Journal of King Saud University – Computer and Information Sciences 35 (2023) 101598

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Mask Positioner: An effective segmentation algorithm for green fruit in complex environment

Yuqi Lu [a], Ze Ji [b], Liangliang Yang [c], Weikuan Jia [a,e,*]

[a] *School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China*
[b] *School of Engineering, Cardiff University, Cardiff CF24 3AA, UK*
[c] *Faculty of Engineering, Kitami Institute of Technology, Koen-cho 165, Kitami, Hokkaido 090-8507, Japan*
[e] *Key Laboratory of Facility Agriculture Measurement and Control Technology and Equipment of Machinery Industry, Zhenjiang 212013, China*

ARTICLE INFO

Article history:
Received 31 December 2022
Revised 22 May 2023
Accepted 25 May 2023
Available online 1 June 2023

Keywords:
Mask Positioner
Complex background
Edge segmentation
Green fruit

ABSTRACT

In order to enable intelligent orchard management and the application of harvesting robots, it is necessary to improve the accuracy of computer vision technology for green fruit segmentation in complex orchard environments. However, existing segmentation algorithms are unable to generate precise fruit masks in such environments. This paper proposes a novel and efficient segmentation algorithm called Mask Positioner for accurate fruit segmentation. The Mask Positioner applies a layer-by-layer filtering approach to refine feature maps generated by the detail refinement network, resulting in a refined mask. The selected pixels are then input to the order decoder to determine their relevance to the fruit region. Finally, the determined pixels are used to generate the final mask, resulting in accurate and efficient fruit segmentation. Mask Positioner is verified by a green persimmon dataset made for the complex background. The experimental results show that the segmentation accuracy of Mask Positioner reaches 67.4%, and the detection accuracy reaches 69.1%. For small fruits, its detection and segmentation accuracy are at least 1.0 and 3.2 percentage points higher than other algorithms. Additionally, the generalization ability of the algorithm is verified using a green apple dataset. Experiments show that it does well in the green fruit segmentation.

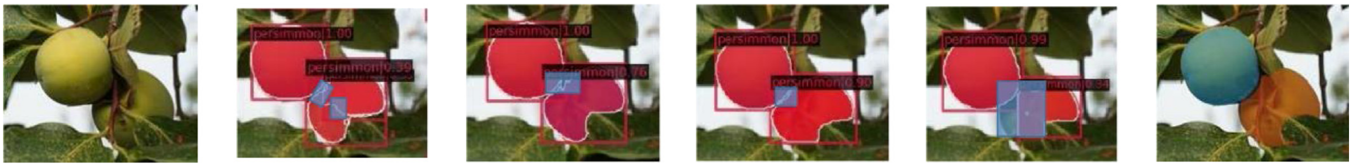
1. Introduction

In the field of orchard management, scientific yield prediction remains a challenging task. In which, accurate irrigation and fertilization are crucial for optimal fruit growth, while intelligent robots can replace human labor in fruit picking, saving resources and improving efficiency. Additionally, yield prediction can guide orchard thinning and allow farmers to plan fruit storage and sales in advance. However, these tasks often rely on expensive and expert-dependent methods, making them difficult for small-scale orchards to implement. This study aims to address these challenges using advanced computer vision techniques. Recent developments in deep learning have greatly improved the accuracy of object detection and segmentation, leading to successful applications in agriculture (Sun et al., 2022; Tang et al., 2023), including fruit detection (Inkyu et al., 2016; Bargoti et al., 2017a), pest identification (Patel et al., 2011; Ebrahimi et al., 2017; Ngugi et al., 2021), and autonomous field operations (Yang et al., 2021).

Although segmentation has been widely applied in field of agriculture, there is still a challenge in accurately segmenting green fruits, particularly green fruits with small size characteristics. Segmentation of small green fruits in the orchard is a challenging task due to the complex and dynamic environment, as fruits are often obscured by leaves and overlap each other. Moreover, the changing daytime light conditions and artificial illumination at night further complicate the task of accurate segmentation. These factors have a significant impact on the object segmentation technology that is crucial for orchard intelligence, making it difficult to achieve high segmentation accuracy. From Fig. 1, it is evident that some algorithms such as GCNet or HRNet often have areas with inaccurate segmentation. These areas typically correspond to irregular edges of the fruit, which are obscured by branches and leaves. These edges are represented by white lines in Fig. 1. Furthermore, when two fruits overlap each other, it can be difficult for the mask to accurately cover and distinguish the fruit boundary, as depicted by blue boxes.

[*] Corresponding author at: School of Information Science and Engineering, Shandong Normal University, Jinan 250358, China.
E-mail address: wkjia@sdnu.edu.cn (W. Jia).

https://doi.org/10.1016/j.jksuci.2023.101598
1319-1578/© 2023 The Author(s). Published by Elsevier B.V. on behalf of King Saud University.



Original image Mask RCNN Mask RCNN+Cascade GCNet HRNet Mask Positioner

Fig. 1. The edges of fruits are not clearly distinguishable when they are overlapped or covered by leaves. Besides, the generated masks for the covered parts may not be smooth enough, as indicated by the blue boxes in the image. However, the Mask Positioner can accurately distinguish between two target fruits and generate smooth and accurate masks for both of them.

To address the issue of imprecise segmentation for small green fruits, this study proposes an efficient segmentation algorithm called Mask Positioner. It consists of four main components: backbone, detail refinement network (DRN), pixel filtering, and order decoder. The Resnet50 (He et al., 2016) is utilized as the backbone network for feature extraction of input images in this study. Similar to the classic Mask RCNN (He et al., 2017), the region proposal network (RPN) generates anchor boxes, which are refined by the ROI Align operation in the next step. Differently, the study incorporates a detail refinement network to fully integrate features, thereby avoiding the loss of feature information of small green fruits. Moreover, Mask Positioner extracts the pixels of the area with inaccurate segmentation and inputs them into the order decoder for judgment. Mask Positioner aims to refine the mask of hard-to-segment areas such as fruit edges. The experimental results show that the segmentation accuracy of small-sized fruits is significantly improved, and the generalization of the proposed method on green fruit is excellent.

Overall, this paper includes main contributions as follows:

1) An effective object segmentation algorithm, Mask Positioner, is proposed in this study. It improves segmentation accuracy by three parts of detail refinement network, filtering pixels and order decoder. The algorithm filters pixels in a layered manner from the feature maps fused by DRN and sends them to the order decoder for determining whether they belong to the fruit region.
2) This study made an immature green persimmon dataset to simulate the real orchard environment and test the effectiveness of Mask Positioner. Our dataset includes small green fruits with variable illumination, green color, and fruit blockage and overlap. These characteristics effectively test the algorithm's performance. A generalization experiment was conducted on a green apple dataset created by ourselves, which proved that Mask Positioner can be applied to other green fruits.
3) Mask Positioner outperforms other algorithms in terms of small-sized fruit segmentation and is also not significantly affected by lighting conditions. The segmentation accuracy of small fruit in persimmon data achieved by Mask Positioner is 67.4%, which is at least 1.0 percentage points higher than other algorithms. Similar results are observed for green apple data, demonstrating its effectiveness in handling complex orchard environments and meeting the demands of computer vision.

2. Literature review

In order to estimate the feasibility of Mask Positioner in advance, much literature has been consulted. Up to now, various studies have been devoted to solving the problem of orchard yield measurement and counting and so on. It mainly includes methods based on machine learning and deep learning. This section reviews the application of various algorithms and summarize their characteristics and performances.

2.1. Technologies based on machine learning

Methods based on machine learning developed relatively early. Dorj et al. (Dorj et al., 2017) developed an algorithm that utilized color features and the watershed algorithm to detect and count citrus fruit. The algorithm was able to achieve a high correlation coefficient of 0.93 between the citrus counting algorithm and human observation. This demonstrated its ability to produce accurate results. In the same year, In 2017, Qureshi (Qureshi et al., 2017) proposed a mango detection algorithm based on texture dense segmentation and shape for mango tree crown. While this method achieved good results, it had limitations in terms of lighting conditions and was found to work best with images taken at night under artificial lighting. Yasar and Akdemir (Yasar & Akdemir, 2017) employed Artificial Neural Networks (ANNs) to extract color features from HSV color space for detecting orange fruits. Their detection algorithm achieved an accuracy of 89.8%. Additionally, Mehta (Mehta et al., 2017) proposed a kernel density clustering method for the segmentation of green fruit. This method used a double sorting algorithm to automatically select the center of the cluster, effectively reducing the calculation cost and achieving almost complete fruit recognition.

It can be seen from the above studies that algorithms based on machine learning mainly rely on the surface features of fruit such as color, texture and size. Support vector machines (SVM) classifier is often used to analyze these features. While machine learning has shown good performance in fruit detection and segmentation, it still faces challenges in accurately segmenting fruits with overlapping and occlusion. Additionally, it has limited adaptability to the orchard environment and is not suitable for large-scale orchard production. Therefore, there is a need for more accurate and robust algorithms with stronger learning ability to detect and segment fruit in orchards.

