Research on Fruit Recognition and Positioning Based on you only look once version4 (YOLOv4)

Kong Dexiao 1,a, Li Junqiu 2,b, Zheng Jiawen 3,c, Xu Jiale 4,d, Zhang Qinghui 5,e*

1School of Big Data and Intelligent Engineering, Southwest Forestry University, Kunming, Yunnan, China
2School of Big Data and Intelligent Engineering, Southwest Forestry University, Kunming, Yunnan, China
3School of Big Data and Intelligent Engineering, Southwest Forestry University, Kunming, Yunnan, China
4School of Big Data and Intelligent Engineering, Southwest Forestry University, Kunming, Yunnan, China
5School of Big Data and Intelligent Engineering, Southwest Forestry University, Kunming, Yunnan, China

a,e-mail: 909956681@qq.com, b,e-mail: 2453575333@qq.com, c,e-mail: 527649713@qq.com, d,e-mail: x.jjl123@qq.com,
*
Corresponding author e-mail: huizh@163.com

Abstract. The recognition and spatial coordinate positioning is an important part of picking equipment of fruits. This paper introduced a method of target detection and pixel learning positioning of fruits based on the Darknet depth framework YOLOv4. We utilized GPU training in Ubuntu 19.10, and used 2000 images of various fruits as the training set for recognition model training, and performed and verified tests in the GPU environment. The results showed that the accuracy of fruit recognition is above 94%, the detection time of a single image is 12.3ms, and the detection rate of the video is 17f/s. The actual test showed that the system has good stability, real-time performance, and correctness of picking objects.

1. Introduction

With the development of modern technology, intelligent recognition technologies are increasingly applied to the design of fruit and vegetable picking equipment. While improving the automation and intelligence of agricultural production, these technologies also reduce production costs and improves work efficiency. However, due to the unstructured environments and shape difference between fruits, high-efficiency and low-loss picking of fruit is a worldwide problem. The realization of fruit recognition and spatial coordinate positioning is an important part of picking equipment design and implementation. At present, a large number of scholars have conducted a lot of research and experiments on picking equipment [1-3]. For example, Ceres R et al [4] designed and implemented a picking robot, which detected the fruit by laser ranging, then precisely positioned it by computer and controlled the mechanical system to achieve picking.
Aiming at the problem that it is difficult to determine the picking point position of fruit in the picking process of tomato harvesting robot, Liang et al. introduced a calculation method based on the corner point of fruit stem skeleton[5]. This method uses a binocular vision image acquisition system to collect 60 sets of fruit string images and obtain the position information of the fruit stalk picking point. The results showed that the success rate of picking point localization was 90%, which provided accurate picking position information for the picking robot. Zhao et al. proposed an apple localization method based on YOLOv3 (you only look once) deep convolutional neural network for apple picking localization, which provided a theoretical basis for the robot to identify apples in a complex environment quickly and efficiently for a long time[6].

2. Method
In this paper, the camera of Raspberry Pi was used to get the fruit images, and the resolution of the images is 640×480. After that, the image data were sent to the server, and the server uses the YOLOv4 recognition algorithm to detect the uploaded images and get the pixel information of the fruit in the image, and then the spatial coordinates of the fruit are positioned according to this information. At the same time, data relaying was also carried out, and the APP on the cell phone was mainly used to control the start and stop of the picking equipment, as well as to view the real-time picking video with the workload of the picking equipment.

