

Depthwise separable convolution offers several advantages over standard convolution. Firstly, depthwise separable convolution requires fewer computations than standard convolution since it performs fewer operations. This enables faster training and deployment of networks using depthwise separable convolution. Secondly, it exhibits better generalization to unseen data because the convolutions performed on each input channel are lightweight. This allows it to better handle variations and noise in the data. Moreover, depthwise separable convolution requires less computation than standard convolution because it requires fewer operations to perform. This means that deep separable convolutional networks can be trained and deployed in a faster time frame.

Furthermore, MobileNetV2 incorporates the novel Inverted Residuals and Linear Bottlenecks units (Figure 5). These units differ from traditional residual network units in that they have reduced input and output dimensions. They achieve this by employing linear convolution to expand the dimensions, followed by depth-wise convolution for feature extraction (Equation (3)). This design significantly reduces the number of parameters in the entire model.

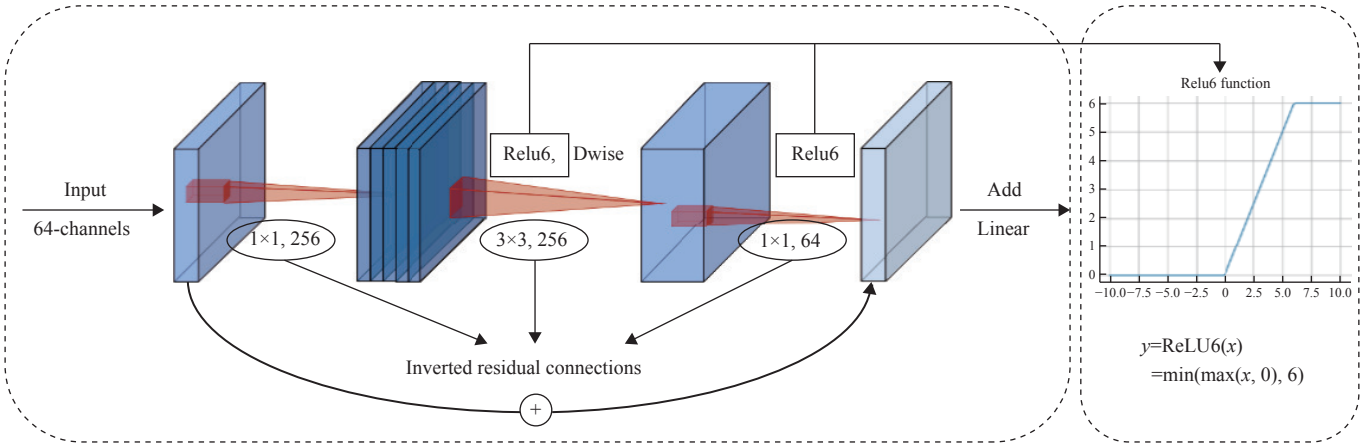


Figure 5 Schematic diagram of the linear neck reverse residual in MobileNetV2

$$\text{MobileNetV2} \xrightarrow{\text{ReLU6}} PW_{1\times1} \xrightarrow{\text{ReLU6}} DW_{3\times3} \xrightarrow{\text{ReLU6}} PW_{1\times1} \xrightarrow{\text{Linear}} FPN \quad (3)$$

where, ReLU6 represents the nonlinear activation function, PW and DW represent convolution kernels of different sizes.

The term “linear bottleneck” refers to the reduction of channel dimensionality by mapping a high-dimensional space to a lower-dimensional space. It is evident that a reverse residual layer with a linear bottleneck is highly suitable for mobile designs because it allows for a significant reduction in memory usage during the inference process by partially decoupling large intermediate tensors. This methodology lessens the requirement for accessing the primary memory in numerous embedded hardware configurations and introduces a compact yet highly effective software-managed cache. Consequently, it aligns well with the constraints of edge devices, which typically operate on platforms with limited computational capabilities.

2.3.2 Neck and head

The neck component of RP-YolactEdge incorporates the Feature Pyramid Network (FPN) structure to effectively handle feature information at various scales and enhance the accuracy of object detection. The feature pyramid mechanism facilitates the fusion of low-resolution, high-semantic features with high-resolution, low-semantic features, resulting in improved detection precision across multiple scales. This capability mitigates detection errors arising from inconsistent camera distances, leading to enhanced multi-class classification accuracy, and holds significant practical implications.