2.2. Technologies based on deep learning

In the past few years, deep learning technology has made significant advancements, particularly in the field of computer vision. One of the most notable achievements is the development of end-to-end detection mode, which has given rise to various neural network algorithms that can be applied to diverse contexts. Bargoti and Underwood (Bargoti & Underwood, 2017b) proposed an image processing framework for fruit detection and counting using orchard image data. The framework combined two methods, Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN), for apple image segmentation. The performance of the segmentation model was evaluated using the F1 score, which is the harmonic mean of accuracy and recall, and was improved to 0.839. Wang et al. (Wang et al., 2017) proposed an algorithm to overcome the effect of illumination changes on computer vision in natural environments by applying improved wavelet transform to fruit images for surface illumination normalization. They also proposed a robust fruit segmentation algorithm by combining image enhancement algorithms for illumination changes in the vision system. The algorithm showed strong robustness against illumination changes and was able to accurately segment fruits of different colors. Jia (Jia et al., 2022) developed an advanced apple detection algorithm that is both fast and accurate. The algorithm was based on Foveabox (Kong et al., 2020) and used EfficientnetV2 (Tan & Le, 2021) as the backbone network, which effectively fused features of different scales. Furthermore, the algorithm utilized an adaptive training sample selection method to distinguish positive and negative samples. This approach resulted in improved detection accuracy, even with a low parameter quantity.

Based on the above studies, it is evident that deep learning-based algorithms have been extensively used in fruit detection and segmentation tasks (Gongal et al., 2015; Koirala et al., 2019). The inherent features of deep learning make these algorithms more robust and powerful. Nevertheless, there is still a lack of accurate segmentation of green fruits, especially green fruits with small size characteristics, which is also the core issue that this paper aims to address.

3. Materials and methods

Section 3 introduces the dataset, including persimmons and apples, as well as the overall structure of the Mask Positioner.

3.1. Dataset

In order to prove the advantage of Mask Positioner for complex orchard environment, immature green persimmon fruit is selected as the research object of this study. Because of its characteristics of small size, green color and occlusion, it is suitable for testing the algorithm performance. A persimmon dataset was made on this basis. And the images were taken in the back mountain of Shandong Normal University (Changqing Lake Campus). The shooting tool was Canon EOS 80D SLR camera with built-in CMOS image sensor. Images were saved as JPG format, 24 bit color. Types of images in the dataset included overlapped and blocked fruits under different lighting conditions (7):00–17:00 in the daytime and LED lights at night). These diversities of images fully simulate the real situation of the orchard in the experiment and restore the working environment of computer vision. Representative images are shown in Fig. 2.

LabelMe was used to process captured images. The image size was unified to 400 × 600 pixels before labeling. Points were used to mark the target contour. The closed part marked was the foreground, the rest was the background. There was only persimmon in the foreground. The object coordinates and information of each image were saved in the corresponding JSON file. After eliminating images without fruits, labeled images were divided into training set and verification set by a ratio of 7:3. Finally the persimmon dataset had 563 images, including 396 images in the training set and 167 images in the validation set.

In order to ensure the effectiveness of Mask Positioner on other green fruits, another dataset was made. The green apple dataset was made to be the object of the generalization experiment. We used Canon EOS 80D SLR camera to collect apple images in Longwang Mountain Apple Production Base (Agricultural Information Technology Experimental Base of Shandong Normal University), Fushan District, Yantai, Shandong Province. Similar to the persimmon images, we shot as many apple images in different situations as possible, including overlapped, blocked fruits during daytime and night. They were made into apple dataset of COCO format in the same way. Finally, there are 953 images in the training set, and 408 images in the validation set. The production of apple dataset also uses the tool of LabelMe like persimmon.

3.2. Mask Positioner

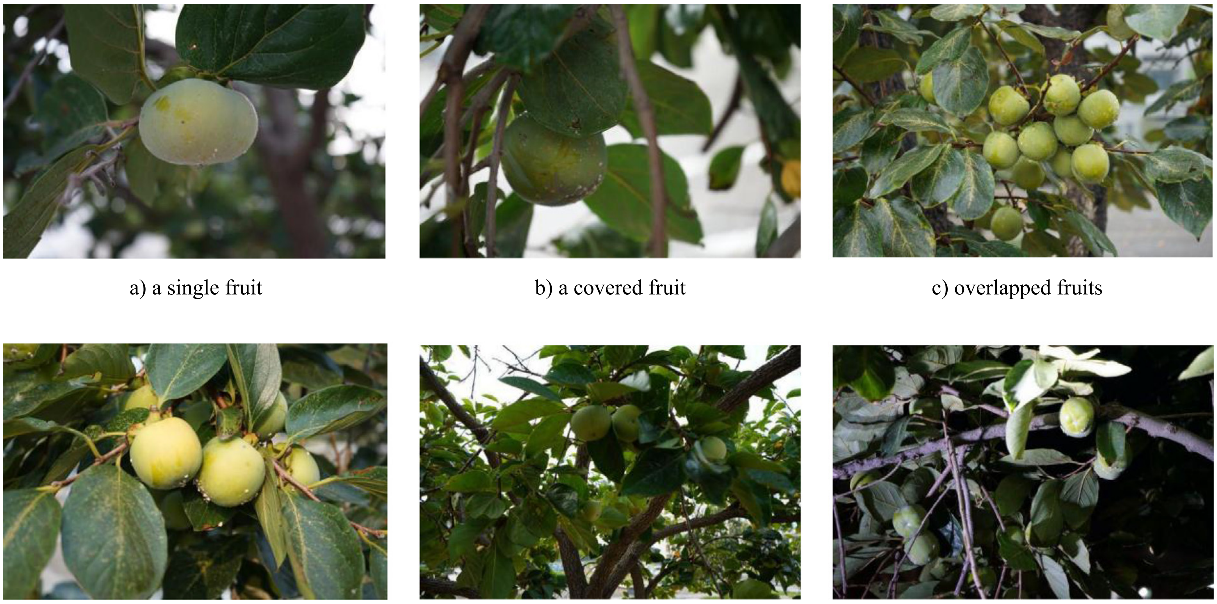
Mask Positioner is an effective algorithm designed for accurate segmentation of green fruits in orchards. Its architecture is illustrated in Fig. 3, which includes backbone, detail refinement network, process for filtering pixels, and order decoder. Mask Positioner employs ResNet50 as the backbone, with RPN used to generate anchor boxes. Additionally, the detail refinement network is used to fuse features extracted from the backbone, with a unique upward fusion of adjacent feature maps through convolution after conventional vertical and horizontal sampling. This approach ensures that semantic and location information is fully integrated, making it beneficial for small fruits. The final feature maps used to generate the anchor boxes are expressed in N2-N4. Moreover, the Mask Positioner filters pixels on these three feature maps, with blurred identity layers being filtered layer by layer, as represented by the pink boxes in Fig. 3. This process is explained in detail in Section 2.2. Finally, the order decoder is used to determine whether the filtered pixels belong to the fruit area. It has a transformer structure with branch attentions, and the pixels identified by the decoder form the final mask, represented by the red box in Fig. 3.

3.2.1. Detail refinement network

Given a persimmon image, it is firstly extracted by backbone. The function of detail refinement network is to fully fuse the extracted features. This step of refinement integrates the semantic and location information in different layers, which is the basis of accurate segmentation. The feature maps C2-C5 of the backbone are input to the DRN, and firstly set as 256 channels by a 1 × 1 convolution layer. The next step is to use the method of nearest interpolation for up-sampling, whose principle is shown in Fig. 4. It is to make the transformed pixel equal to the original nearest pixel. This operation can enlarge the feature map twice, while retaining the semantic information of the high-level feature map to the maximum extent. And the enlarged map is fused with the low-level map of the same size. But due to the loss of up-sampling process, some region feature may not coincide well during the feature map fusion. At this time, a convolution layer with the size of 3 × 3 is used to restore the features. It not only ensures the stability of features, but also eliminates the aliasing effect caused by up-sampling. So that it obtains the mid-maps with both semantic and spatial information, denoted by P2-P5 respectively.

This is usually the end of feature fusion. However, considering the difficulty of green fruit segmentation caused by complex environment, this study continues to conduct further fusion from bottom to top. A convolution layer with the size of 3 × 3 is used to further extract features and fuse them with the adjacent upper map, so that the semantic and spatial details are refined well. In order to ensure the smooth fusion, the stride of this convolution layer is set as 2 to adjust the two feature maps to the same size. Similarly, a 3 × 3 convolution layer is used to eliminate the aliasing effect caused by the maps fusion. The feature maps processed by DRN simultaneously contains the rich semantic information of the top-level feature map and the position information of the low-level feature map. The feature information is helpful for the position embedding of the subsequent sequences. And this full of feature fusion ensures the information integrity of small target fruits.

Correspondingly, the optimization effect of DRN from the perspective of detection is verified on the persimmon dataset. It is generally known that feature pyramid networks (Lin et al., 2017)



a) a single fruit

b) a covered fruit

c) overlapped fruits

d) front-lighting fruits

e) back-lighting fruits

f) night-time fruits

Fig. 2. Fruit images in different states.