2.1. YOLOv4
In 2016, Joseph Redmon et al. proposed the pioneering YOLO algorithm for target classification and position regression of target frames in a separate network, which has a relatively high detection accuracy and a relatively fast detection speed[7]. Based on the one-stage framework, YOLO has been developed and released in five versions so far. However, YOLOv1’s detection method is used in all versions of YOLO which just are improving and optimization based on YOLOv1. The methods of target detection in YOLOv1 are as follows:

1. The input image is divided into \( S \times S \) grid cells. If the center of the target object falls into a grid cell, that grid cell is responsible for detecting that target.
2. Each grid cell predicts \( B(B=2) \) bounding boxes and the confidence scores of these boxes. This score reflects the probability \( P \) that the box contains an object and the positional accuracy, namely Intersection over Union (IOU), of the predicted box. Therefore, the confidence score is defined by these two components.
3. After the target detection is completed, the bounding box of each object in the image is output, each bounding box should contain \( x, y, w, h \), and confidence. The confidence represents the IOU between the box and the ground truth (if no object in the box, the score equals zero directly).
4. Because the location and category need to be predicted at the same time, so each cell outputs not only the bounding box, but also the conditional probability of the object (namely, the probability that the object belongs to a certain category. Of course, these probabilities are conditional on the grid cell containing the object), and each grid outputs the category with the highest probability.

YOLOv4 was proposed in 2020, and it combines several improvement techniques to achieve a high level of accuracy and speed, making it one of the best performing detection algorithms in the current stage of target detection[8].

2.2. Recognition model training
In the actual picking process, the video images captured by the Raspberry Pi camera include other non-fruit objects such as leaves, branches and trunks in addition to fruits. As shown in Fig. 1a, the system uses the OpenCV computer vision and machine learning software library to process the video for better fruit detection, recognition, and pixel localization in complex environments, and then uses the YOLOv4 target detection algorithm based on the deep learning framework Darknet for fruit object detection and fruit pixel localization. The Darknet framework is implemented in C language and is easy to install, portable, and supports both CPU and GPU (CUDA) computing. In terms of performance YOLOv4 runs
2 times faster than EfficientDet and has comparable performance, increasing the average accuracy (AP) and frame rate (FPS) of YOLOv3 by 10% and 12% respectively.

![Original image](a) Original image  ![Detection result](b) Detection result

Figure 1: Recognition and target detection of fruit

For model training, 500 images of various fruits were taken under unobstructed lighting conditions, and 500 images of each fruit were processed in four ways, including 90°, 180°, and 270° rotation and X-axis mirroring. Then, we obtained a training data set of 2500 images for each fruit, of which 2000 images were used for model training and 500 images were used for model validation. After the training set was determined, the image annotation tool was used to annotate the fruit in the image and generate an XML annotation file in PASCAL VOC format.

Five parameters (class_fruit,x,y,w,h) were used to describe the fruit in the image in the generated annotation file. Class_fruit denotes the type of fruit, x and y denote the coordinates of the center of the fruit in the image, and w and h denote the length and width of the target frame. After that, the configuration file in YOLOv4 is modified. The modifications to the configuration file in this project are as follows: batch is 16 (number of samples per batch of training), subdivisions are 16 (number of batch segments), max_batches is 6000 (number of iterations), steps is 4800, 5400 (learning rate, batch*0.8 is 4800, batch* 0.9 is 5400).

The hardware environment for model training was an Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz (8CPUs) processor, 16GB running memory, GTX 1080Ti graphics card with 8GB of video memory, and Ubuntu 19.10 operating system as the software environment. The system was trained to recognize four fruit models of apple, nectarine, yellow apricot, and plum.

2.3. Spatial positioning

Getting the spatial coordinates of the fruit is a very critical step to achieve fruit picking, and the system determines the fruit coordinates by getting the coordinates of the fruit relative to the camera, as shown in Fig. 8. In the figure, the XOY and XOZ planes of the fruit and the camera are established respectively, P denotes the image captured by the camera, A denotes the fruit, a denotes the size of the fruit A in the image, and ay and ax denote the number of pixel points occupied by each of a in the x and y directions in the image.