To further optimize model execution speed, a multi-level feature pyramid structure (P3-P7) was adopted for processing non-keyframes in video streams. This approach maximizes feature reuse efficiency and ensures consistent execution accuracy throughout the video stream. Leveraging the interplay between adjacent keyframes and non-keyframes, the deformed feature pyramid connections were exploited, specifically utilizing the P5 and P4 layers from keyframes for downsampling predictions. This strategy minimizes iterations in the backbone network for non-keyframes, thereby accelerating the prediction process.

This approach yields superior results in practical testing scenarios, such as when a robot moves at a constant speed in a tomato field, where consecutive frames in the captured video stream exhibit high similarity. The aforementioned processing technique ensures significant improvements in operational efficiency while maintaining segmentation and detection accuracy (Figure 5).

Instead of using a linear network to process the raw RGB frames, a set of semantically rich features computed by the model’s backbone network was leveraged for repeated utilization. Specifically, the FeatFlowNet structure is employed to input the features already extracted from the backbone network, requiring fewer convolutional layers. This strategy reduces the parameter complexity resulting from iterative connections in deep networks and consequently decreases the overall computational time of the model (Figure 6).

Specifically, the FeatFlowNet structure estimates the flow map $\boldsymbol{M}(I^k, I^n)$ between the preceding keyframe I^k and the current non-keyframe I^n in the video stream. It then performs an inverse mapping transformation of features from I^k to I^n: by projecting each pixel in I^n using $X+\delta X$ onto I^k, where $\delta X=M_X(I^k, I^n)$ (Equation (4)). The pixel values $F(x)$ are then computed using the bilinear interpolation equation as follows:

$$F^{k\to n}(x) = \sum_{\mu}^{\theta(\mu, x+\delta x)} F^k(x) \quad (4)$$

where, μ is a fixed value; θ represents the weights of different point positions in bilinear interpolation; k represents the k-th keyframe around the flow data I; n represents the k-th non-keyframe around the flow data I.

After the parameter splitting process, the parameters will be decoupled through the prediction Head and ProtoNet, ultimately achieving the output of instance segmentation results.

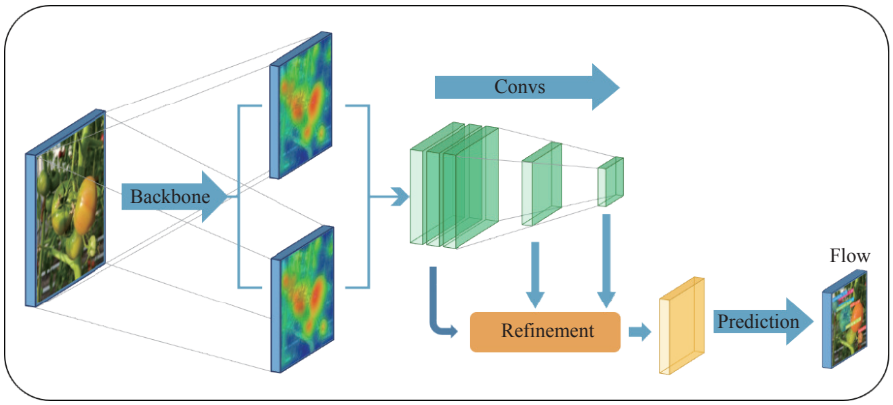


Figure 6 FeatFlowNet in video stream estimation method

2.3.3 Loss function

The main objective of maturity detection is to accurately distinguish between different ripeness stages of fruits and annotate the results on the video (image). Therefore, the accuracy in multi-class classification plays a crucial role in evaluating this approach. However, the dataset used in this experiment captures various real-world scenarios encountered during tomato data acquisition. For instance, there is a substantial variation in the number of tomato fruit images at different ripeness stages obtained within a short time span. Furthermore, the progression of ripeness in tomatoes can vary under differing environmental conditions. Ignoring the disparities in the quantity of samples from different maturity stages could markedly impact the detection outcomes. On the other hand, consciously equalizing the number of samples from each category could either diminish the dataset's size or necessitate additional time for data collection, thereby introducing further complications.

