IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 29, 2020, accepted March 15, 2020, date of publication March 18, 2020, date of current version March 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2981823

Fast Method of Detecting Tomatoes in a Complex Scene for Picking Robots

ZHI-FENG XU[1], RUI-SHENG JIA[1,2], YAN-BO LIU[1], CHAO-YUE ZHAO[1], AND HONG-MEI SUN[1,2]

[1]College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
[2]Shandong Province Key Laboratory of Wisdom Mine Information Technology, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding authors: Rui-Sheng Jia (jrs716@163.com) and Hong-Mei Sun (shm0221@163.com)

This work was supported in part by the Natural Science Foundation of Shandong Province, China, under Grant ZR2018MEE008, and in part by the Key Research and Development Program of Shandong Province, China, under Grant 2017GSF20115.

ABSTRACT At present, there are two main problems with fruit-and vegetable-picking robots. One is that complex scenes (with backlighting, direct sunlight, overlapping fruit and branches, blocking leaves, etc.) obviously interfere with the detection of fruits and vegetables; the other is that an embedded platform needs a lighter detection method due to performance constraints. To address these problems, a fast tomato detection method based on improved YOLOv3-tiny is proposed. First, we improve the precision of the model by improving the backbone network; second, we use image enhancement to improve the detection ability of the algorithm in complex scenes. Finally, we design several groups of comparative experiments to prove the rationality and feasibility of this method. The experimental results show that the f1-score of the tomato recognition model proposed in this paper is 91.92%, which is 12% higher than that of YOLOv3-tiny; the detection speed on a CPU can reach 25 frames/s, and the inferential speed is 40.35 ms, equivalent to that of YOLOv3-tiny. Through comparative experiments, we can see that the comprehensive performance of our method is better than that of other state-of-the-art methods.

INDEX TERMS Real-time object detection, deep learning, picking robot, tomato, embedded device.

I. INTRODUCTION

The detection of fruit is the primary task and a major design difficulty for fruit- and vegetable-picking robots. The precision of detection is related to the efficiency of the picking robot. The occlusion or overlapping of fruits has always been a difficult problem in fruit detection [1]. Fruit recognition has been widely studied at home and abroad [2]–[9]. The detection methods for fruit targets include chromatic aberration [10], [11], k-means clustering [12], fuzzy C-means clustering [13], K-nearest neighbors [14], artificial neural networks [15], and support vector machines [16], [17]. The above methods can detect fruit in an image, but detection is based on the color, shape or texture of the fruit. When the color of the fruit surface is uneven, shadowed or blocked due to light or natural environmental factors, the detection precision will be significantly reduced. Zhao Jinying *et al.* used the Otsu dynamic threshold segmentation method [18] based on R-G color features to segment an image and determine the locations of tomatoes. However, this method is very sensitive to light and loses its object in a backlit environment. Wachs *et al.* [12] used a combination of color images and thermal images to identify apples, but the method of thermal imaging systems can only be used in the afternoon under direct sunlight, which cannot meet the requirements of the all-weather work of picking robots. Compared with conventional methods, deep convolutional neural networks have shown great advantages in the field of object detection in recent years. A deep neural network makes it possible to recognize tomatoes in complex situations because of its automatic extraction of high-dimensional features. Convolutional neural networks are mainly divided into two methods. The first is two-stage object detection. The core idea of the two-stage object detection method is to generate region proposals and classify the region. Representative methods are R-CNN [19], Fast R-CNN [20] and Faster R-CNN [21]. Sa *et al.* [22] used Faster R-CNN to detect sweet peppers and oranges. The precision of the model is very high, and its generalization ability is strong, but because the regional proposal step consumes many computing resources, the detection time

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa Rahimi Azghadi.

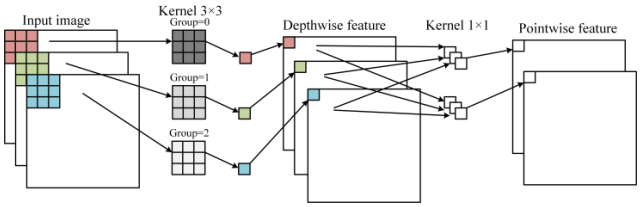


FIGURE 1. Depth-separable convolution.

is great, and the model cannot meet real-time processing requirements. The other method is one-stage object detection. Its core idea is to use a single CNN to process a whole image directly to detect an object and predict the object category. It is usually faster than the two-stage method, and the representative methods include SSD [23] and YOLO [24]. Real-time scene-parsing through object detection running on an embedded device is very challenging due to the limited memory and computing power of embedded devices [25]. Due to the limitation of computing resources, it is obvious that the current one-stage methods are not suitable for embedded devices except for YOLOv3-tiny.

For the above problems, based on YOLOv3-tiny, we propose a fast tomato detection method for a picking robot in complex scenes. To make the method suitable for embedded devices, an improved depth-separable convolution and residual structure replace the standard convolutional network in the original method. This increases the depth of the network and greatly reduces the amount of calculation to achieve a good balance between the precision and real-time performance of tomato detection. Then, to overcome the problems of overlapping fruits and branch occlusion, the features of the tomato are extracted by a neural network, and the region of the tomato is identified and detected. In addition, the data amplification and multiscale training strategies are integrated to improve precision while maintaining detection speed. Finally, we use the image enhancement algorithm in the training and detection phase of the model to solve the problem of tomato detection under the conditions of backlighting, darkness and other light conditions. By increasing the contrast of tomato images, the detection ability of the robot in a complex illumination scene is improved. In this paper, a number of comparative experiments are designed to validate the rationale and feasibility of this method.

II. RELATED WORK

A. DEEP SEPARABLE CONVOLUTION

Deep separable convolution [25] is a form of factorized convolution, which can be divided into two types of convolutions: depthwise convolution (DW) and pointwise convolution (PW). Its details are shown in Figure 1. DW is different from standard convolution. The convolution kernel of standard convolution is used in all input channels, while DW convolution uses different convolution kernels for each input channel. The difference between PW and standard convolution is that PW only uses a 1×1 convolution kernel.

The depth-separable convolution process is as follows: first, DW convolution is used to convolve different input channels, and then PW convolution is used to combine the result with the previous output. The number of FLOPs is an index used to measure the computation amount of a convolutional neural network. The higher the number of FLOPs, the larger the amount of computation and the more system resources are used by CNN. We use the formula below to compare the FLOPs of a standard convolution and a depth separable convolution.

The calculation process of standard convolution is as follows:

$$K_h \times K_w \times C_{in} \times C_{out} \times W \times H \tag{1}$$

In formula (1), $K_h \times K_w$ is the convolution kernel size, C_{in} is the number of input channels, and C_{out} is the number of output channels. W and H are the width and height of the output characteristic map respectively.