In coordinate positioning, we should get the conversion relationship between the pixel point in the image and the actual object size. Firstly, we obtained the conversion ratio by using the camera to shoot the coin. While the diameter of the coin is 3cm, the shooting distance is 1m, and the shoted images have 163 pixels, we acquired the conversion ratio $k=163/3=54.3$ pixels/cm. $P_y$ and $P_z$ in the figure indicate a center point in the y-axis and z-axis to the central axis. According to the similarity principle, $k=P_y/Y_L$ and $k=P_z/Z_L$, from which we can get the formulae for calculating the y and z coordinates in the spatial coordinates of the fruit object, as follows:

$$Y_L = P_y/k$$

$$Z_L = P_z/k$$

Based on the principle of monocular camera ranging, the x-coordinate in the spatial coordinates of the fruit is calculated as follows:

$$D = W \times F / P$$
where $D$ denotes the relative distance between the fruit and the camera, $W$ denotes the width of the fruit, $F$ denotes the camera focal length, and $P$ denotes the number of pixels occupied by the fruit in the image. The focal length of the Raspberry Pi camera is $F=543$ pixels. As shown in Fig. 2, $W = a_x/k$, $P = a_x$, $D = X_L$, thus the calculation formula is modified as follows:

$$X_L = (a_x/k \times F)/a_x$$

(4)

3. Results

After training the model, the fruit target detection is performed on the live image uploaded by Raspberry Pi, and the confidence level, type, pixel size and center pixel position of the fruit in the image are outputted after the detection. As shown in Fig. 1b, the fruit is framed by a blue-green border and labeled as "lizi" with 100% confidence level.

The single picking time of the picking system is shown in Tab. 1, and $T_a$ in the table is the total time of picking 60 times for various fruit species. Since the formula of single fruit picking time is $T = T_1 + T_2$, and $T_2$ is an uncertain value, so the single picking specific time $T$ cannot be accurately determined. Therefore, the average time $T_n$ is used here instead of $T$. From the table, we can learn that the longest single picking time $T_n$ is 17.98s and the shortest single picking time $T_n$ is 16.58s.

| Fruit type | $T_a$ | $T_n$ | $T_1$ | $T_2$ |
|------------|-------|-------|-------|-------|
| Apple      | 17'41'' | 17.68s | 15s   | 2.68s |
| Nectarine  | 16'35'' | 16.58s | 15s   | 1.58s |
| YellowApricot | 17'59'' | 17.98s | 15s   | 2.98s |
| Plum       | 17'45'' | 17.75s | 15s   | 2.75s |

TABLE. 2 Training model test results of different kinds of fruits

| Fruit type          | mAP/% | Accuracy/% | Recall/% | IOU/% |
|---------------------|-------|------------|----------|-------|
| Apple               | 85.72 | 94.52      | 86.5     | 81.21 |
| Nectarine           | 86.54 | 94.63      | 87.3     | 81.72 |
| YellowApricot       | 85.21 | 94.71      | 86.4     | 82.61 |
| Plum                | 86.13 | 95.01      | 87.1     | 83.01 |

The results of testing the trained models for different types of fruits are listed in Tab. 2. Each fruit has 2000 images in the training set and 500 images in the test set. We used GPU for model training in Ubuntu 19.10 operating system, and the mean accuracy (mAP), accuracy, recall and intersection ratio (IOU) for each category obtained by testing with the trained models are shown in Tab. 2.

The results of spatial coordinate calculation for four kinds of fruits in the system picking process are shown in Tab. 3-6. In these tables, the fruit width indicates the actual width of the fruit, the bounding box
information is the pixel information obtained by the recognition algorithm after recognizing the fruit, which are the center pixel coordinates \((x, y)\) and the occupied pixel width \(w\) and height \(h\). The calculated spatial coordinates are the coordinate positions of the fruit relative to the camera calculated by the algorithm, and the actual spatial coordinates are the coordinate information obtained by the actual measurement.