Furthermore, the objective of this task is to achieve multi-stage classification detection, which differs from single detection or binary classification tasks. It often requires higher classification accuracy. Therefore, the PolyLoss loss function was used in the classification task[42]. PolyLoss treats the loss function as a linear combination of polynomial functions and designs it to optimize the cross-entropy function using gradient descent (Equation (5)). Specifically, within the PolyLoss framework, the polynomial terms in the gradient expansion capture different sensitivities to P_t (Equation (6)). The first gradient term is 1, providing a constant gradient independent of the value of P_t. In contrast, when $j \gg 1$ and P_t approaches 1, the j^{th} term is strongly suppressed. Thus, the coefficient $\frac{1}{j}$ precisely offsets the j^{th} power of the polynomial base, resulting in the gradient of the cross-entropy loss being the sum of the polynomial $(1-P_t)^j$.

$$L_{CE} = -\log(P_t) = \sum_{j=1}^{\infty}(1-P_t)^j = (1-P_t) + \frac{1}{2}(1-P_t)^2 + \cdots \quad (5)$$

$$-\frac{dL_{CE}}{dP_t} = \sum_{j=1}^{\infty}(1-P_t)^{j-1} = 1 + (1-P_t) + (1-P_t)^2 + \cdots \quad (6)$$

Through this improvement, the training errors caused by the differences were successfully addressed in sample quantities, and further, the loss rate was reduced, resulting in excellent multi-classification accuracy. This improvement has also been proven to reduce the false positive rate in testing.

The model was evaluated using mean Average Precision (mAP) (Equation (7)).

$$\text{mAP} = \frac{\sum_{i=1}^{k}(AP_i)}{k} \quad (7)$$

where, k represents the total number of evaluations; AP_i represents the accuracy at different recall conditions (Equation (8)), as follows:

$$\text{AP} = \sum_{i=1}^{n-1}(r_{i+1} - r_i)P_{\text{inter}}(r_i + 1) \quad (8)$$

where, r_i corresponds to the recall value at the first interpolation point, which is sorted in ascending order of Precision as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (9)$$

where, TP represents the count of true positives and FP represents the count of true positives false positives

2.4 Tomato counting

To evaluate the model's classification results in real-time detection, the intersection of the detection box and a vertical counting line to track the total number of fruit instances at different stages (Equation (9)) were utilized. It was determined whether the centrally positioned vertical line (L_c) intersects with the detection box, satisfying specific conditions based on the width of the image or video frame (W_p). To avoid duplicate counting of the same fruit instance across frames, the centroid of the detection box was replied to as the counting criterion. If the centroid intersects with the line L_c, the count for the corresponding maturity stage is incremented by one (Equation (10)).

$$L_{cx} = \frac{1}{2}W_p \quad (10)$$

$$\begin{cases} M_x = \frac{1}{2}(U_{x1} + L_{x1}) \\ M_x = L_{cx} \end{cases} (M_x, U_{x1}, L_{x1}, L_{cx}) \in (0, W_p) \quad (11)$$

The coordinates of the top-left and bottom-right corners of the detection box are denoted as (U_x, U_y) and (L_x, L_y), respectively. The subscripts 1 and 2 represent the previous frame and the current frame's detection boxes, while the centroid coordinates are represented as (M_x, M_y). Based on experimental comparisons, this method effectively reduces duplicate counting during dynamic detection. The real-time counting results can be viewed on the webpage https://youtu.be/mY31sPLOreI.

In summary, the RP-YolactEdge model was utilized to incorporate several optimizations and enhancements compared to Yolact. It offers higher real-time performance and accuracy, making it suitable for real-time object detection and segmentation tasks such as the tomato fruit ripeness detection in this study.