The calculation process of the first depth-separable convolution is as follows:

$$K_h \times K_w \times C_{in} \times W \times H + C_{in} \times C_{out} \times W \times H \tag{2}$$

The ratio of the floating-point operations of depth- separable convolution and standard convolution is:

$$\frac{K_h \times K_w \times C_{in} \times W \times H + C_{in} \times C_{out} \times W \times H}{K_h \times K_w \times C_{in} \times C_{out} \times W \times H} = \frac{1}{C_{out}} + \frac{1}{K_{h,w}^2} \tag{3}$$

From formula (3), we see that when the convolution kernel size is 3×3, the number of FLOPs of depth-separable convolution is approximately 1/9 that of standard convolution. A new idea of depth-separable convolution is proposed: different convolution kernels are used to convolve different input channels, which decomposes the standard convolution operation into two processes. The combined effect of the two convolutions is similar to that of a standard convolution, and the number of calculations and model parameters are greatly reduced. Therefore, we use the improved depth-separable convolution instead of standard convolution, which is helpful in building a lightweight and efficient backbone network.

B. YOLOv3-TINY

YOLO is an end-to-end object detection algorithm based on deep learning, which has the advantages of fast running speed and real-time effect in a GPU environment. YOLOv3 [26] is the third iteration of YOLO, and it has improved detection precision compared with that of YOLO and YOLOv2 [27]. YOLOv3 uses darknet-53 for feature extraction, and darknet-53 is more powerful than the darknet-19 of YOLOv2. YOLOv3 uses multiscale prediction, that is, detection on multiscale feature map, to improve the precision of object detection. However, the number of parameters of YOLOv3-416 is 65.86 bn, the model size is approximately 237 m, and the detection speed is only 1∼2 frames/s on an embedded platform. Therefore, YOLOv3 is not suitable

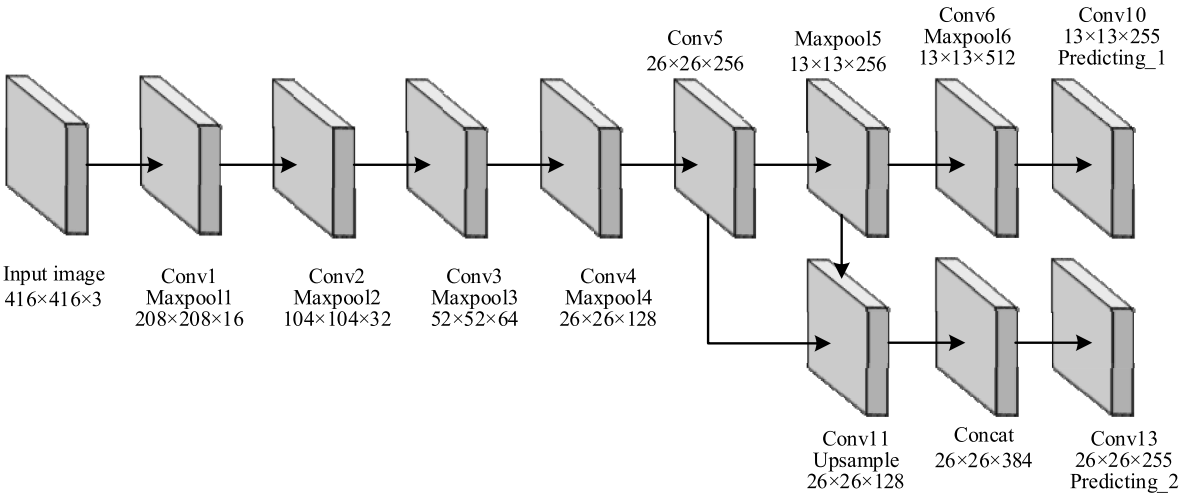


FIGURE 2. YOLOv3-tiny network.

for embedded devices. YOLOv3-tiny is the lightweight version of the YOLOv3 model, and the network structure of YOLOv3-tiny is shown in Figure 2.

The backbone network of YOLOv3-tiny is a 7-layer standard convolution structure rather than a Darknet series. YOLOv3-tiny is the same as the end-to-end object detection method. First, the input layer is a 416 × 416 image, and then after 10 convolutions and 6 subsampling operations, the output feature maps have a size of 13 × 13. At the same time, the feature map after the 5th downsampling is unsampled and convolved to obtain a size of 26 × 26 × 128, and it is subjected to standard convolution twice to obtain a size of 26 × 26 × 255. The output feature maps of both scales contain the prediction information of objects. The number of FLOPs of YOLOv3-tiny is only 5.56 bn, and the model size is only 33.7MB. It can run on embedded or mobile devices. However, its backbone network has only 7 layers, so it cannot extract higher level semantic features, and its precision is low. In practical applications, YOLOv3-tiny can only detect tomatoes close to the camera and that are relatively complete, and the detection precision is poor in complex scenes such as those involving dim light, backlighting, occlusion and so on.

III. PROPOSED SOLUTION

A. NETWORK

At present, the high computing resources of advanced object detection algorithms exceed the capabilities of embedded and mobile devices. For this reason, we design a new neural network architecture based on low computation and high precision. We think that deepening the network layer can make YOLOv3-tiny extract more abundant convolution features. Considering that the deep network model increases the detection time, which is not conducive to real-time performance, the improved depth-separable convolution and residual modules are used to form the backbone network of the detection algorithm. The tomato detection network is shown in Figure 3, where x and y are the prediction coordinates of the x-axis and y-axis, respectively, w is the prediction width, h is the prediction height, and Pr is the prediction confidence.

In Figure 3, the tomato detection network framework includes two parts: the backbone network and the prediction network. With the goal of a lightweight network and enhancement of the feature extraction ability, we redesigned the basic unit of the backbone network and called it the PDP structure. The backbone network consists of one standard convolution and nine PDP structures. With this structure, multilayer feature reuse and fusion can be realized, and the amount of computation introduced by the new structure can be reduced. To detect tomatoes at different distances, the prediction network consists of two branches, corresponding to the output of two scales. The following describes the differences between the PDP structure, standard convolution and depth-separable convolution.

Standard convolution with a BN layer and the a ReLU is shown in Figure 4 (a). To speed up the model convergence, the BN (batch normalization) layer is usually placed between the standard convolution component and the ReLU. The depth separable convolution component is shown in Figure 4 (b), and the improvement of Figure 4 (a) mainly uses DW convolution and PW convolution instead of standard convolution. The effect of this structure is similar to that of standard convolution, which greatly reduces the computation.