In the picking test process, a fixed fruit is selected for each medium fruit, and the picking test is carried out under good lighting conditions and the fruit is unobstructed. The distance between the fruit and the camera was kept constant for each test, and the \(x\) coordinate was kept constant, only the \(y\) and \(z\) coordinates were transformed even for the test verification. As shown in each table, the difference between the calculated coordinates and the actual spatial coordinates was within 1 cm. It shows that the spatial coordinate calculation of fruits can be realized by the spatial coordinate positioning algorithm in this paper. Meanwhile, from the four tables, it can be seen that the spatial positioning algorithm has less error in positioning larger fruits, and the success rate of picking two kinds of fruits, apple, and nectarine, is higher in the actual picking process, which indicates that the positioning algorithm is suitable for larger fruits.

| TABLE. 3 coordinate positioning calculation test of apple |
|-----------------|-----------------|-----------------|-----------------|
| Fruit Width cm  | Bounding box information \((x,y,w,h)/px\) | Calculated spatial coordinates \((x,y,z)/cm\) | Actual spatial coordinates \((x,y,z)/cm\) |
|-----------------|-----------------|-----------------|-----------------|
| 6.0             | (320,240,250,250) | (13.03,0,0)     | (13.2,0,0)      |
| 6.0             | (480,360,250,250) | (13.03,2.88,3.84) | (13.2,2.5,4.0)  |
| 6.0             | (160,120,250,250) | (13.03,-2.88,-3.84) | (13.2,-2.5,-4.0) |
| 6.0             | (480,120,250,250) | (13.03,-2.88,3.84) | (13.2,-2.5,4.0)  |
| 6.0             | (160,360,250,250) | (13.03,2.88,-3.84) | (13.2,2.5,-4.0)  |

| TABLE. 4 coordinate positioning calculation test of nectarine |
|-----------------|-----------------|-----------------|-----------------|
| Fruit Width cm  | Bounding box information \((x,y,w,h)/px\) | Calculated spatial coordinates \((x,y,z)/cm\) | Actual spatial coordinates \((x,y,z)/cm\) |
|-----------------|-----------------|-----------------|-----------------|
| 4.5             | (320,240,200,200) | (12.21,0,0)     | (12.0,0,0)      |
| 4.5             | (480,360,200,200) | (12.21,2.67,3.6) | (12.0,2.5,3.5)  |
| 4.5             | (160,120,200,200) | (12.21,-2.67,-3.6) | (12.0,-2.5,-3.5) |
| 4.5             | (480,120,200,200) | (12.21,-2.67,3.6) | (12.0,-2.5,3.5)  |
| 4.5             | (160,360,200,200) | (12.21,2.67,-3.6) | (12.0,2.5,-3.5)  |

| TABLE. 5 coordinate positioning calculation test of yellow apricot |
|-----------------|-----------------|-----------------|-----------------|
| Fruit Width cm  | Bounding box information \((x,y,w,h)/px\) | Calculated spatial coordinates \((x,y,z)/cm\) | Actual spatial coordinates \((x,y,z)/cm\) |
|-----------------|-----------------|-----------------|-----------------|
| 3.8             | (320,240,180,180) | (11.46,0,0)     | (11.72,0,0)      |
| 3.8             | (480,360,180,180) | (11.46,2.53,3.78) | (11.72,2.45,3.5) |
| 3.8             | (160,120,180,180) | (11.46,-2.53,-3.78) | (11.72,-2.45,-3.5) |
| 3.8             | (480,120,180,180) | (11.46,-2.53,3.78) | (11.72,-2.45,3.5)  |
| 3.8             | (160,360,180,180) | (11.46,2.53,-3.78) | (11.72,2.45,-3.5)  |
TABLE. 6 coordinate positioning calculation test of plum

| Fruit Width cm | Bounding box information (x,y,w,h)/px | Calculated spatial coordinates (x,y,z)/cm | Actual spatial coordinates (x,y,z)/cm |
|----------------|---------------------------------------|------------------------------------------|--------------------------------------|
| 3.0            | (320,240,200,200)                     | (8.15,0.0)                               | (8.37,0.0)                          |
| 3.0            | (480,360,200,200)                     | (8.15,1.8,2.4)                           | (8.37,2.15,2.76)                    |
| 3.0            | (160,120,200,200)                     | (8.15,-1.8,-2.4)                         | (8.37,-2.15,-2.76)                  |
| 3.0            | (480,120,200,200)                     | (8.15,-1.8,2.4)                          | (8.37,-2.15,2.76)                   |
| 3.0            | (160,360,200,200)                     | (8.15,1.8,-2.4)                          | (8.37,2.15,-2.76)                   |