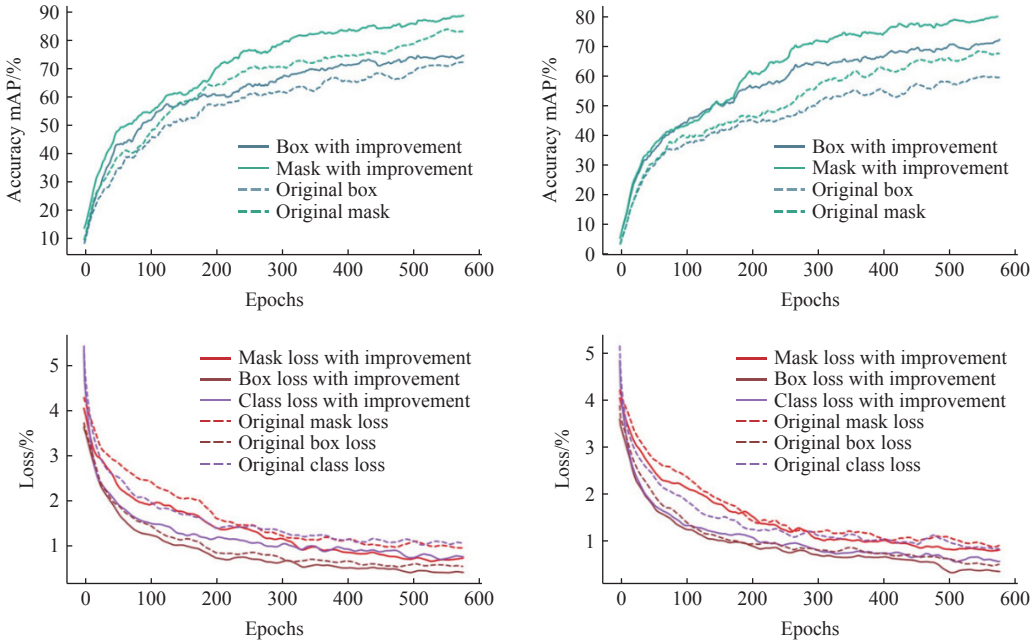
3 Results and discussion

3.1 Ripeness detection accuracy

In this section, the RP-YolactEdge model was trained and tested on the acquired dataset. The accuracy, speed, and ripeness detection performance of the instance segmentation method were primarily analyzed. Compared the proposed method in this study with state-of-the-art real-time instance segmentation methods and conducted ablation studies to dissect our design choices and modules[33].

Transfer learning was implemented using a pre-trained model. Through computational validation, mask mAP of 89.1% and 80.4% were attained, and bounding box mAP (bbox mAP) of 72.2% and 70.1%, utilizing ResNet-101 and MobileNetV2 as backbone networks, respectively (Figure 7). Additionally, the classification loss, box loss, and segmentation loss of different methods were examined. Our RP-YolactEdge model exhibited lower loss rates compared to other approaches, particularly in terms of classification loss, where the proposed model significantly outperformed others. This further demonstrates the necessity of employing the PolyLoss loss function for multi-class ripeness detection of tomatoes.

Tests were conducted on the model using 64 images, which included a total of 198 tomato instances. The results were compared against manual annotations (R-101-FPN). Among the instances in the "GreenRipening" stage (Table 1), 102 instances were correctly detected, achieving an accuracy of 98.1%. For the instances in the "Semimature" stage, 43 instances were accurately detected, resulting in an accuracy of 93.5%. In the "Mature" stage, 47 instances were correctly detected, yielding an accuracy of 97.9%. The average accuracy across all stages was 97.0%. It is important to note that false positives and false negatives were not considered in the model's correct detections. Additionally, the segmented results were dynamically saved to the local storage.



Note: The solid line represents the proposed method in this study, and the dashed line represents the original YolactEdge model.

Figure 7 Comparison of Precision and Loss Rate between RP-YolactEdge Model and YolactEdge Model

Table 1 Accuracy comparison between improved YolactEdge model and manual detection

Maturation period	Legend	Manual detection	Model detection	Accuracy
GreenRipening		104	102	98.1%
Semimature		46	43	93.5%
Mature		48	47	97.9%
Total	--	198	192	97.0%

3.2 Execution efficiency

Tests were conducted using video stream data to evaluate the performance and efficiency of the improved model in this study. With this approach, the real-time dynamic detection of tomato fruit ripeness was achieved. The results show that this method outperforms the original YolactEdge and Yolact models in terms of both detection and segmentation accuracy. The slight shaking of the car body caused by the unevenness of the road has a limited effect on the test results.

