IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 28, 2020, accepted June 18, 2020, date of publication June 29, 2020, date of current version July 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3005386

Detection of Fruit-Bearing Branches and Localization of Litchi Clusters for Vision-Based Harvesting Robots

JINHUI LI[1], YUNCHAO TANG[2], (Member, IEEE), XIANGJUN ZOU[1], GUICHAO LIN[1], AND HONGJUN WANG[1]

[1]College of Engineering, South China Agricultural University, Guangzhou 510642, China
[2]College of Urban and Rural Construction, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China

Corresponding authors: Yunchao Tang (ryan.twain@zhku.edu.cn) and Xiangjun Zou (xjzou1@163.com)

This work was supported in part by the Science and Technology Planning Project of Guangdong Province under Grant 2019A050510035, in part by the National Natural Science Foundation of China under Grant 31571568, and in part by the Research and Development Projects in Key Areas of Guangdong Province under Grant 2019B020223003.

ABSTRACT Litchi clusters in fruit groves are randomly scattered and occur irregularly, so it is difficult to detect and locate the fruit-bearing branches of multiple litchi clusters at one time. This is a highly challenging task related to continuous operation in the natural environment for visual-based harvesting robots to carry out. In this study, a reliable algorithm based on RGB-depth (RGB-D) cameras in the fields was developed to accurately and automatically detect and locate the fruit-bearing branches of multiple litchi clusters simultaneously in large environments. A semantics segmentation method, Deeplabv3, was employed to segment the RGB images into three categories: background, fruit and twig. A pre-processing step is proposed to align the segmented RGB images and remove the twigs that did not bear fruits. Subsequently, the twig binary map image was processed via skeleton extraction and pruning operations, which left behind only the main branches of twigs. A method for non-parametric density-based spatial clustering of application with noise was used to cluster the pixels in the three-dimensional space of the skeleton map of the branches; thus, the fruit-bearing branches belonging to the same litchi clusters were determined. Finally, a three-dimensional straight line was fitted to each cluster via principal component analysis, and the linear information corresponded to the location of the fruit-bearing branches. In the experiments, 452 pairs of RGB-D images under different illumination were collected to test the proposed algorithm. The results show that the detection accuracy of a litchi fruit-bearing branch is 83.33%, positioning accuracy is $17.29° \pm 24.57°$, and execution time for the determination of a single litchi fruit-bearing branch is 0.464s. Field experiments show that this method can effectively guide the robot to complete continuous picking tasks.

INDEX TERMS Continuous picking, location detection of fruit-bearing branches, harvesting robots, RGB-D image.

I. INTRODUCTION

Litchis are characteristic fruits of economic importance in South China, and their planting area and production output rank first globally. In 2017, the planting area and output of litchis in China were about 546,667 m^2 and 2.3 million-tons, accounting for 68.3% and 65.7% of the global total, respectively. Owing to the short fruiting period, picking litchis in time is crucial for a successful harvest. Artificial harvesting is currently a common harvesting method, but it is very laborious and time consuming. Therefore, it is feasible to develop intelligent robotic systems that can pick litchis automatically [1]. Many related studies have developed such fruit-harvesting robots [2]–[7]. The accurate detection and spatial positioning of the target fruits is the key to successfully complete the picking task using such vision-based harvesting robots.

In this regard, fruit detection and localization have been studied extensively [8]–[11]. In [12], the researchers developed a real-time apple detection and localization vision

The associate editor coordinating the review of this manuscript and approving it for publication was Varuna De Silva.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

system with a detection rate of 100% and a positioning accuracy of ±3mm. In [13], a charge-coupled device (CCD) camera was employed to acquire images of apples, and a color- and shape-based support vector machine (SVM) was used to detect the fruits; herein, an 89% success rate was yielded. In [14], a vision system was developed based on an RGB-depth (RGB-D) camera under a light shield to detect bicolored apples using both color and shape information. The algorithm thereof could detect 100% of fully visible apples and 82% of partially occluded apples, and the positioning error was below 10mm. The detection of all fruits (and obstacles) and localization of an apple tree were realized simultaneously in one RGB-D image. A monocular vision system for medium- and large- scale citrus harvesting was developed in [15], wherein perspective transformation-based range estimation was used to obtain three-dimensional (3D) information on fruit position under low computational requirements, with an accuracy of 15mm. In [16], a novel processing algorithm was developed to detect kiwifruits and draw separate lines to separate each fruit according to its growth characteristics; a common camera was placed underneath the fruits to capture images. A 93.7% success rate of detection was yielded for fruit calyxes, and a 92% success rate was yielded for the correct segregation of fruits during night-time with flash. In [17], two CCD color cameras integrated with a window zooming-based algorithm were utilized to locate multiple fruits and vegetables. The results showed that under varying illumination and partially occluded circumstances, the localization errors were less than 7.5 mm for a measuring distance of 300–1600mm. Among the abovementioned studies, those that adopted multiple visual sensing methods combined with advanced algorithms to detect and locate target fruits usually achieved a high success rate. However, those studies mainly focused on fruits that grow in single units, of which the picking points are at the geometric center. These studies are not applicable to stringed fruits, whose picking positions are at the fruit stem or the fruit-bearing branch.