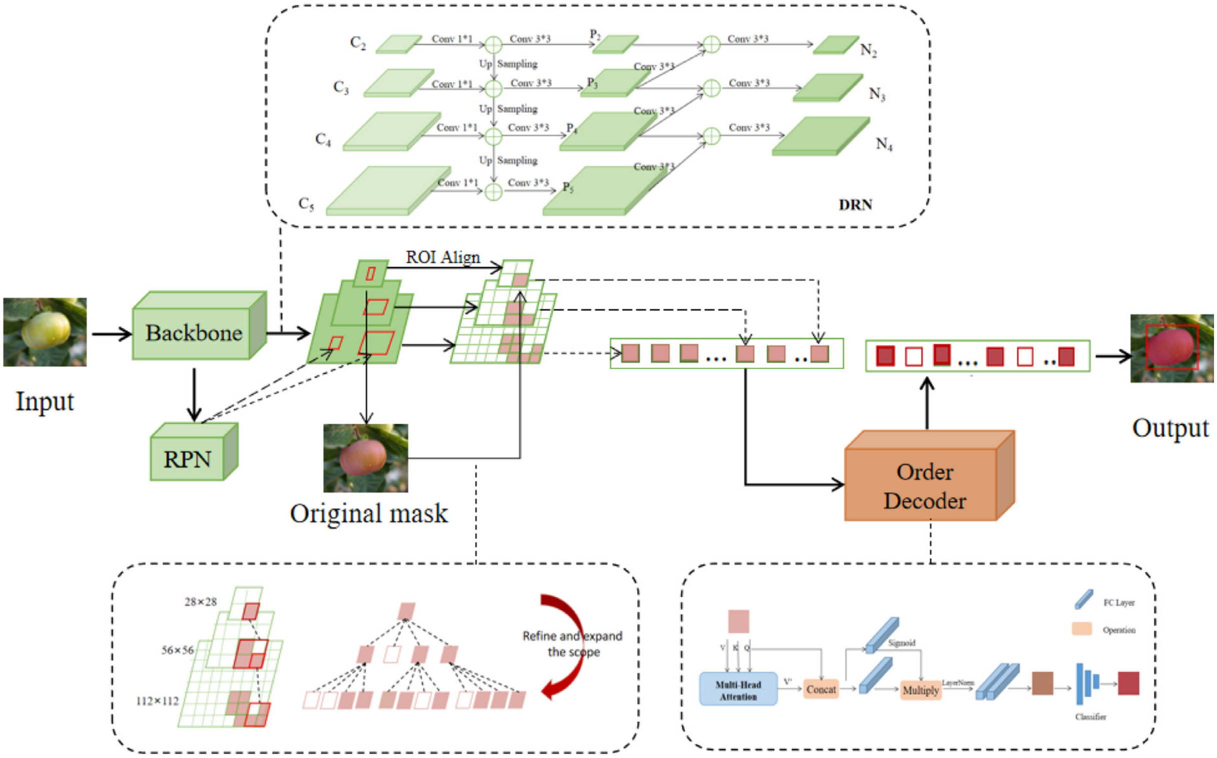


Fig. 3. The structure of Mask Positioner. It shows the internal structure of DRN and Order Decoder in detail, and draw the detection process of pixels that need to be positioned.

(FPN) has been widely used in various algorithms. Owing to the similarity of their function, this study compares them by experiments to prove the advantage of DRN. The results are shown in the Table 1. The effect on Mask Positioner is firstly verified. It can be seen that detection accuracy of Mask Positioner with DRN reaches 69.1%, 1.7 percentage points higher than that with FPN. In order to further prove its effectiveness, the classical algorithm Faster RCNN is also selected for ablation experiments. It can be seen that the accuracy of Faster RCNN with DRN improves 2.2 percentage points than the original.

3.2.2. Pixels that need to be positioned

In the filed of computer version, researchers are committed to generating more accurate masks. However, the sampling operation in the process of feature extraction leads to the necessary information loss. And the loss is usually equal to the positions that are

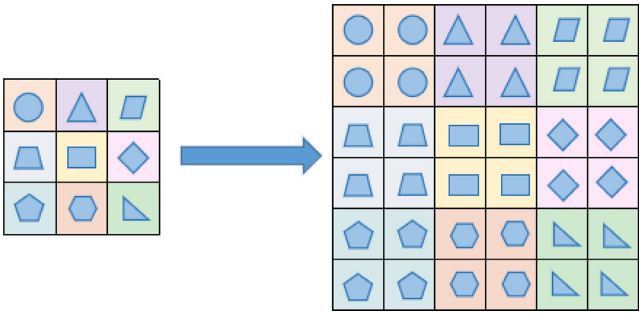


Fig. 4. Schematic diagram of the nearest interpolation method.

difficult to be segmented, which means unclear or discontinuous boundary areas in Fig. 1. For example, the edge areas covered by branches and leaves are irregular shadows in orchard, which makes it difficult to segment these areas accurately. In order to accurately segment these areas, Mask Positioner specially filters out pixels corresponding to the regions in feature maps, and processes them more finely. This process is specifically to determine whether these pixels belong to the target fruits. In this way, the edge parts of the target fruit can be accurately segmented, solving the problem of occlusion fruits in the orchard.

Two factors are required for filtering pixels that need to be positioned: the original mask generated from the top-level feature map, and the three-level feature maps generated by DRN. The core idea is to refine the original mask according to feature maps of each layer through up-sampling. It is well known that the top-level feature map contains the richest semantic information. Therefore, although the mask it produces is not accurate enough, it can cover the entire fruit range. Mask Positioner refines pixels layer by layer from the original mask, referring to the feature maps. The feature maps with more complete information are referred to find out the pixels that need to be positioned from top to bottom. Firstly, the original mask and the top-level feature map are concatenated. For the small size 28 × 28 feature map and the rough mask at this time, four convolutions of size 3 × 3 are used to fuse and adjust the two layers to overcome the low overlap of feature information. Then, the pixels belonging to the fruit are obtained through the binary classifier, namely the first mask layer. In order to combine more features to refine the mask, the first mask layer is up sampled for hierarchical connection. This means that only the lower pixel corresponding to the selected pixel in the upper feature map can be further filtered. It limits the scope of filtering, only filtering pixels with discriminative significance. This process is shown on the left side of Fig. 5. The mask size after up sampling is 56 × 56. After fusing with the second-layer feature map (represented by green part in the Fig. 5) in a similar way, the second mask layer is obtained after two convolutions.

The refinement process for the feature map described above is illustrated in Fig. 5. The same principle is applied to the other two layers. As the mask size continuously doubles, the degree of overlap between regional features increases, making it possible to use only one convolution after the fusion operation of the last layer. As a result, three mask layers with different sizes are generated. This approach enables the next mask to be confined within the sampling range of the adjacent mask, without overly limiting the screening area. The top feature map undergoes pixel filtering first, resulting in a refined but rough mask that carries complete feature information. The subsequent thinning process confirms the detailed information, such as edges. This involves up-sampling the range corresponding to the pixels, which expands the area being refined. To ensure that the features have multi-scale characteristics, the pixels from all of the mask layers are extracted and encoded into a sequence. This sequence is then input into the order decoder for processing and classification.

3.2.3. Order decoder

To enrich the pixel features before entering the order decoder, both the coarse and fine feature maps are utilized. The coarse feature map provides context and semantic information, while the fine feature map restores edge details of the target. In addition to the feature information extracted by DRN, the semantic information from the original mask is also fused using a fully connected layer to fix the feature dimension, followed by position information embedding through addition operations. Despite the nodes being sequenced, the order decoder can still restore relative feature positions through location information. Its structure, depicted in Fig. 6, is based on the transformer architecture and comprises multi-head attention and a Fully Convolutional Network (Long et al., 2015) (FCN).

When the input feature X enters the decoder, it is firstly projected by the weight matrix W as $(Q, K, V) = (XW^q, XW^k, XW^v)$. Q and K represent the information and vector to be queried respectively, and V represents the value obtained from the query. The expression of self attention is shown in the Eq. (1):

$$\mathrm{Attention}(Q, K, V) = \mathrm{Softmax}(\frac{Q \times K^T}{\sqrt{d}\,K}) \times V \tag{1}$$

Accordingly, the expression of each self attention head is shown in the Eq. (2):

$$\mathrm{Head}_i = \mathrm{Attention}(QW_i^q, KW_i^k, VW_i^v) \tag{2}$$

where i represents the number of self attention heads, which is set to 4 in the experiment. For multi-head attention, the output of each head is concatenated together, and then multiplied by the matrix W^0 for linear transformation to obtain the final output result. It is shown in the Eq. (3):

$$V' = \mathrm{Concat}(\mathrm{Head}_1, ..., \mathrm{Head}_i) \times W^0 \tag{3}$$

Table 1
Detection comparison of DRN and FPN on the persimmon dataset.