As shown in Figure 4 (c), the PDP structure is a combination of PW convolution and the structure in Figure 4 (b). To enhance the propagation of gradients, a PW convolution is added before the structure in Figure 4 (b). After adding PW convolution, DW convolution can extract the features of the high-dimensional space. At the same time, a BN layer is added after each step of convolution to speed up the convergence of the model. The output of the first step and the second step uses the nonlinear activation function ReLU6, which makes the model more robust in low-precision calculations. In the third step, the output of the operation does not use the activation function, and the direct linear output reduces

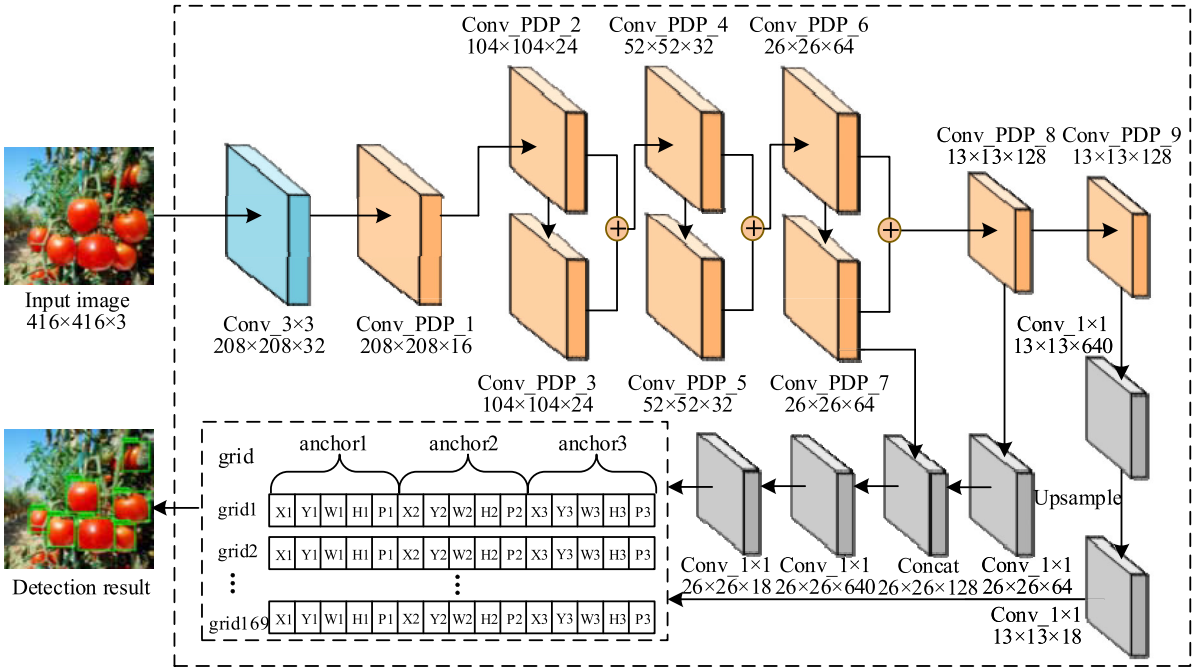


FIGURE 3. Proposed network.

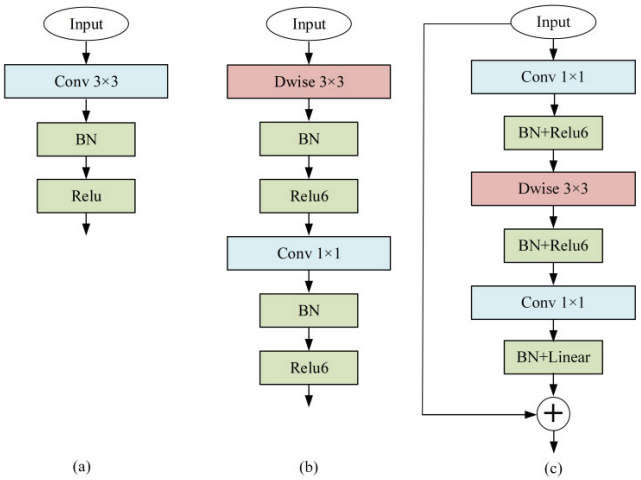


FIGURE 4. (a)Standard convolutional layer with batchnorm and a ReLU; (b) Depthwise-separable convolutions with depthwise and pointwise layers followed by batchnorm and a ReLU6; (c) PDP structure.

information loss. We use the PDP structure to replace the standard convolution of the original network, which greatly reduces the computation. By increasing the number of channels and network layers, we can improve the precision of the model and make it easy to migrate it to embedded devices, mobile devices and other smaller systems.

1) BACKBONE

In Figure 3, n in Conv_PDP_n indicates the number of times the PDP structure is currently used. The backbone network first uses 3 × 3 convolution kernels for the standard convolution of the input image to obtain conv1 to extract features and upgrade dimensions. To enrich the network feature information, we use the idea of residuals to add Conv_PDP_2 and Conv_PDP_3 (we add the feature map and keep the number of channels unchanged) and then use the PDP structure to generate Conv_PDP_4. Conv_PDP_6 and Conv_ PDP_8 are obtained similarly. Therefore, the backbone network executes the PDP structure 9 times until Conv_PDP_9 is generated. We use a convolution of stride=2 to replace the convolution and max-pooling, so that the parameter amount is unchanged, the calculation amount becomes one quarter that of the original network, and the amount of calculation involved in the max-pooling is omitted.

2) PREDICTION NETWORK

The prediction network consists of two branches, corresponding to the output of two scales. The first branch outputs a 13 × 13 × 18 feature map. When the feature map reaches Conv_PDP_8 in the backbone network, the second branch is generated in parallel. The second branch is fused with the 26 × 26 × 64 output of the first branch, and the final output is 26 × 26 × 18. In this paper, the three scale feature maps are not generated by imitating YOLOv3, because the 52 × 52 prediction grid in the original network is responsible for detecting objects at a far distance. In actual detection of dense tomatoes, only tomatoes with a resolution greater than 16 × 16 need to be detected. Small tomatoes are too far away to be the object of the picking robot. At the same time, using a 52 × 52 prediction grid makes the prediction tensor too large, increasing the detection time. To shorten the network training time, anchor boxes are used to help predict bounding boxes.

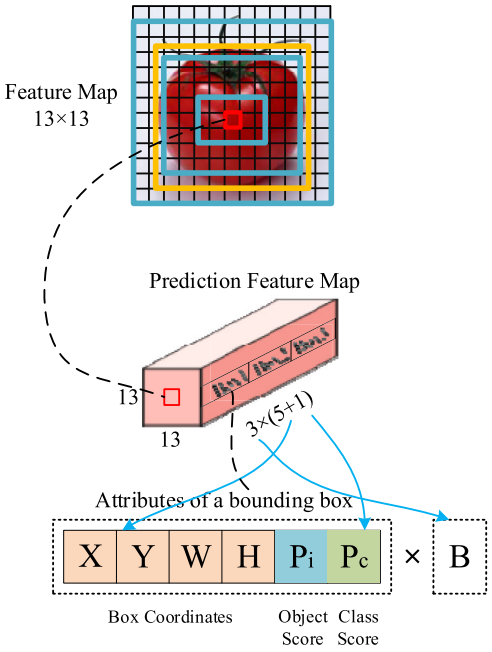


FIGURE 5. Prediction for a 13 × 13 feature map.

The prediction information contained in the 13 × 13 feature diagram is shown in Figure 5.

We used k-means clustering to calculate 6 anchors of the current dataset: (50 × 66), (74 × 99), (91 × 125), (113 × 154), (140 ×190), and (220 × 284). K-means uses the Euclidean distance. For the output of the backbone network, the first three kinds of anchors are used for the 13 × 13 feature map, and the last three kinds of anchors are used for the 26 × 26 feature map to predict three boxes per grid. For 13 × 13 and 26 × 26 outputs, the parameters of each box include x, y, w, and h, the confidence P_i of the tomato and the probability P_c.