In recent years, several studies have explored the direct or indirect detection of branches (or stems), mainly for harvesting or pruning purposes [18]–[23]. For instance, to detect branches of the sweet cherry tree with full foliage under outdoor vibratory picking, the author of [24] developed a pixel-based segmentation and detection method, used a Bayesian classifier to classify images captured at night-time using a stereo-vision camera under controlled lighting, and then linked the same branch together, selecting the best-fit model as the final branch of the cherry tree. The detection accuracy was as high as 89.2% in this case. In another study [25], to address the problem of branch obstacle localization, an apple-harvesting robot was enabled to subtly extract the endpoints and bifurcation points in the branch skeleton image, obtained from RGB images as feature points, and perform stereo matching of those feature points to obtain the position information of the branches with a ±6.2 mm localization error. In [26], the author implemented four deep learning frameworks to segment images of grape bunches acquired via an RGB-D sensor to estimate canopy volume and count the bunches, with a maximum accuracy of 91.52% being yielded despite the input images being of poor quality. In [27], for harvesting purposes, the support wire was used as a visual cue to locate the stems of sweet-peppers on the basis of a stereo-vision system under controlled lighting. This method achieved an accuracy of ±0.4 cm. In [28], the authors studied the R-G color model of cherry tomato images; therein, the cherry tomatoes were planted in a greenhouse and treated with artificial pruning. *CogPMAlignTool* was employed to identify four marginal fruits of a mature bunch to guide the robot to finish the task indirectly, yielding an 83% successful harvest. In another study [29], an acceptable method was proposed to locate the picking point of a litchi cluster via application of different color spaces when one image contained only a bunch of fruits, with an 83% accuracy rate and near-real-time results being yielded. Some of the abovementioned studies needed either a cue or ideal conditions for good performance to be achieved. In some of the others, the picking branches could only be identified indirectly after the fruits were first detected.

We can therefore conclude that RGB-D-based methods are suitable for fruit detection in the field, the recognition and location of branches is a crucial step for fruit bunch harvesting, and advanced algorithms should be utilized to improve the location accuracy and success of a harvest. However, changes in illumination; occlusions of leaves, fruits and branches; and disturbances due to wind remain obstacles regarding accurate detection of fruits by robots under natural harvest environment conditions. Our previous research mainly focused on the detection and localization of single-string litchis via picking of one string at a time and usage of an eye-in-hand system (i.e., a camera installed at the end-effector of the robot that moves with the robot). In [30], the litchi stem was extracted using images based on morphological processing collected in their natural environment with an 80% success rate. In this study, one image included one litchi cluster, and the stem was not occluded. In [31] and [32], two different localization methods for litchis were proposed on the basis of binocular stereo vision. In order to extract the litchi clusters using the images, different segmentation and matching methods were employed that are resistant towards illumination changes and partial occlusions when the litchi images were collected under certain conditions (target fruits had a pixel ratio in the images). The highest average recognition rate and matching success rate were 98.8% and 98%, respectively, and the algorithm execution time was approximately 3 s. We also built an error model of the manipulator, analyzed the fault-tolerant range of end-effector, and designed a five degree-of-freedom (DOF) manipulator and picking end-effector to verify the effectiveness of the algorithm [33], [34]. We further investigated the litchi clustering recognition and picking point calculations under night-time [35] and disturbance conditions [36] to enable the robot to successfully complete the picking task under certain specific conditions.