	AP % (Average Precision)	AP_{50} % (IOU threshold is 0.5)	AP_{75} % (IOU threshold is 0.75)
Mask Positioner + FPN	67.4	83.9	76.9
Mask Positioner + DRN	69.1	84.2	78.4
Faster RCNN (Ren et al., 2015) + FPN	63.6	91.0	80.4
Faster RCNN (Ren et al., 2015) + DRN	65.8	91.7	79.3

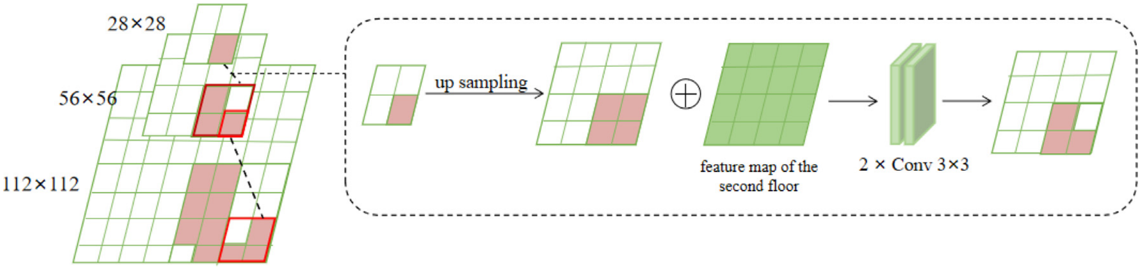


Fig. 5. The mask thinning process diagram of the second layer.

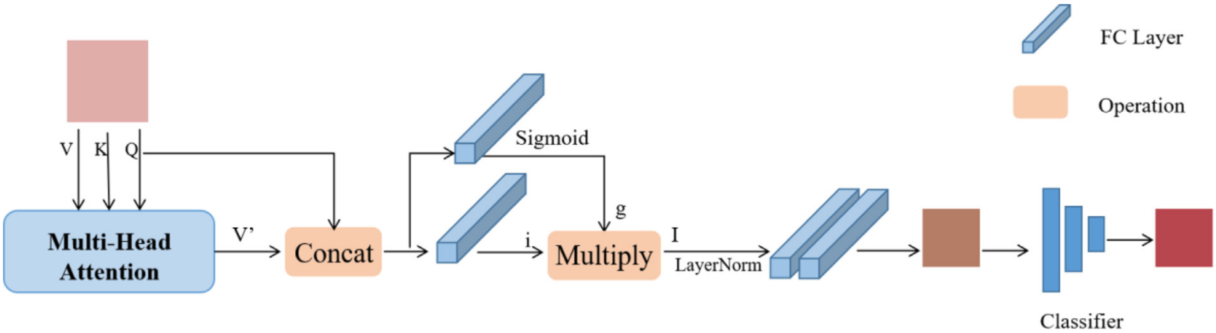


Fig. 6. Structure of the Order Decoder.

After the traditional multi-head attention, Order Decoder also constructs two branch structures, and the results of the branch are represented by i and g. For multi-head attention, these two branches help to establish the relationship between different attention heads, and filter useful attention results. It can also avoid the false similarity of multi-head attention output when Q and K/V are completely unrelated. In this process, the full connection layer is used, which is based on attention results and context information. The principle is shown in the Eq. (4) and Eq. (5):

$$i = W_q^i Q + W_v^i V' + b^i \tag{4}$$

$$g = \mathrm{Sigmoid}(W_q^g Q + W_v^g V' + b^g) \tag{5}$$

In which, W_q^i, W_v^i, W_q^g, $W_v^g \in R^{D\times D}$, represent the weight parameters of the full connection layer. Though they are different, they are all the weight matrix corresponding to their respective vectors. And b^i, $b^g \in R^D$, represent the bias parameters of full connection layer. D is the dimension of variables, and V' is the results of the multi-head attention. Q represents the query vector projected after the feature is input into the decoder. Finally, the element multiplication is used to get the final result of the two branches, which is shown in the Eq. (6):

$$I = i \odot g \tag{6}$$

Next, the LayerNorm is used for horizontal normalization. Considering that the attention mechanism does not fit the complex process well enough, two full connection layers are added to enhance the decoder's ability. The obtained pixels are put into a simple MLP classifier to determine whether they belong to the target fruit.

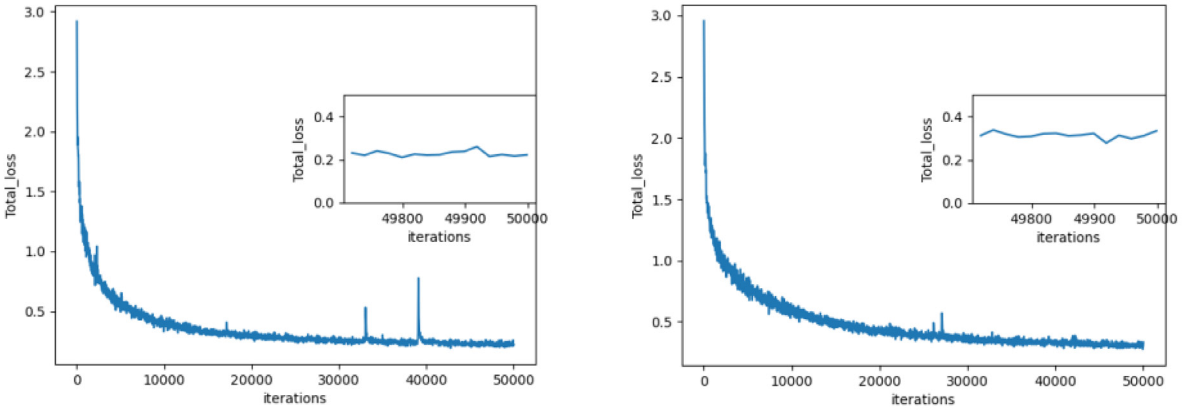
3.2.4. Loss

The loss function of Mask Positioner is composed of four parts: the loss of detection, the loss of original mask, the loss of filtering pixels, and the loss of the final refined area. The loss function is as shown in the Eq. (7):

$$loss = f_{det\,ection} + f_{original} + f_{pixels} + f_{mask} \tag{7}$$

The generation of bounding boxes is similar to other common detection methods: the anchor boxes given by RPN are aggregated by ROI Align operation to adjust the region of interest to the corresponding feature position. The loss of detection represented by $f_{det\,ection}$ includes two vectors, namely classification and regression. The process of classification uses the cross entropy loss of two classification, calculating the logarithm loss for each anchor, and it divides the total number of anchors. The SmoothL1Loss is used in the process of regression to smooth the error when the difference between the predicted value and the real value approaches zero, which can prevent the problem of gradient explosion to a certain extent. The $f_{original}$ represents the loss of the original mask. When filtering pixels, the binary cross entropy loss of the weighted average method namely f_{pixels} is used to help detecting the pixels that need to be located. The average absolute error f_{mask} is used to represent the loss of whether the refined region label judgment is consistent with the real label. The trend of total_loss during the training process have been visualized as shown in the Fig. 7.

Firstly, it can be seen that during the 40,000 iterations, there were two significant fluctuations in the training process on the persimmon dataset between the 30,000th and 40,000th iterations. Therefore, the number of iterations is increased to 50,000. From the Fig. 7, the curve of total_loss tends to be flat and almost linear from the 40 K to 50 K training process, and no significant fluctuations are observed. To further determine the algorithm's convergence, the total_loss of the last 300 iterations is zoomed. It can be seen that the total_loss changes are all within 0.1, and there is no continued downward trend. Besides, during the last 10,000 iterations, total_loss did not show any significant fluctuations and was almost linear, indicating that the algorithm had fully converged by this point.



a) Total_loss of persimmon.

b) Total_loss of apple.

Fig. 7. Total_loss with the last 300 iterations in training process.

4. Results

This section explains the training details and experimental data, including comparative experiments on persimmon and generalization experiments on apple.

4.1. Implementation details

The experiment is conducted on a server equipped with Ubuntu 16.04 operating system. It is also equipped with four GTX A30 graphics cards and V11.4 CUDA. In order to accelerate the convergence speed and reduce the training time, the backbone is trained in advance with the ImageNet dataset. The learning rate is set to 0.02 at the pre-training stage. In the process of formal training, the initial learning rate is set to 0.075, and the iterations is set to 50 k due to the trend of the total_loss, which is enough and reasonable for the training process. The learning rate decreases three times during the 50 K iterations, which are 0.02, 0.002 and 0.0002 respectively. Experiments set four images per batch in the experiment.

Besides, the optimizer plays a crucial role in training deep learning models by adaptively adjusting the learning rate and updating parameters to improve model performance and convergence speed. So an optimizer is used during the model training process to save training time and make algorithm's convergence more reasonable. However, it is worth noticing that our aim is not to improve the accuracy of Mask Positioner through the optimizer, as this would affect the performance validation of the algorithm itself. Therefore, during our training process, the optimizer is only used to save time and prevent the algorithm from overfitting. Currently, commonly used optimizers include Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam), and Adaptive Gradient (Adagrad), among others. Among them, SGD is one of the most basic optimizers, with only one hyperparameter for learning rate, which makes it very convenient for parameter setting. At the same time, SGD is more likely to enable the model to achieve good generalization performance compared to other optimizers. Therefore, it is very friendly to the training of the Mask Positioner algorithm for green fruit segmentation.