3) LOSS FUNCTION

The loss function of this algorithm consists of four parts. The first part concerns the prediction of central coordinates, as shown in equation (4); the second part concerns about the prediction of the boundary box regression, as shown in equation (5); the third part concerns the prediction of object categories, as shown in equation (6); the fourth part concerns the prediction of object confidence, as shown in equation (7).

$$Loss_1 = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \quad (4)$$

In formula (4), x_i and y_i are the x and y coordinates of the predicted object, and $\hat{x}_i$ and $\hat{y}_i$ are the x and y coordinates of the actual object.

$$Loss_2 = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \ell_{ij}^{obj} \times \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{y_i} - \sqrt{\hat{y}_i} \right)^2 \right] \quad (5)$$

In formula (5), w_i and h_i are the width and height of the predicted object, respectively, and $\hat{w}_i$ and $\hat{h}_i$ are the width and height of the actual object, respectively.

$$Loss_3 = \sum_{i=0}^{S^2} \ell_{ij}^{obj} \sum_{j=0}^{B} \left[\left(p_i(C) - \hat{p}_i(C) \right)^2 \right] \quad (6)$$

In formula (6), $p_i(C)$ is the confidence of the predicted object and $\hat{p}_i(C)$ is the confidence of the actual object.

$$Loss_4 = \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[(C_i - \hat{C}_I)^2 \right] + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} \left[(C_i - \hat{C}_I)^2 \right] \quad (7)$$

In formula (7), C_i is the category of the predicted object, and $\hat{C}_I$ is the category of the actual object. The parameters λ are used to weight parts of the loss functions differently. This is necessary to increase the model stability. $\lambda_{coord} = 5$ and $\lambda_{noobj} = 0.5$ are the weights to balance the proportion of each $Loss_i$. S is a grid cell, B is a bounding box, *obj* contains an object, and *noobj* does not contains an object. The total loss is:

$$Total\ Loss\ Function = \sum_{i=1}^{4} Loss_i \quad (8)$$

B. TOMATO TRAINING AND DETECTION PROCESS

The main process of fast detection of tomatoes includes two parts: model training and model inference. As shown in Figure 6, model training requires the tomato dataset and the boundary box labels to be fed into the convolutional neural network; iterative training is then conducted, and a fully trained model is obtained. In model inference, the image acquired by the robot camera is input into the trained model, and the position of a tomato in the image is acquired.

The specific process is as follows:

1) TRAINING DETAIL

Step 1. Data collection. A total of N tomato images in the actual data set and network were collected, which is denoted as $f_N(x, y)$. An individual image is denoted as $f_i(x, y)$.

Step 2. Dataset preprocessing. First, $f_i(x, y)$ is cut and scaled to 416 × 416. The bounding box of a tomato in $f_i(x, y)$ is then manually labeled. The information of the bounding box is composed of the upper left coordinate (x_1, y_1) and the lower right coordinate (x_2, y_2). The bounding box information is saved in the format of the Pascal VOC dataset and recorded as L_N.

Step 3. Image space conversion. By transforming $f_i(x, y)$ from the *RGB* space to the *HSI* space, three channel components H (hue), S(saturation) and I (brightness) are obtained.

Step 4. Dataset image enhancement. We use adaptive histogram equalization for the brightness channel I, and the result is denoted as $Clahe(I)$. Then, H, S,and $Clahe(I)$ are

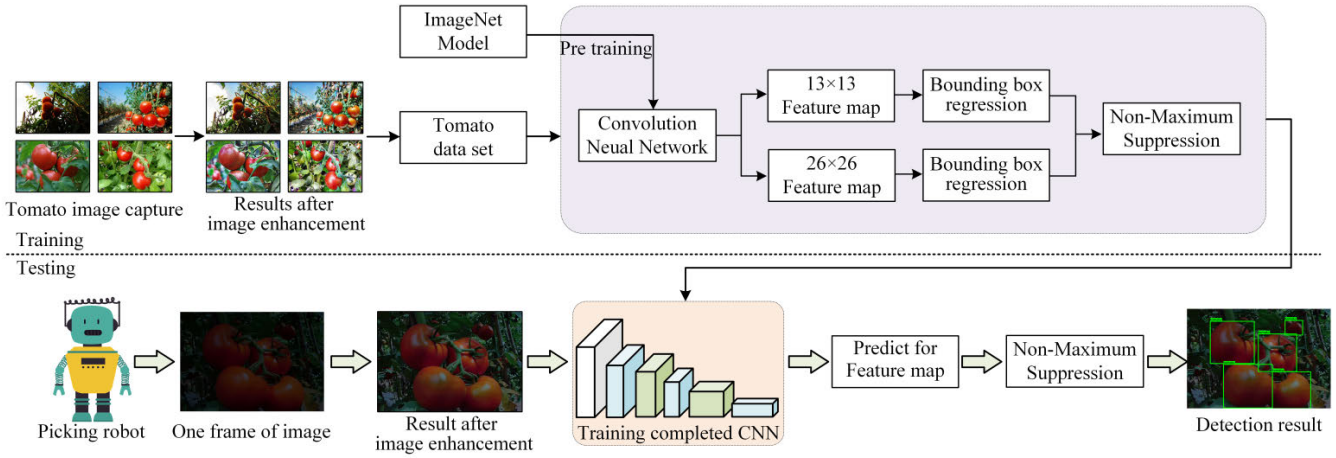


FIGURE 6. The training and detecting process of the tomato detection method.

fused into a new HSI image. Finally, $f_i(x, y)$ is transformed from the *HSI* space to the *RGB* space, and the enhanced $f_i(x, y)$ is recorded as $F_i(x, y)$. The dataset obtained after image enhancement is denoted as $F_N(x, y)$.

Step 5. Training network model. First, the Imagenet model is used to initialize the backbone network parameters, and then $F_i(x, y)$ and L_N are input into the network framework. After the processing of the backbone network and prediction network, the feature maps of scales 13×13 and 26×26 are obtained. The feature map contains the coordinates of the upper left corner (x, y), w, h, P_i and P_c. The loss function is used to adjust the parameters and judge whether the loss is minimal. If the loss is minimal, the model is saved.

2) DETECTING DETAIL

Step 1. Obtaining the tomato image. Fruit- and vegetable-picking robots obtain real-time video through cameras. Extracting the video frame by frame, the extracted image are recorded as $p_j(x, y), j = 1, 2, 3 \cdots M$. Finally, we obtain a set of tomato images $p_M(x, y)$.

Step 2. Tomato image space conversion. By transforming $p_j(x, y)$ from the *RGB* space to the *HSI* space, the 3 channel components H (hue), S (saturation) and I (brightness) are obtained.

Step 3. Enhancing the tomato image. Finally, the group of images to be detected after image enhancement is recorded as $P_M(x, y)$.