From the perspective of runtime and execution efficiency, the method of this study achieved execution speeds of 80.9 fps (ResNet-101) and 90.1 fps (MobileNetV2) on Nvidia 3070ti devices. Even without using TensorRT for acceleration, the model still achieves prediction speeds of 29.2 FPS (ResNet-101) and 54.0 FPS (MobileNetV2) (Table 2). These speeds significantly outperform the Mask R-CNN model or other commonly used instance segmentation models.

Table 2 Comparison of detection performance between RP-YolactEdge and other methods

Method	Backbone	RTX/fps	Params/M
RP-YolactEdge	R-101-FPN	80.9√	49.6M
RP-YolactEdge	R-50-FPN	82.3√	31.7M
RP-YolactEdge	MobileNetV2-FPN	90.1√	8.5M
Yolact	R-101-FPN	30.2	55.1M
Yolact	R-50-FPN	35.6	47.5M
Mask R-CNN	R-101-FPN	12.1	62.9M
Mask R-CNN	R-50-FPN	14.7	43.3M

Note: √ denotes suitability for real-time detection tasks.

Tests were also conducted on small industrial control computers such as RTX2060 Super and edge devices like Jetson AGX Xavier. With MobileNetV2 as the backbone network and TensorRT for inference acceleration, the respective execution speeds were 65.5 fps and 28.7 fps. Other key hardware models used in the tests are shown in the table (Table 3). This demonstrates that our method can function effectively even with lower computational power and can be seamlessly integrated with higher-resolution sensors, enabling precise dynamic detection of tomato fruit ripeness.

Table 3 The hardware and software used for training and testing

Hardware	Training	Testing
CPU	12th Gen Intel® Core i7-12700	11th Gen Intel® Core i7-1165G7
RAM	16 GB	8 GB
GPU	Nvidia 3070ti	NVIDIA GeForce RTX 2060
Operating system	Windows 11	Ubuntu18.04
Deeplearning framework	Pytorch	Pytorch

3.3 Ablation Experiment and Discussion

Comprehensive generalization validation and ablation studies were conducted to evaluate the robustness and efficacy of our proposed method.

Initially, compared the efficacy of using the detection method alone versus the instance segmentation method alone for tomato fruit ripeness detection. The results indicated that relying solely on the detection method failed to accurately detect and count fruits in scenarios involving occlusion or closely connected fruits. This limitation arises due to the detection method's emphasis on learning features within the entire bounding box region, whereas the single-stage instance segmentation method incorporates both instance boundary features and bounding box positioning.

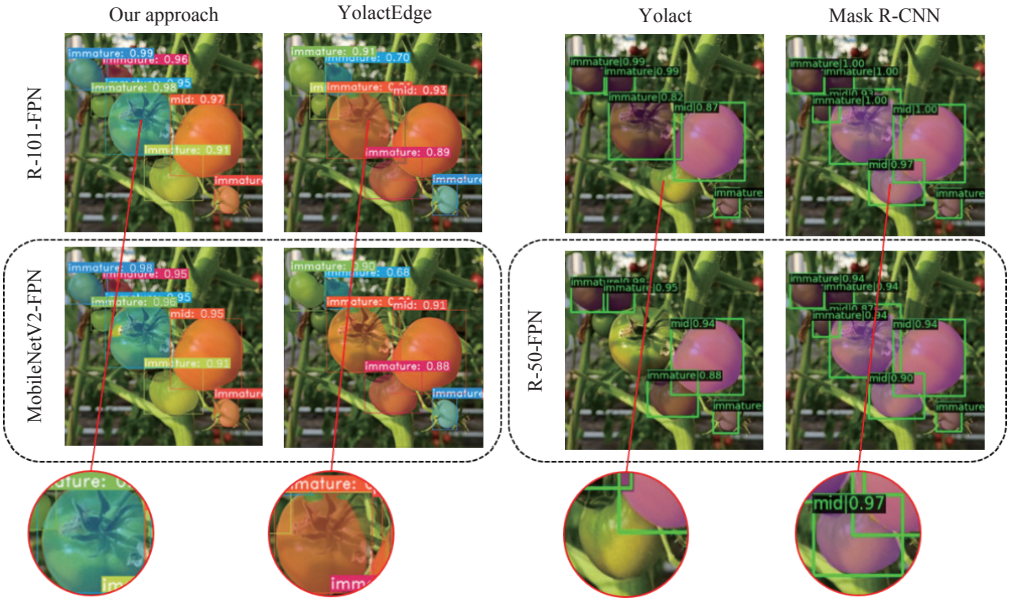
Additionally, we conducted a controlled experiment to evaluate the impact of utilizing PolyLoss versus cross-entropy loss when the dataset remains consistent. We also investigated the necessity of employing random enhancement to address class imbalance in the samples, ensuring the models remained consistent (Table 4). The outcomes demonstrated that our improved method outperformed the original YolactEdge model, exhibiting enhanced detection accuracy and reduced classification loss.