Through the actual picking test, the picking system can realize the automatic picking of four kinds of fruits: apple, nectarine, yellow apricot and plum, and the system works stably. However, this system design and research are still at the experimental stage, and there are some shortcomings:

a) The data sets used in the fruit recognition model are all unobstructed fruit images taken under sufficient light conditions, so the picking system can only pick unobstructed fruits under sufficient light conditions at present. If we need to design a better picking system, we need more live images of fruits for model training.

b) The calculation of the spatial coordinate positioning of the fruit depends on the bounding box information output from the YOLOv4 recognition model, which is not accurate enough and can easily lead to errors in spatial coordinate positioning.

4. Conclusion

This intelligent fruit picking device is designed based on Raspberry Pi, six-degree-of-freedom robot arm, cell phone APP and server to build the system, which can realize the automatic identification and automatic picking of various fruits. During the picking process, the Raspberry Pi takes real-time video and uploads it to the server, which uses the YOLOv4 target detection and recognition algorithm based on the Darknet deep learning framework to identify the fruit and calculate the pixels, and finally realizes the spatial coordinate positioning of the fruit. Then the server will return the spatial coordinates of the fruit to Raspberry Pi, and the Raspberry Pi robot arm will carry out attitude solving and motion trajectory planning to realize the automatic picking of the fruit according to the spatial coordinates. Through the cell phone APP, we can remotely view the video of the picking site and control the working and stopping of the picking equipment.

In the future, we will focus on the automatic control of the robotic arm movement, and by making our own training set and training the recognition model, we can recognize a variety of fruit items and improve the intelligence of the picking equipment.

Acknowledgment

The authors would like to thank the Biological Resources Digital Development and Application Project of Yunnan Science and Technology Department (202002AA10007), Scientific Research Foundation of Yunnan Education Department (2020J0402, 2021J0153) for their support in this research.

References

[1] R. Bogue, "Fruit picking robots: has their time come?", Industrial Robot, Vol. 47(2), 2020, pp. 141-145.
[2] Li Peilin, Sang-heon Lee, and Hung-Yao Hsu. "Review on fruit harvesting method for potential use of automatic fruit harvesting systems." Procedia Engineering, Vol. 23, 2011, pp. 351-366.
[3] Bechar, Avital, and Clement Vigneault. "Agricultural robots for field operations: Concepts and components." Biosystems Engineering, Vol. 149, 2016, pp. 94-111.
[4] Ceres R, Pons J L, Jimenez A R, et al. “Design and implementation of an aided fruit harvesting robot (Agribot).” Industrial Robot: An International Journal, 1998, Vol. 25(5), pp. 337-346.
[5] Liang Xifeng, Jin Chaoqi, Ni Meidi, Wang Yongwei, et al. “Acquisition and experiment on
location information of picking point of tomato fruit clusters.” Transactions of the Chinese Society of Agricultural Engineering, 2018, Vol. 34(16), pp. 163-169. (in Chinese)

[6] Zhao Dean, Wu Rendi, Liu Xiaoyang, et al. “Apple positioning based on YOLO deep convolutional neural network for picking robot in complex background.” Transactions of the Chinese Society of Agricultural Engineering, 2019, Vol. 35(03), pp. 172-181. (in Chinese)

[7] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[8] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection." arXiv preprint arXiv:2004.10934, 2020.