Step 4. Inputting the enhanced tomato image input to the model. $P_j(x, y)$ is input into the trained model to obtain a set of prediction boxes. Non-maximum suppression is used to filter the prediction boxes with high overlap and output the final prediction box.

IV. EXPERIMENT

We use the Adam optimization function to train the network in an end-to-end joint manner. During network training, a model pretrained on Imagenet is used to initialize the network parameters. The initial learning rate is set to 0.001, the weight attenuation rate is set to 0.0005, and the verification period is set to 50. When the model reaches convergence, the training is stopped, and the model is saved.

A. TOMATO DATASET

The main collection site of the tomato dataset is a high- efficiency agricultural base in Qingdao, China. The main varieties collected are Xiazhuang-69, Dahong-1 and Xiafen-F1, which have the most common tomato characteristics. On July 22 and August 25, 2019, using the iPadPro 11 camera (five-mirror lens, 12 megapixels and f/1.8), image acquisition was conducted in four directions 1∼2 m from each crown: east, west, north and south. Under the conditions of smooth lighting and backlighting, a total of 800 images of naturally growing tomato crowns were collected. To ensure the diversity of the tomato dataset, 200 tomato images were acquired and screened by a web scraping tool, taking into account the influence of the dataset proportion on the training balance. Preliminary experiments suggested that the ratio of the number of images in daytime, evening and night should follow the ratio of 2:1:1. However, in all three cases, the number of tomatoes should be the same. Because the light in the daytime is more complex and diverse (side light, backlight, etc.), and the light in the evening and night is similar and uniform, the number of daytime images should be greater. There are 4378 tomatoes in the dataset of 1000 tomato images. Among the images, 500 (167 in smooth light,167 in side light and 166 in backlight, for a total of 2310 tomatoes) were taken in the daytime, 250 (1157 tomatoes) in the evening and 250 (1011 tomatoes) at night. Some tomato dataset images are shown in Figure 7, which are in side light, smooth light, backlight, dark, blocked, dense and other conditions.

In this experiment, 1000 tomato images were collected in JPG format. To shorten the training time of the model, the images were cropped uniformly and reduced to

TABLE 1. Comparison of test results of advanced detection based on the tomato dataset.

	Method	Backbone	Precision	F1-Score	FLOPs	FPS	Parameters	Time(ms)
Two-stage	Faster R-CNN [21]	VGG-16	87.08	89.13	132.24	—	137.12	—
	Faster R-CNN [21]	ResNet-101	89.10	91.43	115.25	—	105.31	—
	R-FCN [30]	ResNet-101	91.13	93.69	137.89	—	112.07	—
	Mask R_CNN [31]	ResNeXt-101	**92.75**	**95.27**	135.34	—	124.21	—
One-stage	SSD [23]	VGG-16	86.89	88.72	73.31	0.5	70.08	2094
	RetinaNet400[32]	ResNet-101	89.26	92.36	55.56	0.25	53.49	3978
	YOLOv3 [26]	DarkNet-53	**90.38**	**92.76**	65.86	1	62.62	1047
	YOLOv3-tiny	Tiny	77.21	78.67	7.36	23	10.05	43.47
	Mini-YOLOv3[25]	Mini-Darknet	89.77	90.31	14.81	13	14.50	80.24
	Slim-YOLOv3[33]	Slim Net	81.40	83.76	30.51	11.70	20.80	89.41
	Our Method	Proposed Net	88.70	91.92	**5.56**	**25**	**8.71**	**40.35**

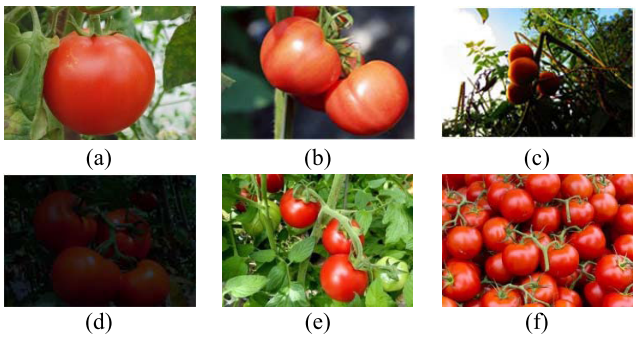


FIGURE 7. (a) Side-lit tomato; (b) tomatoes under smooth light; (c) backlit tomatoes; (d) tomatoes under dark light; (e) shaded tomatoes; (f) dense tomatoes.

416 × 416 pixels. We considered the appearance and shape of the tomatoes in the process of sample labeling, and we labeled 4378 tomato bounding boxes in the 1000 original samples. We used software to annotate the dataset manually, and the annotation information was saved in Pascal VOC dataset format, including the categories and bounding boxes of the object tomatoes.

Image enhancement of training samples can improve the quality of the samples and improve the detection precision of the CNN. In an orchard under natural light, especially in a scene with high light intensity, a shadow on the fruit surface is caused by the mutual occlusion or backlighting of tomatoes, which makes the color of the fruit very different from that of the diffuse reflection area under normal light and affects the quality of the tomato image. The data quality will affect the model training. The training of deep learning models requires a large amount of data, and 1000 images are not sufficient. Before the training, the dataset was enhanced [29]; in this process, the change range of the hue was from 1 to 1.5 times, the change range of the exposure was from 1 to 1.5 times, and the change range of the number of colors was from 0.9 to 1.1 times, so 5500 images were ultimately generated for training.

B. ENVIRONMENT AND EVALUATION INDEX

In this experiment, based on the i5-7300HQ CPU and NVIDIA GTX 1050ti GPU, a TensorFlow deep learning framework was built under in Windows 10. The training and detection of the tomato object detection network model was programmed in Python.

We use the precision, f1-score, FLOPs, FPS, parameters, and time (ms) evaluation indexes to evaluate our method and other comparison methods. The calculation formulas of precision (P) and f1-score (F1) are shown in formula (9).

$$p = \frac{TP}{TP + FP}, \quad R = \frac{TP}{TP + FN}, \quad F1 = \frac{2PR}{P + R}, \tag{9}$$

where P is the precision rate, R is the recall rate, TP is the number of true positive samples, FP is the number of false positive samples, and FN is the number of false negative samples.

C. RESULTS AND ANALYSIS

In Table 1, our method is compared with some state-of-the-art object detection methods. All methods were trained with the tomato dataset and optimal parameters. The comparison information involved in the table includes the backbone, precision, f1-score, FLOPs, FPS, parameters and time (ms), all of which are calculation results on the CPU. Since the FPS and time of the two-stage methods are far lower than that those of one-stage methods, the FPS and time in the two-stage methods are not discussed.

Table 1 shows that the one-stage object detection method is significantly faster than the two-stage detection method. The six indexes of our method are better than those of YOLOv3-tiny. The precision of our method is close to that of YOLOv3, the time is 1/25 that of YOLOv3, and the number of FLOPs is 1/13 that of YOLOv3. We think that the main reason for maintaining the precision and having few FLOPs is that a combination of the PDP structure and residual structure is used in the backbone network. Although Faster R-CNN, R-FCN [30], and Mask R_CNN [31], have higher precision

FIGURE 8. The detection results of our method in some tomato datasets.