Table 4 Ablation experiments of RP-YolactEdge

Backbone	Classification loss function	Box mAP	Mask mAP	Class Loss
R-101-FPN*	PolyLoss	76.85	90.17	0.54
MobileNetV2-FPN*	PolyLoss	73.11	80.08	0.66
R-101-FPN	PolyLoss	74.36	83.72	0.61
MobileNetV2-FPN	PolyLoss	71.40	78.79	0.74
R-101-FPN*	Cross-entropy	75.84	88.23	0.58
MobileNetV2-FPN*	Cross-entropy	72.51	78.44	0.68
R-101-FPN	Cross-entropy	71.8	81.9	0.84
MobileNetV2-FPN	Cross-entropy	67.54	76.08	0.93

Note: * indicates the use of random enhancement for dataset augmentation.

A comparative analysis of different instance segmentation models was also conducted with the approach in terms of detection performance. Through the comparison, it was observed that this method outperforms the original YolactEdge and YOLACT models in both segmentation and detection tasks (as shown in the first, second, and third columns of Figure 8). While the Mask R-CNN model exhibits slightly better segmentation performance than our method, its classification accuracy is relatively lower for occluded instances (as shown in the first and fourth columns of Figure 8).



Note: Column from left to right: the first represents our method, the second shows the results of the YolactEdge without optimization, the third displays the results of the Yolact, and the fourth presents the results of Mask R-CNN.

Figure 8 Comparison of the proposedmethod in this study and other approaches in practical working scenarios

4 Conclusions

This study focuses on the real-time detection of tomato fruit ripeness in controlled environments using a plant phenotype robot and an improved instance segmentation model. A remotely operated phenotype robot was utilized as the detection device and deployed the RP-YolactEdge instance segmentation model on the device. In the experimental process, the lightweight MobileNetV2 was employed as the backbone network and incorporated the FeatFlowNet structure to process video stream data from adjacent

keyframes and non-keyframes. This approach not only ensured high detection accuracy but also significantly improved data prediction efficiency. Random data enhancement techniques were applied to expand the dataset, enriching the samples with diverse real-world scenarios. Furthermore, the PolyLoss classification loss function was employed to enhance classification accuracy, addressing the significant variation in sample quantities across different classes.

The experimental results demonstrate that the proposed model exhibits high performance in terms of real-time capability and accuracy. It verifies the effectiveness of utilizing a phenotyping robot for real-time tomato fruit ripeness detection in controlled environments. In future work, it is planned to employ a gimbal with greater degrees of freedom to mount a camera, combined with models that offer improved classification and prediction performance. This will enable the solution to achieve a wider field of view and accomplish dynamic detection of plant fruit from multi-scales.

Acknowledgements

This work was funded by the National Key R&D Program (Grant No. 2022YFD2002305), Beijing Nova Program (Grant No. Z211100002121065; 20220484202), Collaborative Innovation Center of Beijing Academy of Agricultural and Forestry Sciences (Grant No. KJCX201917).

[References]

[1] Begum N, Hazarika M K. Maturity detection of tomatoes using transfer learning. Measurement: Food, 2022; 7: 100038.

[2] Chen G, Muriki H, Pradalier C, Chen Y, Dellaert F. A hybrid cable-driven robot for non-destructive leafy plant monitoring and mass estimation using structure from motion. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), 2023; pp.11809–11816.

[3] Chen S M, Xiong J T, Jiao J M, Xie Z M, Huo Z W, Hu W X. Citrus fruits maturity detection in natural environments based on convolutional neural networks and visual saliency map. Precision Agriculture, 2022; 23: 1515–1531.