The growth of litchi clusters on a tree is typically random and irregular, so it is difficult to simultaneously detect and locate the fruit-bearing branches of multiple litchi bunches in one image, which presents a major challenge to vision-based harvesting robots that are intended to perform continuous operations under natural environments. However, to date, no such research has been reported in this regard. In a recent study [37], the authors considered prioritizing robotic grasping of stacked fruit clusters with small stalks on the basis of RGB-D images, which is one of the most relevant studies addressing this challenge. Therefore, we have also attempted to research this issue. Inspired by object segmentation based on deep convolutional neural networks and fruit 3D localization methods, this study aims to develop a combined algorithm to detect and locate small fruit-bearing branches in a large environment where one captured image contains multiple litchi strings to be picked. The color and depth information in one RGB-D image were combined to provide reliable picking information for the harvesting robot. Because some small visible branches in the image may contain important picking information, a precise pixel-based semantic segmentation method (Deeplabv3) was used to segment fruits and twigs from the background. Subsequently, a series of morphological processing methods and 3D spatial positioning methods were used to extract more accurate information for picking fruit-bearing branches. This allowed for simultaneous extraction and location of all fruit clusters or obstacles in a given environment scene. The effectiveness of the proposed method was validated via experiments.

The remainder of this article is organized as follows: Section II describes the materials, methods, and algorithms of the sensor system. The results are presented and discussed in Section III, and the conclusions and suggestions for further research are included in Section IV.

II. MATERIALS AND METHODS

A. SENSOR SYSTEM AND IMAGE ACQUISITION

In this study, Kinect V2, a low-cost RGB-D camera was used, which consists of a color camera and an infrared camera that can generate an RGB image with a resolution of 1920×1080 pixels and a depth image with a resolution of 512×424 pixels at 30 fps. For the experiments, the Guiwei litchi variety was used, and 452 pairs of images were collected from: (1) an orchard in Zengcheng, Guangzhou, China on June 27, 2018 (sunny) and (2) a Litchi Cultivation Experimental orchard base of South China Agricultural University, Guangzhou, China on May 20, 2019 (heavy rain to clear conditions). All kinds of illuminations were included, and no artificial shade or lighting interference was incorporated. As per [38], the images in this study were randomly divided into a training set and a test set in a ratio of 80:20.

The Kinect camera was placed within a range of 500–800 mm from the litchi trees, which not only covered the field of vision of the camera but also the reachable area of the picking robot. This vision sensor was mounted on a litchi harvesting robot, as shown in **FIGURE 1**. The picking system also included a six-DOF manipulator equipped with a liftable walking platform, independently developed pneumatic-drive end-effector affixed on the end-axle of the manipulator, and computer (Intel Core i7-8700 CPU @ 3.20GHz and 8GB RAM with Windows 10 operating system) to integrate the algorithm. The program environment for the localization software system included opencv-3.4.1 and Visual C++ 2017; MATLAB R2017b was used to evaluate the algorithm for the in-filed experiments. An image annotation tool, namely LabelMe, was used to manually label images at the pixel level as background, fruit, or twig. This task was however highly time-consuming and took 15 days for two people to complete because of the large number of images used and the small and scattered target in each of those images. Subsequently, those data were converted to the TFrecord type and registered for image segmentation.

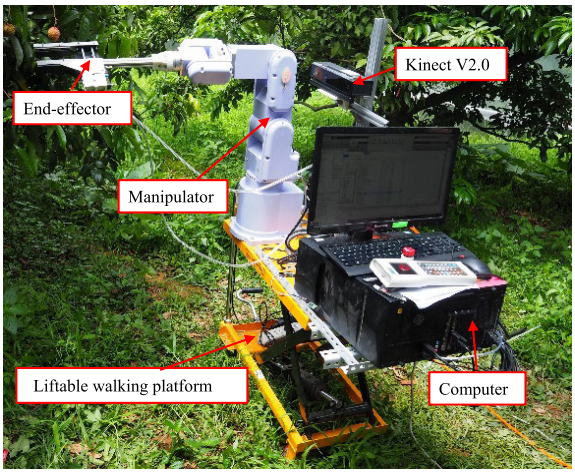


FIGURE 1. The litchi harvesting robot system.

B. FERTILE BRANCH DETECTION AND LOCATION ALGORITHM

The algorithm's overall procedure for vision-based fertile branch detection and 3D spatial localization of litchi clusters is shown in **FIGURE 2**. It involves three main steps:

Step 1: A powerful semantic segmentation tool, Deeplabv3, was used to segment the captured RGB images to obtain the pixel area of litchi fruits and twigs. Then, the segmented and depth images were aligned according to the depth-to-color mapping relationship. Details pertaining to this are presented in Section B-1.

Step 2: The fruits-bearing branches were detected via morphological processing. This included the following processes: removal of the fruitless twigs in the aligned images; extraction of the skeleton and removal of the burr of these twigs to obtain single-pixel twigs; and pruning of those twigs to obtain the main branches, which are also the fruit-bearing branches. More details are presented in Section B-2.