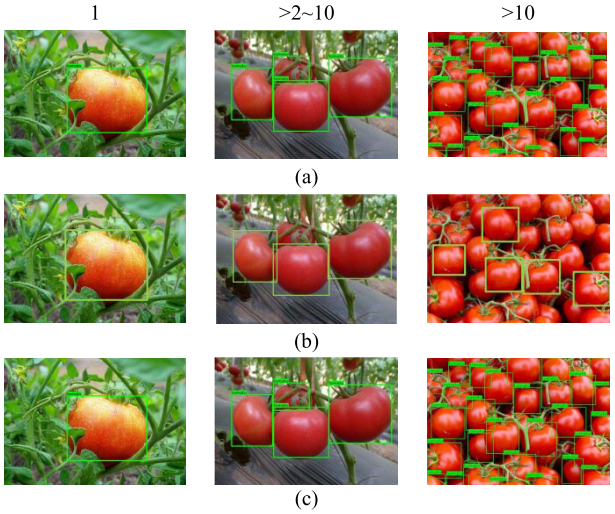


FIGURE 9. Three methods were used to detect different amounts of tomatoes. (a) Detection result of the YOLOv3 method; (b) detection result of the YOLOv3-tiny method; (c) detection result of our method.

than our method, our method is much faster than these two-stage methods. In particular, we compared two of the latest excellent lightweight methods: Mini-YOLOv3 [25] and SlimYOLOv3 [33]. The experimental results show that the indexes of our method are better than those of two methods.

We use the trained model to test some tomato datasets, as shown in Figure 8. According to Figure 8, our method can successfully detect tomatoes in smooth, backlit, dark, dense, occluded and complex scenes.

To further verify the effectiveness of the model, it is necessary to evaluate the detection efficiency of the method under various practical conditions. In the next experiments, we took the number of tomatoes, light intensity and picking time as the control variables; the test results of three methods (YOLOv3, YOLOv3-tiny and our method) under the above conditions were compared, and the performance of the methods was evaluated with the f1-score.

1) COMPARISON OF DETECTION RESULTS UNDER DIFFERENT NUMBERS OF TOMATOES

In the actual picking process of the robot, the number and size of tomatoes changes with the distance between the camera and the tomato tree. When the number of tomatoes is small and their size is large, the object to be detected is clear and complete, so it is easier to detect. However, in a multi-object image, due to the reduction in size and the increase in the number of objects, adhesion and occlusion may occur, and detection is difficult. Therefore, we set up a comparative experiment of tomato detection under different numbers of tomatoes, which are divided into one tomato, multiple tomatoes and dense tomatoes, to compare the detection performance of the three methods under different numbers of tomatoes. The test results are shown in Figure 9.

In the test set of this experiment, 336 images containing 1410 tomatoes, are divided into three categories according to the number of fruits. In these images, 75 one-tomato images contain 75 tomatoes, 203 many-tomato images contain 576 tomatoes, and 58 dense tomato images contain 759 tomatoes. The resolution of the smallest tomatoes in the dense tomato images is not less than 15×15. For each class, 30 images are randomly selected as the experimental test set, and three different methods are used to detect the positive sample number; the total sample number is synthesized, as well as the undetected sample number and the false detection sample number, and the parameters rate and the recall rate are calculated to obtain the f1-score. We repeat the above steps three times, take the average, and finally average the three types of results to obtain the comprehensive results, as shown in Table 2.

TABLE 2. Test results of three methods for images with different numbers of tomatoes.

Tomato number	Method	F1%			
		1	2	3	Average
1	YOLOv3	96.56	96.62	95.76	96.31
	YOLOv3-tiny	86.95	83.94	89.42	86.77
	Our Method	95.23	96.14	95.67	95.68
>2~10	YOLOv3	93.17	91.41	93.89	92.82
	YOLOv3-tiny	83.23	80.57	79.63	81.14
	Our Method	87.49	86.34	86.92	86.91
>10	YOLOv3	70.72	69.59	71.79	70.70
	YOLOv3-tiny	41.43	37.36	39.33	39.37
	Our Method	66.29	68.23	64.85	66.45
Average	YOLOv3	86.81	85.87	87.14	86.60
	YOLOv3-tiny	70.53	67.29	69.46	69.09
	Our Method	83.00	83.57	83.01	83.01

It can be seen from Table 2 that the algorithm of YOLOv3 performs best. The f1-score of our method is higher

TABLE 3. Test results of three methods for different illumination of tomato images.

Illumination angles	Method	F1%			
		1	2	3	Average
Side Light	YOLOv3	97.25	97.38	96.89	97.17
	YOLOv3-tiny	87.57	85.94	89.42	87.64
	Our Method	96.36	95.76	96.21	96.11
Back Light	YOLOv3	94.59	94.32	93.71	94.21
	YOLOv3-tiny	80.52	78.44	81.69	80.22
	Our Method	85.34	81.49	83.27	83.37
Direct Sunlight	YOLOv3	96.28	94.26	97.65	96.06
	YOLOv3-tiny	86.65	88.74	85.25	86.88
	Our Method	91.43	93.27	89.87	91.52
Average	YOLOv3	96.04	95.32	96.08	95.81
	YOLOv3-tiny	84.91	84.37	85.45	84.91
	Our Method	91.04	90.17	89.78	90.33

than that of YOLOv3-tiny and close to that of YOLOv3. The fewer tomatoes there are, the higher the f1-score. When the number increased from 1 to less than 10, the f1-score did not decrease much. However, for dense tomato images, f1-score dropped by nearly 20 percentage points. Because of the different sizes of tomatoes in dense images and situations of adhesion and occlusion, some tomatoes are not detected. At the same time, because of the inherent characteristics of convolutional neural networks, continuous convolution will lose the characteristics of small objects when the deep network has high resolution. Base on the comprehensive results, our method is competent for the detection of different numbers of tomatoes.

2) COMPARISON OF DETECTION METHODS UNDER DIFFERENT ILLUMINATION ANGLES

In the experiment in this section, the angle of light on the tomato at the time of shooting is taken as the control variable, and the lighting conditions include side light, backlight and even light. Among the images, 124 side-lit images include 357 tomatoes, 67 backlit images include 149 tomatoes, and 87 images with direct sunlight include 145 tomatoes. The results are shown in Figure 10, and the statistical results are shown in Table 3.

Figure 10 shows that the texture of a tomato is clear under side light, the surface light intensity is uniform, and it is easier to detect. The brightness of the tomato and its branches and leaves decreased remarkably under the backlight condition, and the boundary between them was not obvious. When the tomato is under direct sunlight, part of its surface brightness increases, and the color of the tomato is bright white without texture features. In the latter two cases, the difficulty of tomato detection increased significantly.