[4] Huang Y P, Si W, Chen K J, Sun Y. Assessment of tomato maturity in different layers by spatially resolved spectroscopy. Sensors, 2020; 20: 7229.

[5] Zheng T X, Jiang M Z, Li Y F, Feng M C. Research on tomato detection in natural environment based on RC-YOLOv4. Computers and Electronics in Agriculture, 2022; 198: 107029.

[6] Atefi A, Ge Y, Pitla S, Schnable J. Robotic technologies for high-throughput plant phenotyping: Contemporary reviews and future perspectives. Front Plant Sci, 2021; 12: 611940.

[7] Dong M, Yu H Y, Zhang L, Wu M Z, Sun Z P, Zeng D Q, et al. Measurement method of plant phenotypic parameters based on image deep learning. Wireless Communications and Mobile Computing, 2022; 2022: 7664045.

[8] Kolhar S, Jagtap J. Convolutional neural network based encoder-decoder architectures for semantic segmentation of plants. Ecological Informatics, 2021; 64: 101373.

[9] Rahul M S P, Rajesh M. Image processing based automatic plant disease detection and stem cutting robot. In: 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), Tirunelveli: IEEE, 2020; pp.889–894.

[10] Numsong A, Posom J, Chuan-Udom S. Artificial neural network-based repair and maintenance cost estimation model for rice combine harvesters. Int J Agric & Biol Eng, 2023; 16(2): 38–47.

[11] Feng X B, He P J, Zhang H X, Yin W Q, Qian Y, Cao P, et al. Rice seeds identification based on back propagation neural network model. Int J Agric & Biol Eng, 2019; 12(6): 122–128.

[12] Lu W, Zeng M J, Qin H H. Intelligent navigation algorithm of plant phenotype detection robot based on dynamic credibility evaluation. Int J Agric & Biol Eng, 2021; 14(6): 195–206.

[13] Xiao D Y, Liang G, Liu C L, Huang Y X. Phenotype-based robotic screening platform for leafy plant breeding. IFAC-PapersOnLine, 2016; 49(16): 237–241.

[14] Li Y T, He L Y, Jia J M, Chen J N, Lyu J, Wu C Y. High-efficiency tea shoot detection method via a compressed deep learning model. Int J Agric & Biol Eng, 2022; 15(3): 159–166.

[15] Lu Y, Chen X Y, Wu Z X, Yu J Z, Wen L. A novel robotic visual perception framework for underwater operation. Frontiers of Information Technology & Electronic Engineering, 2022; 23(11): 1602–1619.

[16] Wang Y Q, Fan J C, Yu S, Cai S Z, Guo X Y, Zhao C J. Research advance in phenotype detection robots for agriculture and forestry. Int J Agric & Biol Eng, 2023; 16(1): 14–25.

[17] Yu S, Fan J C, Lu X J, Wen W L, Shao S, Guo X Y, et al. Hyperspectral technique combined with deep learning algorithm for prediction of phenotyping traits in lettuce. Frontiers in Plant Science, 2022; 13: 927832.

[18] Hu H M, Kaizu Y, Zhang H D, Xu Y W, Imou K, Li M, et al. Recognition and localization of strawberries from 3D binocular cameras for a strawberry picking robot using coupled YOLO/Mask R-CNN. Int J Agric & Biol Eng, 2022; 15(6): 175–179.

[19] Widiyanto S, Nugroho D P, Daryanto A, Yunus M, Tri D. Monitoring the growth of tomatoes in real time with deep learning-based image segmentation. International Journal of Advanced Computer Science and Applications, 2021; 12(12): 0121247.

[20] Zhang Y, Tian Z H, Ma W Q, Zhang M, Yang L L. Hyperspectral detection of walnut protein contents based on improved whale optimized algorithm. Int J Agric & Biol Eng, 2022; 15(6): 235–241.

[21] Liu W, Zou S S, Xu X L, Gu Q Y, He W Z, Huang J, et al. Development of UAV-based shot seeding device for rice planting. Int J Agric & Biol Eng, 2022; 15(6): 1–7.

[22] Weyler J, Milioto A, Falck T, Behley J, Stachniss C. Joint plant instance detection and leaf count estimation for in-field plant phenotyping. IEEE Robot and Automation Letters, 2021; 6(2): 3599–3606.