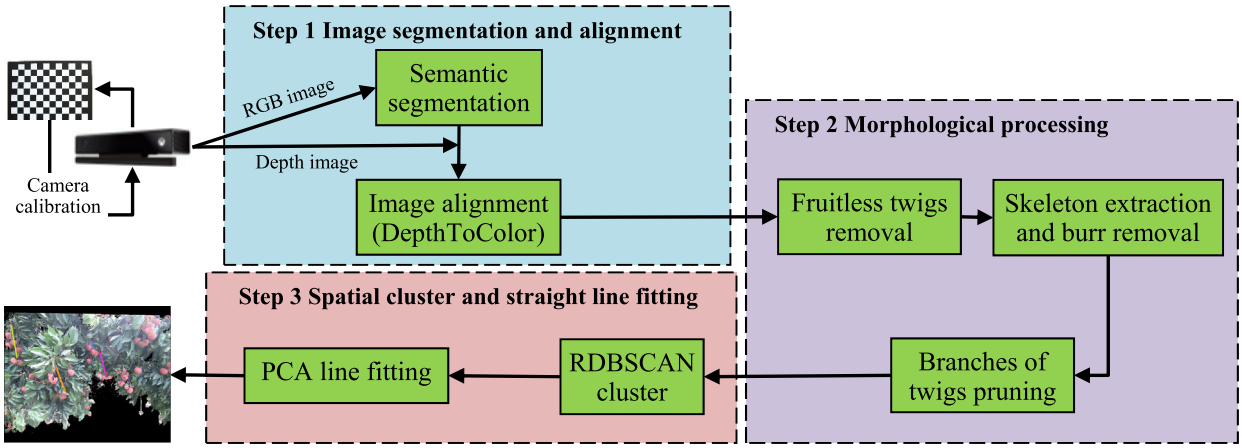


FIGURE 2. Algorithm flow chart of litchi fruit-bearing branch detection and localization based on RGB-D images.

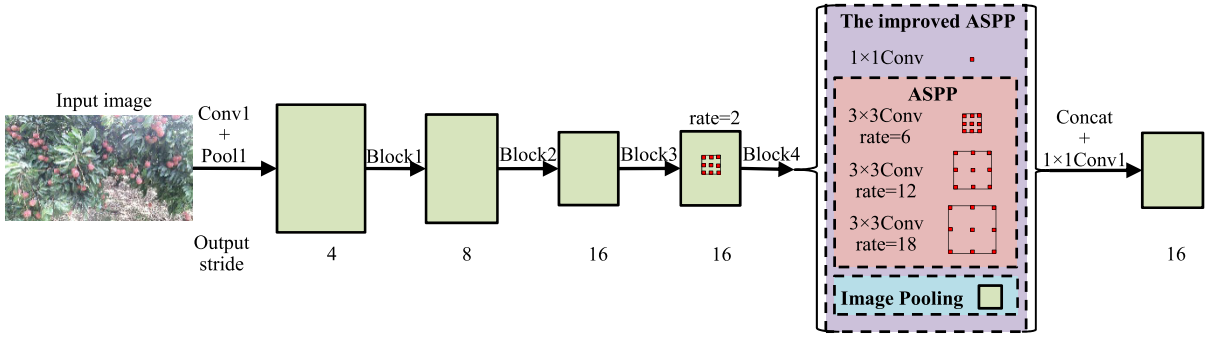


FIGURE 3. Parallel modules of Deeplabv3; the improved ASPP to capture the multi-scale information of target.

Step 3: In order to locate the picking line of the fruit-bearing branches in the 3D space, the branches projected into the space were clustered via revised density-based spatial clustering of applications with noise (DBSCAN) and fitted with a straight line via principal component analysis (PCA), and the position data of those branches in the 3D space were obtained to determine the picking position. More details are presented in Sections B-3 and B-4.

1) IMAGE SEGMENTATION AND ALIGNMENT

The purpose of image segmentation is to segment the fruits and twigs from the background in the RGB images. An advanced semantic segmentation general framework, Deeplabv3, was employed [39]. It combines deep convolutional neural networks (DCNNs) with different atrous rates of atrous convolutions to go deeper or applies the improved atrous spatial pyramid polling (ASPP) to perform multi-scale semantic segmentation of objects. Some highly effective parallel modules of Deeplabv3 are shown in **FIGURE 3**. To expand the receptive field to extract multi-scale information, it duplicates several copies of the original last block in ResNet [40], marked as block 4 in **FIGURE 3**, following which it arranges them in cascade or in parallel (shown as ASPP in **FIGURE 3**, consisting of three 3×3 convolution with rates of 6, 12, and 18). The improved ASPP also applies one 1×1 convolution before ASPP to upsample the feature to the desired spatial dimension and global average pooling on the last feature map after ASPP to incorporate global context information. The resulting features are then concatenated to generate the final results after a 1×1 convolution.