As seen from Table 3, the F1 of the YOLOv3 method is the highest. Because of its backbone(darknet-53), it has

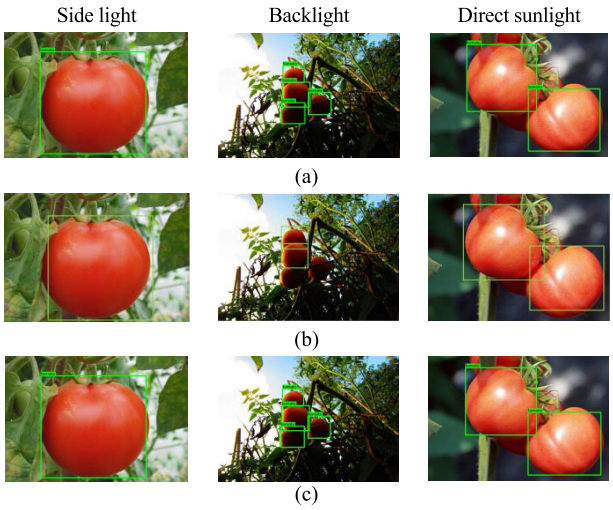


FIGURE 10. Detection results of tomatoes by three methods under different illumination. (a) Detection result of the YOLOv3 method; (b) detection result of the YOLOv3-tiny method; (c) detection result of our method.

a strong feature extraction ability. The f1-score of our method is approximately 5 percentage points lower than that of YOLOv3 and 4 percentage points higher than that of YOLOv3-tiny. The performance of the three methods is worst under backlight conditions. The reason is the lack of brightness under backlighting, which directly eliminated some tomatoes, reducing the model f1-socre.

3) COMPARISON OF DETECTION METHODS UNDER DIFFERENT PICKING TIMES

In this experiment, the imaging time of the tomatoes was taken as the control variable, and the conditions were day, evening and night. In the tomato dataset, the tomato images in the daytime include three kinds of lighting scenes (167 in smooth light, 167 in side light and 166 in backlight). The radio of images in the three kinds of scenes is 1:1:1. Backlit images are taken during the day and tend to be darker. Therefore, the characteristics of backlit images are similar to those of nightfall and dark scenes. The number of images in the evening and in the dark is 250. It is assumed that the number of backlit images is evenly distributed among the images in the evening and in the dark. We can obtain 334 images in brighter scenes (daytime), 333 in the evening and 333 in the dark. To maintain the distribution of proportions under various conditions in the dataset, and we weight the images by the number of tomatoes. Because of the backlit images, the total number of images in daylight conditions is greater than that in evening and dark conditions. This may affect the f1-score of daylight conditions, so we removed the images under backlighting conditions in this comparison experiment. We selected images of tomatoes in daylight (35 in smooth light, 35 in side light), with a total of 150 tomatoes, 70 images of tomatoes in the evening, with a total of 153 tomatoes and 70 images of tomatoes in the night, with a total of 151 tomatoes. The experimental method is the same as that in the previous section. The specific implementation results are

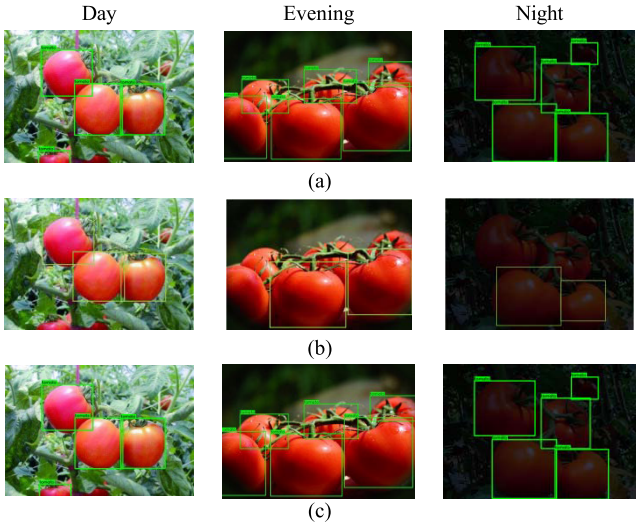


FIGURE 11. Tomato detection results for three methods at different times. (a) Detection result of the YOLOv3 method; (b) detection result of the YOLOv3-tiny method; (c) detection result of our method.

TABLE 4. Test results of three methods for different illumination of tomato images.

Times	Method	F1%			
		1	2	3	Average
Day (not including backlight)	YOLOv3	93.53	94.37	93.26	93.72
	YOLOv3-tiny	89.21	89.23	88.45	88.96
	Our Method	93.67	92.89	93.18	93.25
Evening	YOLOv3	92.34	92.67	91.78	92.26
	YOLOv3-tiny	88.11	87.32	87.81	87.75
	Our Method	91.65	92.15	92.31	92.03
Night	YOLOv3	92.12	92.03	91.56	91.90
	YOLOv3-tiny	87.39	87.78	88.36	87.84
	Our Method	91.67	92.02	91.85	91.84
Average	YOLOv3	92.66	93.02	92.20	92.63
	YOLOv3-tiny	88.24	88.11	88.21	88.18
	Our Method	92.33	92.35	92.45	92.38

shown in Figure 11, and the statistical results are shown in Table 4.

As seen from Figure 11, tomatoes appear normally in the daytime, and their color and texture characteristics are in good condition. The features of tomatoes in the evening are similar to those in backlight. At night, due to the lack of light source, unlike in cases of daytime backlight and diffuse reflection of light on tomatoes, most parts of tomatoes are black. In this case, regardless of the color, nature, or texture characteristics of tomatoes, there are almost none; some of them are incomplete. Our method and the YOLOv3 method can successfully detect tomatoes at different times. The YOLOv3-tiny method can only detect the part of tomatoes closest to the camera.

It can be seen from Table 4 that the f1-score of the three methods are not significantly different in the three scenarios. The f1-score of our method in the three scenarios is close to the performance of YOLOv3. The performance of YOLOv3-tiny in all three scenarios is 5 percentage points lower than the performance of our method.

To summarize the above three experiments, the method of YOLOv3-tiny finds detection difficult in dense, backlit, evening, night and other environments, and its f1-score is generally low. The YOLOv3 achieves the best detection results, but it has the slowest speed and requires the most FLOPs. In contrast, the f1-score of our method under various conditions is close to that of YOLOv3, and its number of FLOPs is only 1/8 that of YOLOv3. Three groups of comparative experiments show that our method can adapt to the detection of tomatoes in most situations, which makes it possible for a tomato-picking robot equipped with this vision method to work all day.

V. CONCLUSION

Based on the YOLOv3-tiny detection method, we propose a fast detection method for tomatoes in complex scenes for picking robots. First, we use improved depth- separable convolution and a residual structure to replace the standard convolution network in the original network, which increases the network depth and greatly reduces the number of FLOPs. Then, a fusion of data amplification and multiscale training strategy improves precision while maintaining detection speed. Finally, we propose using an image enhancement algorithm in the training and detection phase of the model. By increasing the contrast of the tomatoes, the detection ability of robots in complex illumination scenes is improved. Experimental results show that compared with other state-of-art methods, our method has the fastest detection speed, the fewest FLOPs and higher detection precision. Because most fruit-picking robots are embedded devices or mobile devices, our method has obvious advantages.