[23] Hosoi F, Nakabayashi K, Omasa K. 3-D modeling of tomato canopies using a high-resolution portable scanning Lidar for extracting structural information. Sensors, 2011; 11(2): 2166–2174.

[24] Yang L L, Tian W Z, Zhai W X, Wang X X, Chen Z B, Wen L, et al. Behavior recognition and fuel consumption prediction of tractor sowing operations using smartphone. Int J Agric & Biol Eng, 2022; 15(4): 154–162.

[25] Li H, Issaka Z, Jiang Y, Tang P, Chen C. Overview of emerging technologies in sprinkler irrigation to optimize crop production. Int J Agric & Biol Eng, 2019; 12(3): 1–9.

[26] Zu L L, Zhao Y P, Liu J Q, Su F, Zhang Y, Liu P Z. Detection and segmentation of mature green tomatoes based on mask R-CNN with automatic image acquisition approach. Sensors, 2021; 21(23): 7842.

[27] Fan J C, Zhang Y, Wen W L, Gu S H, Lu X J, Guo X Y. The future of Internet of Things in agriculture: Plant high-throughput phenotypic platform. Journal of Cleaner Production, 2021; 280: 123651.

[28] Jin Y, Liu J, Xu Z, Yuan S, Li P, Wang J, et al. Development status and trend of agricultural robot technology. Int J Agric & Biol Eng, 2021; 14(4): 1–19.

[29] Xiang L R, Nolan T M, Bao Y, Elmore M, Tuel T, Gai J Y, et al. Robotic assay for drought (RoAD): An automated phenotyping system for brassinosteroid and drought responses. Plant Journal, 2021; 107: 1837–1853.

[30] Yang C Z. Plant leaf recognition by integrating shape and texture features. Pattern Recognition, 2021; 112: 107809.

[31] He K M, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. 2017 IEEE international Conference on Computer Vision (ICCV), Venice: IEEE, 2017; pp.2980–2988.

[32] Bolya D, Zhou C, Xiao F Y, Lee Y J. YOLACT: Real-time instance segmentation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul: IEEE, 2019; pp.9156–9165.

[33] Liu H, Rivera Soto R A, Xiao F, Jae Lee Y. YolactEdge: Real-time instance segmentation on the edge. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an: IEEE, 2021; pp.9579–9585. doi: .

[34] Young S N, Kayacan E, Peschel J M. Design and field evaluation of a ground robot for high-throughput phenotyping of energy sorghum. Precision Agriculture, 2019; 20: 697–722.

[35] Miao Z H, Yu X Y, Li N, Zhang Z, He C X, Li Z, et al. Efficient tomato harvesting robot based on image processing and deep learning. Precision Agriculture, 2023; 24: 254–287.

[36] Peng H X, Xue C, Shao Y Y, Chen K Y, Liu H N, Xiong J T, et al. Litchi

detection in the field using an improved YOLOv3 model. Int J Agric & Biol Eng, 2022; 15(2): 211–220.

[37] Omasa K, Ono E, Ishigami Y, Shimizu Y, Araki Y. Plant functional remote sensing and smart farming applications. Int J Agric & Biol Eng, 2022; 15: 1–6.

[38] Li H H, Wei Y Y, Zhang H M, Chen H, Meng J F. Fine-grained classification of grape leaves via a pyramid residual convolution neural network. Int J Agric & Biol Eng, 2022; 15(2): 197–203.

[39] Yin X, Li W H, Li Z, Yi L L. Recognition of grape leaf diseases using MobileNetV3 and deep transfer learning. Int J Agric & Biol Eng, 2022; 15(3): 184–194.

[40] Yang Z K, Li W Y, Li M, Yang X T. Automatic greenhouse pest recognition based on multiple color space features. Int J Agric & Biol Eng, 2021; 14(2): 188–195.

[41] Sandler M, Howard A, Zhu M L, Zhmoginov A, Chen L-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City: IEEE, 2018; pp.4510–4520.

[42] Leng Z Q, Tan M X, Liu C X, Cubuk E D, Shi X J, Cheng S Y, et al. Polyloss: A polynomial expansion perspective of classification loss functions. arXiv Preprint, 2022; arXiv: 2204.12511. .