The SGD optimizer used the momentum and weight decay functionalities mentioned in the original manuscript, which can effectively prevent overfitting. They were set to 0.9 and 0.0001, respectively. Momentum refers to weighting the previous gradient direction and the current gradient direction during gradient descent to reduce the variance of gradient updates and achieve faster convergence. Weight decay refers to adding an L2 regularization term to the loss function to penalize large weight values in the model and prevent overfitting.

4.2. Data augment

The dataset is enhanced through the application of two data augmentation methods prior to training: translation transformation and rotation, as illustrated in Fig. 8.

In the translation transformation, the original image is translated to the lower right and upper left directions, with both horizontal and vertical translation pixel values set to 50. The translated result is shown in Fig. 8b), where the part that is translated out is filled with black color. In the rotation, the image is set to rotate twice at random angles, as shown in Fig. 8c). When the image is rotated, it is first scaled by a factor of 0.8 and then cropped to remove the parts outside the image range. The scaling ensures that the target fruit is retained as much as possible and is not cropped. The remaining blank areas are also filled with black color.

Data augmentation generates new training samples by applying a series of random transformations to the original data, thereby increasing the diversity and quantity of the dataset. This can prevent the algorithm from relying too much on specific samples, thus reducing the risk of overfitting. In addition, data augmentation can enrich image data, allowing the Mask Positioner to learn fruit features from more samples. After the images are augmented, the features that the Mask Positioner can learn become more diverse and extensive, making it more robust and better suited to different fruits. Therefore, data augmentation can improve the performance of the Mask Positioner, specifically its segmentation ability. In general, data augmentation is necessary for improving both the performance and the prevention of overfitting of the Mask Positioner.

4.3. Evaluation metrics

The Average Precision (AP) is selected as the evaluation index in the experiment. AP is the area under PR curve with Recall as the horizontal axis and Precision as the vertical axis. The calculation formula of AP is shown as Eq. (8), in which P is the Precision, R is the Recall. Both P and R are explained in detail in the next section. Other evaluation indicators used in the experiment also belong to the same type: AP_{50} is the measured value of AP when the IOU threshold is 0.5; AP_{75} is the AP measurement value when IOU is

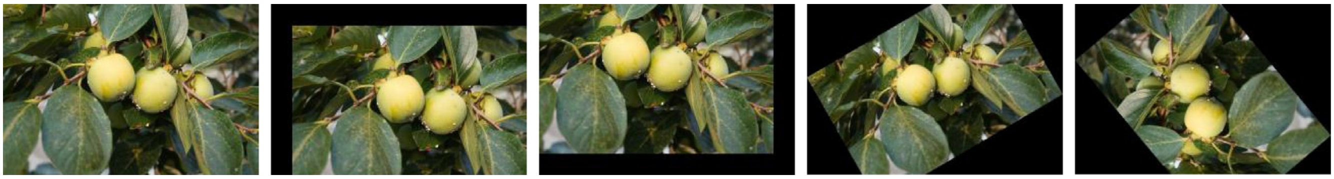

a) original b) translation transformation c) rotation

Fig. 8. Images of data augmentation.

0.75; AP_s, AP_m and AP_l represent AP measurement values of small, medium and large size objects respectively. The size specification method of the COCO dataset is used here due to the similarity of image size between two datasets. It is defined that fruits with area less than 32 × 32 are small-sized objects, fruits with area greater than 96 × 96 are large-sized objects, and fruits with area between them are medium-sized objects.

$$AP = \int_0^1 P(R)dR \tag{8}$$

In the definition of AP, P is represents the proportion of the number of predicted positive samples to the number of real positive samples; R represents the proportion of positive samples correctly predicted by the algorithm in the real positive samples. The calculation equations are shown in Eqs. (9) and (10), where TP represents the number of detection frames whose intersection to parallel ratio is greater than the set threshold; FP represents the number of detection frames whose intersection ratio is less than the set threshold, or the number of redundant detection frames generated under the same target; FN indicates the number of targets not detected.

$$\text{Precision} = \frac{TP}{TP + FP} \times 100\% \tag{9}$$

$$\text{Recall} = \frac{TP}{TP + FN} \times 100\% \tag{10}$$

4.4. Comparative experiments

In order to verify the advantage of Mask Positioner, it is compared with the currently popular and widely used algorithms. In order to conduct a clearer comparison, experiment results are placed in two tables to show the accuracy comparison of detection and segmentation respectively. The detection results are shown in Table 2. It can be seen that Mask Positioner has the highest detection accuracy, which reaches 69.1% on the persimmon dataset. This data is 4.9 and 3.6 percentage points higher than that of HRNet (Sun et al., 2019) and GCNet (Cao et al., 2019). And also 1.2 and 0.3 percentage points higher than that of Mask RCNN (He et al., 2017) and Mask RCNN with Cascade (Li et al., 2015) respectively. Although the detection accuracy of Mask Positioner is only a little higher than that of Mask RCNN improved by Cascade, it is better at detecting small fruits. Its detection accuracy of fruits has improved by at least 1.0 percentage point compared with other algorithms. This advantage enables Mask Positioner to deal with small immature fruits well.

The segmentation accuracy of Mask Positioner is shown in Table 3. It can be seen that it has the highest segmentation accuracy, with AP reaching 67.4% on persimmon dataset. Other algorithms also have their own performances. The backbone of GCNet is optimized through the attention mechanism, which mainly focuses on object detection. Therefore, it can be seen from Table 3 that its segmentation performance of green fruits is far inferior to other algorithms, only 60.2%. HRNet is an algorithm dedicated to segmentation. It can maintain high-resolution representation and achieve multi-scale fusion throughout the process, which is more beneficial to the segmentation function. Consequently, its segmentation effect is much better than that of detection, reaching 66.0%. Mask RCNN equipped with Cascade performs well on the fruit segmentation, second only to the Mask Positioner. The accuracy of Mask RCNN without Cascade support is only 64.6%, reducing 1.8 percentage points. While, Mask Positioner has the highest accuracy of 67.4%, at least 1.0 percentage point higher than other algorithms. This is due to the enhancement of DRN for feature fusion and effective filtering of pixels.

In order to prove the segmentation effect of Mask Positioner on small fruits, experiments deliberately compare the segmentation accuracy with different sizes, as shown in Table 4. It can be seen that algorithms have poor differences of segmentation accuracy for medium-sized persimmon fruits. However, the accuracy of Mask Positioner has been greatly improved for small-sized fruits, reaching 25.5%. This data is 3.2 percentage points higher than the best algorithm of Mask RCNN with Cascade. In contrast, the lowest precision of HRNet is only 12.6%, which is not suitable for small fruits at all. Experiments show that Mask Positioner can effectively deal with small fruit segmentation in complex environment of orchards.

4.5. Generalization experiments

In order to verify the segmentation effect of Mask Positioner on other fruits, this study conducted a generalization experiment. The same parameters as the above experiments are set and experiments are conducted on the same server. The experimental results are shown in the following tables. The detection accuracy of the algorithms on the apple dataset is shown in the Table 5. It can be seen that Mask Positioner has the highest detection accuracy, reaching 57.9%. It increased at least 2.2 percentage points, which is far higher than other algorithms.

Table 2
Detection performance comparison between algorithms on different sizes for persimmon.

Algorithms	AP %	AP_s%	AP_m %	AP_l %	References
HRNet	64.2	28.5	66.1	74.8	(Sun et al., 2019)
GCNet	65.5	17.4	56.6	55.5	(Cao et al., 2019)
Mask RCNN	67.9	24.4	69.4	85.8	(He et al., 2017)
Mask RCNN + Cascade	68.8	28.1	67.3	80.0	(Li et al., 2015)
Mask Positioner	69.1	29.5	70.4	86.5	

Table 3
Segmentation performance between algorithms for persimmon.

Algorithms	AP % (Average Precision)	AP_{50} % (IOU threshold is 0.5)	AP_{75} % (IOU threshold is 0.75)	References
HRNet	66.0	84.8	76.0	(Sun et al., 2019)
GCNet	60.2	84.6	66.6	(Cao et al., 2019)
Mask RCNN	64.6	85.8	73.4	(He et al., 2017)
Mask RCNN + Cascade	66.4	88.2	76.3	(Li et al., 2015)
Mask Positioner	67.4	84.2	78.2	

Table 4
Segmentation performance between algorithms on different sizes for persimmon.

Algorithms	AP_s%	AP_m %	AP_l %	References
HRNet	12.6	67.7	85.0	(Sun et al., 2019)
GCNet	15.4	64.3	67.3	(Cao et al., 2019)
Mask RCNN	19.9	69.8	87.1	(He et al., 2017)
Mask RCNN + Cascade	22.3	68.3	81.5	(Li et al., 2015)
Mask Positioner	25.5	68.2	88.6	

Table 5
Detection performance between algorithms for apple.