REFERENCES

[1] R. Hussin, M. R. Juhari, N. W. Kang, R. C. Ismail, and A. Kamarudin, "Digital image processing techniques for object detection from complex background image," *Procedia Eng.*, vol. 41, pp. 340–344, Dec. 2012, doi: 10.1016/j.proeng.2012.07.182.

[2] G.-Q. Jiang and C.-J. Zhao, "Apple recognition based on machine vision," in *Proc. Int. Conf. Mach. Learn. Cybern.*, vol. 3, Jul. 2012, pp. 1148–1151.

[3] J. Lu and N. Sang, "Detecting citrus fruits and occlusion recovery under natural illumination conditions," *Comput. Electron. Agricult.*, vol. 110, pp. 121–130, Jan. 2015.

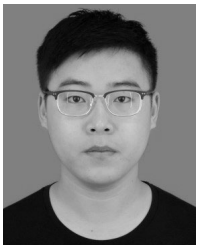
[4] W. Ji, D. Zhao, F. Cheng, B. Xu, Y. Zhang, and J. Wang, "Automatic recognition vision system guided for apple harvesting robot," *Comput. Electr. Eng.*, vol. 38, no. 5, pp. 1186–1195, Sep. 2012.

[5] M. Rizon, N. A. N. Yusri, M. F. A. Kadir, A. R. bin Mamat, A. Z. A. Aziz, and K. Nanaa, "Determination of mango fruit from binary image using randomized Hough transform," *Proc. SPIE*, vol. 9875, Dec. 2015, Art. no. 987503.

[6] A. Silwal, A. Gongal, and M. Karkee, "Identification of red apples in field environment with over-the-row machine vision system," *Agricult. Eng. Int. (CIGR J.)*, vol. 16, no. 4, pp. 66–75, 2014.

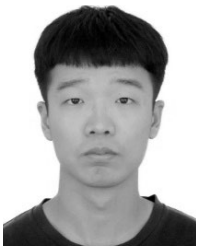
[7] J. Rakun, D. Stajnko, and D. Zazula, "Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry," *Comput. Electron. Agricult.*, vol. 76, no. 1, pp. 80–88, Mar. 2011.

[8] S. Chaivivatrakul, M. N. Dailey, "Texture-based fruit detection," *Precis. Agricult.*, vol. 15, no. 6, pp. 662–683, Dec. 2014.
[9] J. Feng, S. W. Wang, G. Liu, and L. H. Zeng, "A separating method of adjacent apples based on machine vision and chain code information," in *Proc. Int. Conf. Comput. Comput. Technol. Agricult.*, vol. 368, D. Li and Y. Chen, Eds. Berlin, Germany: Springer, 2012, pp. 258–267.
[10] A. Arefi, A. M. Motlagh, K. Mollazade, and R. F. Teimourlou, "Recognition and localization of ripen tomato based on machine vision," *Austral. J. Crop Sci.*, vol. 5, no. 10, pp. 1144–1149, 2011.
[11] R. Zhou, L. Damerow, Y. Sun, and M. M. Blanke, "Using colour features of cv.'Gala' apple fruits in an orchard in image processing to predict yield," *Precis. Agricult.*, vol. 13, no. 5, pp. 568–580, Oct. 2012.
[12] J. P. Wachs, H. I. Stern, T. Burks, and V. Alchanatis, "Low and high-level visual feature-based apple detection from multi-modal images," *Precis. Agricult.*, vol. 11, no. 6, pp. 717–735, Dec. 2010.
[13] A. Zhu and L. Yang, "An improved FCM algorithm for ripe fruit image segmentation," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Aug. 2013, pp. 436–441.
[14] R. Linker, O. Cohen, and A. Naor, "Determination of the number of green apples in RGB images recorded in orchards," *Comput. Electron. Agricult.*, vol. 81, pp. 45–57, Feb. 2012.
[15] A. Arefi and A. M. Motlagh, "Development of an expert system based on wavelet transform and artificial neural networks for the ripe tomato harvesting robot," *Austral. J. Crop Sci.*, vol. 7, no. 5, p. 699, 2013.
[16] L. Qiang, C. Jianrong, L. Bin, D. Lie, and Z. Yajing, "Identification of fruit and branch in natural scenes for citrus harvesting robot using machine vision and support vector machine," *Int. J. Agricult. Biol. Eng.*, vol. 7, no. 2, pp. 115–121, 2014.
[17] C. Zhao, W. S. Lee, and D. He, "Immature green citrus detection based on colour feature and sum of absolute transformed difference (SATD) using colour images in the citrus grove," *Comput. Electron. Agricult.*, vol. 124, pp. 243–253, Jun. 2016.
[18] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. 9, no. 1, pp. 62–66, Jan. 1979.
[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[20] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[22] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "DeepFruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, 2016.
[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
[25] Q.-C. Mao, H.-M. Sun, Y.-B. Liu, and R.-S. Jia, "Mini-YOLOv3: Real-time object detector for embedded applications," *IEEE Access*, vol. 7, pp. 133529–133538, 2019.
[26] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861
[27] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767
[28] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," 2016, *arXiv:1612.08242*. [Online]. Available: http://arxiv.org/abs/1612.08242
[29] S. Bargoti and J. Underwood, "Deep fruit detection in orchards," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May/Jun. 2017, pp. 3626–3633.
[30] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.
[31] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Oct. 2017, pp. 2961–2969.
[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2980–2988.
[33] P. Zhang, Y. Zhong, and X. Li, "SlimYOLOv3: Narrower, faster and better for real-time UAV applications," 2019, *arXiv:1907.11093*. [Online]. Available: http://arxiv.org/abs/1907.11093



ZHI-FENG XU was born in Shandong, China, in 1994. He received the B.S. degree from the Shandong University of Technology, China, in 2018. He is currently pursuing the M.S. degree with the Shandong University of Science and Technology. His research interests include image processing and deep learning

RUI-SHENG JIA is currently a Professor with the College of Computer Science and Engineering, Shandong University of Science and Technology, China. He has more than 30 first-author publications. He has more than 25 coauthor publications. His research interests include artificial intelligence, big data processing, information fusion, micro seismic monitoring, and inversion.



YAN-BO LIU was born in Shandong, China, in 1996. He received the B.S. degree from Liaocheng University, China, in 2018. He is currently pursuing the M.S. degree with the Shandong University of Science and Technology. His research interests include image processing and deep learning.

CHAO-YUE ZHAO was born in Shandong, China, in 1996. She received the B.S. degree from Shandong Women's University, China, in 2018. She is currently pursuing the M.S. degree with the Shandong University of Science and Technology. Her research interests include image processing and deep learning.

HONG-MEI SUN received the B.S. and M.S. degree in computer science from the Shandong University of Science and Technology, China, in 1995 and 2005, respectively. She is currently a Lecturer with the College of Computer Science and Engineering, Shandong University of Science and Technology, China. She is the Leader of the Key Research and Development Projects of Shandong Province, China. She has four first-author publications and has five coauthor publications. Her research interests include micro seismic monitoring technology and software engineering.