Algorithms	AP % (Average Precision)	AP_{50} % (IOU threshold is 0.5)	AP_{75} % (IOU threshold is 0.75)	References
HRNet	52.3	81.7	57.3	(Sun et al., 2019)
GCNet	53.7	82.2	59.0	(Cao et al., 2019)
Mask RCNN	53.1	81.7	57.3	(He et al., 2017)
Mask RCNN + Cascade	55.7	82.0	59.8	(Li et al., 2015)
Mask Positioner	57.9	80.8	65.1	

Table 6
Segmentation performance between algorithms for apple.

Algorithms	AP % (Average Precision)	AP_{50} % (IOU threshold is 0.5)	AP_{75} % (IOU threshold is 0.75)	References
HRNet	50.4	80.0	53.9	(Sun et al., 2019)
GCNet	51.9	80.4	56.1	(Cao et al., 2019)
Mask RCNN	51.8	80.1	54.7	(He et al., 2017)
Mask RCNN + Cascade	50.3	79.3	52.6	(Li et al., 2015)
Mask Positioner	53.5	79.8	59.5	

Table 7
Detection performance between algorithms on different sizes for apple.

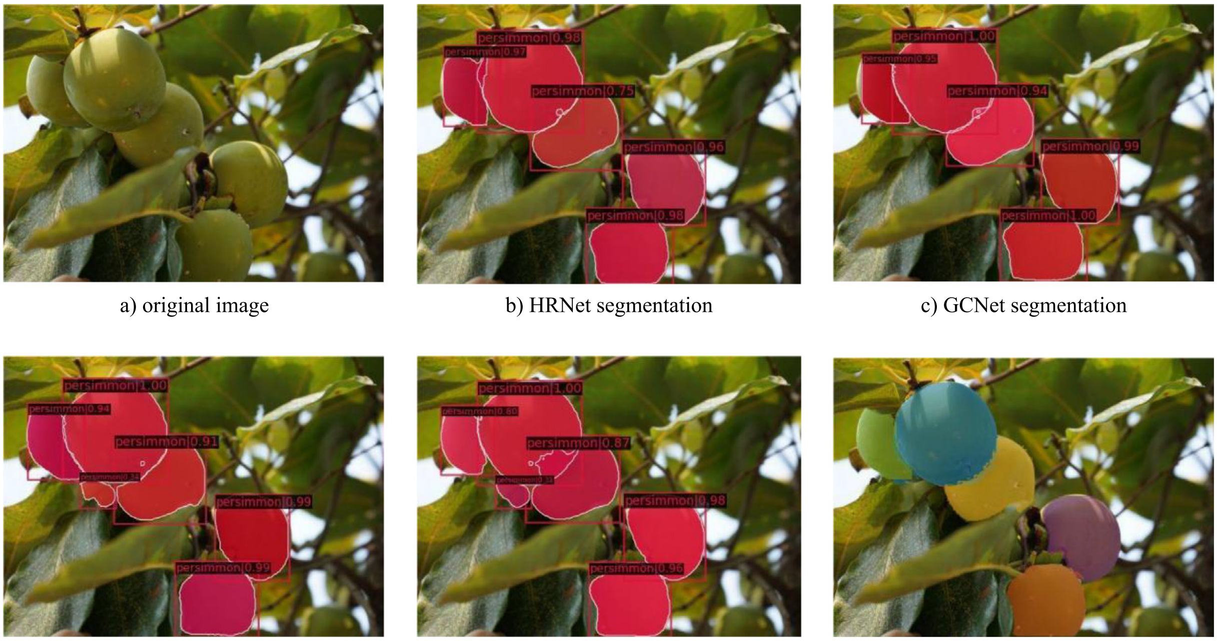
Algorithms	AP_s%	AP_m %	AP_l %	References
HRNet	39.6	60.1	75.0	(Sun et al., 2019)
GCNet	39.3	60.5	76.7	(Cao et al., 2019)
Mask RCNN	38.2	60.2	78.0	(He et al., 2017)
Mask RCNN + Cascade	38.1	63.8	85.2	(Li et al., 2015)
Mask Positioner	40.2	65.6	84.8	

Table 8
Segmentation performance between algorithms on different sizes for apple.

Algorithms	AP_s%	AP_m %	AP_l %	References
HRNet	33.9	56.6	78.2	(Sun et al., 2019)
GCNet	34.3	59.5	80.2	(Cao et al., 2019)
Mask RCNN	33.8	58.2	81.1	(He et al., 2017)
Mask RCNN + Cascade	30.7	56.9	82.8	(Li et al., 2015)
Mask Positioner	34.5	60.0	83.3	

The segmentation accuracy of algorithms is shown in the Table 6. It can be seen that our algorithm still maintains the highest accuracy, reaching 53.5%. This is 1.6–3.2 percentage points higher than other algorithms, which means its advantages.

Moreover, this study still compare the detection and segmentation effect for different sizes, and the results are shown in the following tables. It can be seen from Table 7 that Mask Positioner still maintains the highest detection accuracy of small-sized green



a) original image

b) HRNet segmentation

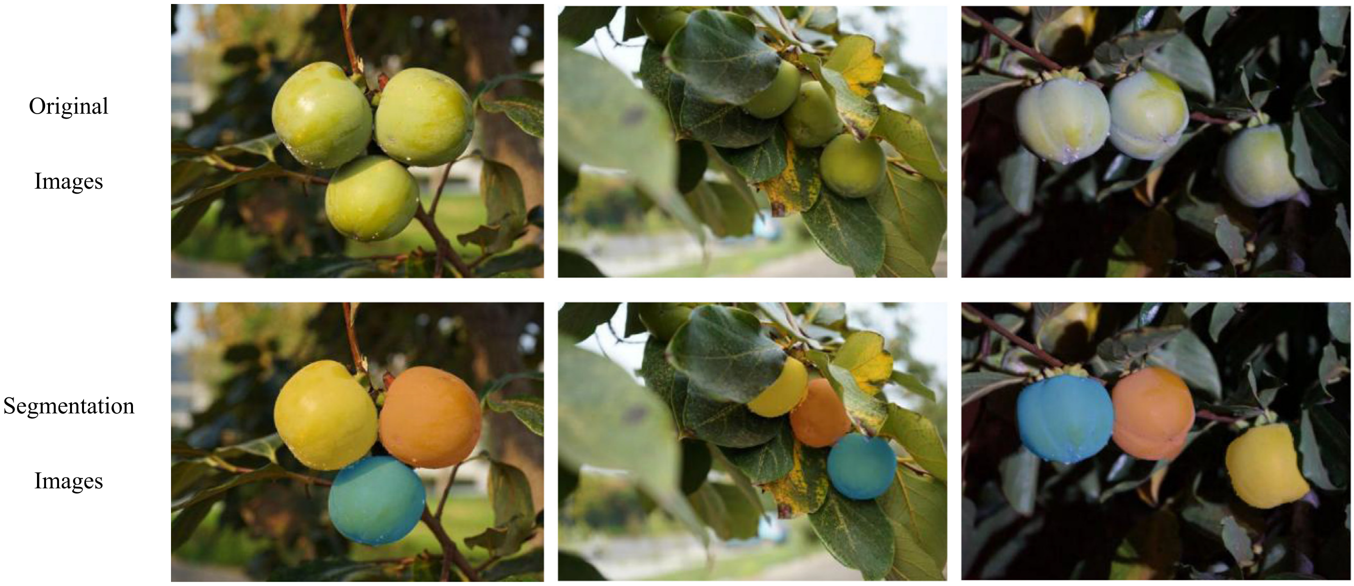
c) GCNet segmentation

d) Mask RCNN segmentation

e) Mask RCNN + Cascade segmentation

f) Mask Positioner segmentation

Fig. 9. Comparison images of segmentation effect. For example, the edge of the rightmost fruit (detected as a purple mask in figure f) is blocked by the leaves, resulting in an irregular shape. Unlike other algorithms which treat its left edge as smooth curves, only Mask Positioner can accurately distinguish the irregular fruit area form leaves.



a) front-lighting segmentation

b) back-lighting segmentation

c) night-time segmentation

Fig. 10. Segmentation effect of Mask Positioner under different lighting conditions. Under various lighting conditions, it can still accurately segment overlapped fruits, especially for fruit edges.

apple fruit. Table 8 shows that it is advanced to segmentation of small fruit, with an accuracy of 34.5%. The results show that Mask Positioner has good detection and segmentation effect for green apple fruits, so that it can meet segmentation tasks of green fruit in the real orchard environment.

5. Discussion

Section 5 discusses the segmentation effect, generalization, and other performance of the Mask Positioner.

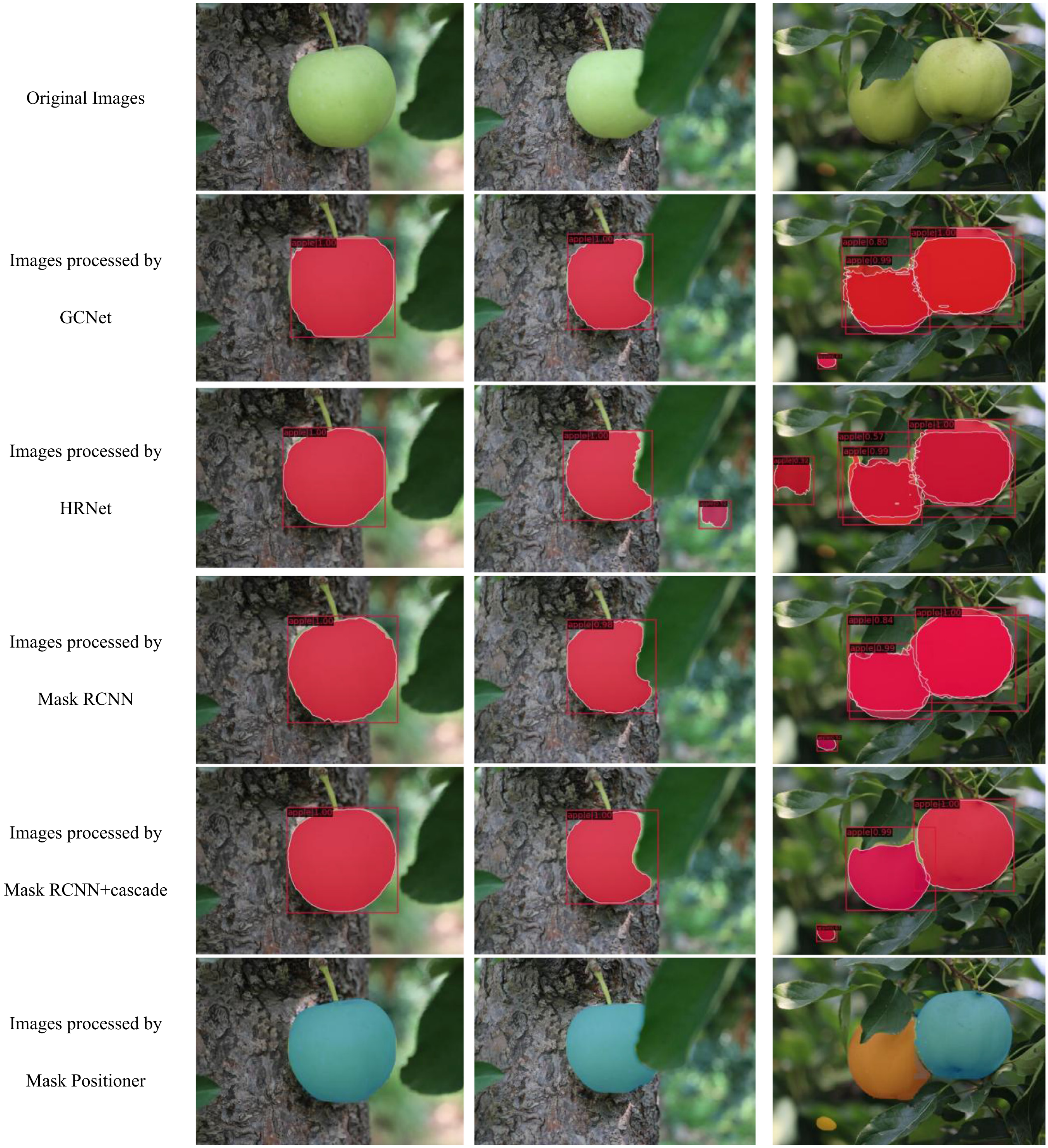


Fig. 11. Segmentation effect of the different algorithms on single, occluded and overlapping apple images. The masks produced by Mask Positioner have smooth and precise edges.

5.1. Segmentation effect

A typical image is used to visually show the segmentation effect of algorithms, as shown in Fig. 9. This is because there are many incomplete fruits of different situations in this picture at the same time. There are five green persimmon fruits in the image, of which the leftmost and middle fruits are overlapped, and the two fruits on the right are partially covered by branches and leaves. It can be seen from the figure b that masks of the occluded fruit produced by HRNet is not smooth enough, and there are redundant edges. In image c, masks generated by GCNet did not accurately cover the edge of the leftmost fruit, and Mask RCNN mistakenly judges the leaves as the sixth fruit in image d. Maybe due to the same baseline, Mask RCNN with Cascade also has this problem. In contrast, Mask Positioner covers all areas of fruits accurately and distinguishes the fruit from the complex background. In figure f, the

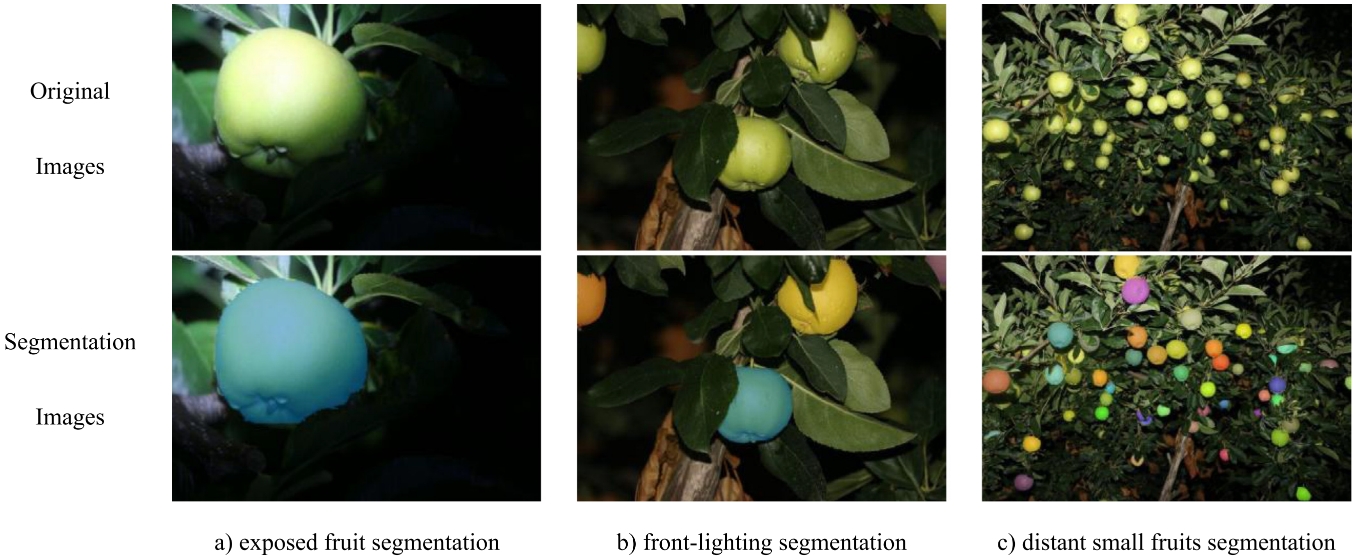

Original Images
Segmentation Images
a) exposed fruit segmentation
b) front-lighting segmentation
c) distant small fruits segmentation

Fig. 12. Segmentation effect of apple fruit images under different lighting conditions at night.

Table 9
Comparison of Parameter, FLOPs and FPS between algorithms.

Algorithms	Parameter (M)	FLOPs (G)	FPS (img/s)	References
HRNet	49.9	338.5	3.8	(Sun et al., 2019)
GCNet	54.2	260.2	3.9	(Cao et al., 2019)
Mask RCNN	44.17	260.1	4.9	(He et al., 2017)
Mask RCNN + Cascade	69.2	393.9	4.6	(Li et al., 2015)
Mask Positioner	53.9	413.6	3.3	

irregular fruit edges covered by the leaves on the right side are accurately distinguished, and masks of this area are not covered on the branches and leaves.

The experiment also verifies the segmentation effect of Mask Positioner in different light conditions. Similarly, images with occluded and overlapped fruits under different light are selected to show the segmentation effect, as shown in Fig. 10. Inside, figures a, b and c represent the conditions of front-lighting, back-lighting and night-time respectively. Under different lighting conditions, Mask Positioner can accurately segment the target fruits. Fig. 10 shows that the segmentation effect of Mask Positioner is not affected by lighting conditions, so that it can meet the needs of computer vision technology for 24-hour work in applications.

5.2. Generalization

Similarly to persimmon, three apple images of different occlusion conditions are used to verify the segmentation effect of Mask Positioner, as shown in the Fig. 11. In the absence of occlusion, all the algorithms can produce corresponding masks. Although the masks produced by GCNet and Mask RCNN have serrated section. For a single fruit covered by leaves, algorithms can also segment it roughly. It is obvious that the generated mask is not smooth enough for HRNet. As for the fruits that are occluded and overlapped at the same time, some algorithms will do redundant segmentation, such as GCNet. While Mask Positioner allocates accurately masks for the two fruits in the third figure. It can be seen that the edge of masks is smooth and the boundary between the two fruits is enough clear.

In addition to the above daytime conditions, this study also simulates the night working environment of computer vision technology. Segmentation effect of Mask Positioner at night is tested based on it, which is shown in the Fig. 12. Three typical night images are used to show the advantage of Mask Positioner. They from left to right are respectively severe exposure image, LED illuminating image and image with small fruits in the distant view. It can be seen from the previous two pictures that Mask Positioner can also accurately segment fruits in the different night light. The third picture proves the effectiveness of Mask Positioner for small fruits segmentation at night.

5.3. Other performance

This study adds other three metrics in addition to AP to comprehensively evaluate the algorithm performance. Parameter refers to the total number of parameters that need to be trained during the algorithm training process, which measures the size of the algorithm (space complexity). Floating-point operations (FLOPs) are understood as the computational complexity and can be used to measure the algorithm's complexity. Frames Per Second (FPS) represents the number of images the algorithm can process per second or the time it takes to process one image to evaluate detection speed. The shorter the time, the faster the speed. The performance of Mask Positioner (on persimmon dataset) on the GPU for these three metrics is shown in the Table 9.

From the Table 9, it can be seen that the number of parameters of Mask Positioner is kept in the middle, which is less than 10 M larger than that of the classical segmentation model Mask RCNN, but basically comparable to that of GCNet. So that its parameter is within a reasonable range, and Mask Positioner did not blindly increase the number of parameters to enhance the algorithm's fitting ability when constructing the network. Secondly, Mask Positioner has the largest FLOPs, which means that its time complexity is the highest among these algorithms, and this also leads to a slower inference speed.

This extra memory overhead is due to the use of the order decoder based on the transformer structure. This is because it uses self-attention mechanism to calculate the interaction between different positions in the input, which requires comparison of all positions in the input, resulting in an increase in computational complexity. However, the Mask Positioner achieves the highest accuracy, which is also attributed to the concept of separately refining edge-blurred regions. The algorithm sacrifice this computational cost, utilizing more computing resources, in order to exchange for higher accuracy through encoding and decoding of edge information. Currently, there have been studies in this area based on convolutional neural networks (Tan et al., 2019; Yu et al., 2020), which provides a direction for further optimization of Mask Positioner. For example, a new convolutional structure (Chen et al., 2023) was proposed to improve FPS of algorithms based on CNN.

6. Conclusion

In order to realize the orchard intelligent management and the application of picking robots, this study is committed to improving the accuracy of computer vision technology. However, in the complex orchard environment, the irregular area caused by occlusion makes it difficult to segment the target fruit accurately. In this study, we propose an efficient green fruit segmentation algorithm, called Mask Positioner, which accurately generates masks by selectively screening and processing the pixels requiring positioning. The DRN module is constructed in it, which combines the semantic information and location information of the top-level and low-level feature maps, avoiding the information loss of small size target fruit while refining the feature maps. In order to make the mask and the target area match completely, it screens the pixels that are not easy to judge (usually the edge part) and fuses them with multi-scale information. This is done in the hope that these pixels will carry more and more detailed information, which is conducive to subsequent pixel classification. The double attention structure of the order decoder effectively decodes the information and accurately judges whether the pixel belongs to the target itself.

Experiments show that Mask Positioner is more accurate than current mainstream algorithms based on CNN. Its detection accuracy on the green persimmon dataset reaches 69.1%, which is 0.3–4.9 percentage points higher than that of the classical algorithms, such as Mask RCNN. Especially for the detection of small fruit, the accuracy reaches 29.5%, which is 1.0 percentage point higher than the HRNet with the best detection effect of small fruit. Furthermore, the segmentation accuracy of Mask Positioner has also been improved. Its segmentation accuracy reaches 67.4%, which is 1.0–2.8 percentage points higher than the classical algorithms. For small size fruit, the segmentation accuracy of Mask Positioner is 3.2 percentage points higher than that of Mask RCNN improved with Cascade. In order to prove the generalization of the Mask Positioner on green fruits, this study also produced a green apple dataset. The experiment shows that the detection and segmentation accuracy of Mask Positioner on green apples are 57.9% and 53.5% respectively. What is the most important is that, it is advanced to other algorithms for the detection and segmentation of small fruit. The above fully proves that Mask Positioner is good at green fruit detection and segmentation under complex background. The mask generated by it accurately covers the overlapped and blocked fruit under different lighting conditions, so that it can cope with the working environment of computer vision technology, and achieve wide application.

Funding

This work is supported by Natural Science Foundation of Shandong Province in China (No.: ZR2020MF076); National Nature Science Foundation of China (No.: 62072289); New Twentieth Items of Universities in Jinan (2021GXRC049); Taishan Scholar Program of Shandong Province of China.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Bargoti, S., Underwood, J., 2017. Deep fruit detection in orchards. In: 2017 IEEE international conference on robotics and automation (ICRA). IEEE, pp. 3626–3633.

Bargoti, S., Underwood, J.P., 2017b. Image segmentation for fruit detection and yield estimation in apple orchards. J. Field Rob. 34 (6), 1039–1060.

Cao, Y., Xu, J., Lin, S., Wei, F., Hu, H., 2019. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In: Proceedings of the IEEE/CVF international conference on computer vision workshops, .

Chen, J., Kao, S. H., He, H., Zhuo, W., Wen, S., Lee, C. H., Chan, S. H. G., 2023. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. arXiv preprint arXiv:2303.03667.

Dorj, U.O., Lee, M., Yun, S.S., 2017. An yield estimation in citrus orchards via fruit detection and counting using image processing. Comput. Electron. Agric. 140, 103–112.

Ebrahimi, M.A., Khoshtaghaza, M.H., Minaei, S., Jamshidi, B., 2017. Vision-based pest detection based on SVM classification method. Comput. Electron. Agric. 137, 52–58.

Gongal, A., Amatya, S., Karkee, M., Zhang, Q., Lewis, K., 2015. Sensors and systems for fruit detection and localization: a review. Comput. Electron. Agric. 116, 8–19.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969.

Inkyu, S., Zongyuan, G., Feras, D., Ben, U., Tristan, P., Chris, M.C., 2016. Deepfruits: a fruit detection system using deep neural networks. Sensors 16 (8), 1222.

Jia, W., Wang, Z., Zhang, Z., Yang, X., Hou, S., Zheng, Y., 2022. A fast and efficient green apple object detection model based on Foveabox. J. King Saud Univ.-Computer Information Sci.

Koirala, A., Walsh, K.B., Wang, Z., McCarthy, C., 2019. Deep learning–Method overview and review of use for fruit detection and yield estimation. Comput. Electron. Agric. 162, 219–234.

Kong, T., Sun, F., Liu, H., Jiang, Y., Li, L., Shi, J., 2020. Foveabox: Beyound anchor-based object detection. IEEE Trans. Image Process. 29, 7389–7398.

Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G., 2015. A convolutional neural network cascade for face detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5325–5334.

Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117–2125.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440.

Mehta, S.S., Ton, C., Asundi, S., Burks, T.F., 2017. Multiple camera fruit localization using a particle filter. Comput. Electron. Agric. 142, 139–154.

Ngugi, L.C., Abelwahab, M., Abo-Zahhad, M., 2021. Recent advances in image processing techniques for automated leaf pest and disease recognition–A review. Information Process. Agric. 8 (1), 27–51.

Patel, H.N., Jain, R.K., Joshi, M.V., 2011. Fruit detection using improved multiple features based algorithm. Int. J. Computer Appl. 13 (2), 1–5.

Qureshi, W.S., Payne, A., Walsh, K.B., Linker, R., Cohen, O., Dailey, M.N., 2017. Machine vision for counting fruit on mango tree canopies. Precis. Agric. 18 (2), 224–244.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Adv. Neural Information Process. Syst., 28

Sun, K., Xiao, B., Liu, D., Wang, J., 2019. Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 5693–5703.

Sun, M., Xu, L., Chen, X., Ji, Z., Zheng, Y., Jia, W., 2022. Bfp net: balanced feature pyramid network for small apple detection in complex orchard environment. Plant Phenomics.

Tan, M., Le, Q., 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. PMLR, pp. 6105–6114.

Tan, M., Le, Q., 2021. Efficientnetv2: Smaller models and faster training. In: International Conference on Machine Learning. PMLR, pp. 10096–10106.

Tang, Y., Zhou, H., Wang, H., Zhang, Y., 2023. Fruit detection and positioning technology for a Camellia oleifera C. Abel orchard based on improved YOLOv4-tiny model and binocular stereo vision. Expert Syst. Appl. 211 (118573).

Wang, C., Tang, Y., Zou, X., SiTu, W., Feng, W., 2017. A robust fruit image segmentation algorithm against varying illumination for vision system of fruit harvesting robot. Optik 131, 626–631.

Yang, L., Chen, Y., Tian, W., Xu, Y., Ou, F., Wu, C., 2021. Field road segmentation method based on improved UNet. Trans. Chinese Soc. Agric. Eng. (Trans. CSAE) 37 (09), 185–191.

Yaşar, G.H., Akdemir, B., 2017. Estimating yield for fruit trees using image processing and artificial neural network. Int. J. Adv. Agric. Environ. Eng. (IJAAEE) 4 (1), 8–11.

Yu, C., Wang, J., Gao, C., Yu, G., Shen, C., Sang, N., 2020. Context prior for scene segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 12416–12425.