The input to Deeplabv3 is the RGB images captured using Kinectv2.0 at a resolution of 1920×1080 pixels. We fine-tuned Deeplabv3 from its original structure in [39] by using a publicly pre-trained model [41] and Xception_65 [42] backbone network on our small training dataset. The implementation details include: (1) Making our own data sets according to PASCAL-VOC2012 [43] and initializing the parameters for Deeplabv3 model through reuse of all trained weights except the logits; (2) Improving the result by fine-tuning hyperparameters such as with a learning rate of 0.0001 and batch size of 2; and (3) 50,000 training iterations to stop the learning procedure.

Subsequently, we aligned the segmented image and the depth image. The process of mapping the world coordinate point $M(X_W, Y_W, Z_W)$ to the image point $m(v, u)$ is shown as **FIGURE 4**. We could determine an RGB value for every depth data in depth images within 1200mm (images out of the

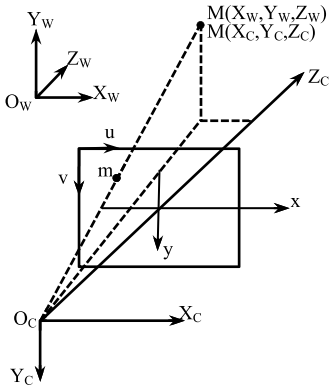


FIGURE 4. Mapping relationship between the world coordinate system and image.

range are seen as background) using (1).

$$\begin{cases} \mathbf{D}_i = \boldsymbol{Dpt}\ (v_i, u_i) \\ \boldsymbol{u}_{i_{rgb}} = \boldsymbol{W}\ (\mathbf{1}, \mathbf{1})\, \boldsymbol{u}_i + \boldsymbol{W}\ (\mathbf{1}, \mathbf{2})\, \boldsymbol{v}_i + \boldsymbol{W}\ (\mathbf{1}, \mathbf{3}) + \boldsymbol{W}\ (\mathbf{1}, \mathbf{4})\ /\boldsymbol{D}_i \\ \boldsymbol{v}_{i_{rgb}} = \boldsymbol{W}\ (\mathbf{2}, \mathbf{1})\, \boldsymbol{u}_i + \boldsymbol{W}\ (\mathbf{2}, \mathbf{2})\, \boldsymbol{v}_i + \boldsymbol{W}\ (\mathbf{2}, \mathbf{3}) + \boldsymbol{W}\ (\mathbf{2}, \mathbf{4})\ /\boldsymbol{D}_i \\ \boldsymbol{D2C}_i(v_i, u_i, :) = \boldsymbol{Rgb}(v_{i_{rgb}}, u_{i_{rgb}}, :) \end{cases} \tag{1}$$

Here, (v_i, u_i) is pixel i's pixel coordinates, $Dpt\ (v_i, u_i)$ is the depth data in the corresponding coordinates, and W is the camera-related 4∗4 matrix that can be calculate from (2).

$$\begin{cases} M = [R, T] \\ W = RR * M * LR^{-1} \end{cases} \tag{2}$$

where R and T are the rotation and translation matrices; M is the external parameter matrix; and RR and LR are the internal parameter matrices of the two cameras of Kinect. They can be calculated after camera calibration using [44]. After the training, Deeplabv3 could segment the RGB image (**FIGURE 5-a**) to a color binary map (**FIGURE 5-d**). The camera distortion parameters were not considered in this study. The aligned RGB image is shown in **FIGURE 5-b** and the aligned segmented image is shown in **FIGURE 5-e**. Clearly, the frame information around the unaligned segmented image was discarded, leaving the information of the center vision after alignment. It can be seen from **FIGURE 5-f** that there are some pixel holes in the alignment image, caused by the loss of some depth information owing to the influence of uneven illumination.

2) MORPHOLOGICAL PROCESSING

After the image segmentation and alignment, detailed information about the fruits and twigs at the pixel scale was obtained. In order to determine the position of the picking branches on the two-dimensional (2D) image, the information of fruit-bearing branch in the image needed to be roughly extracted. Therefore, in this study, a morphological-based

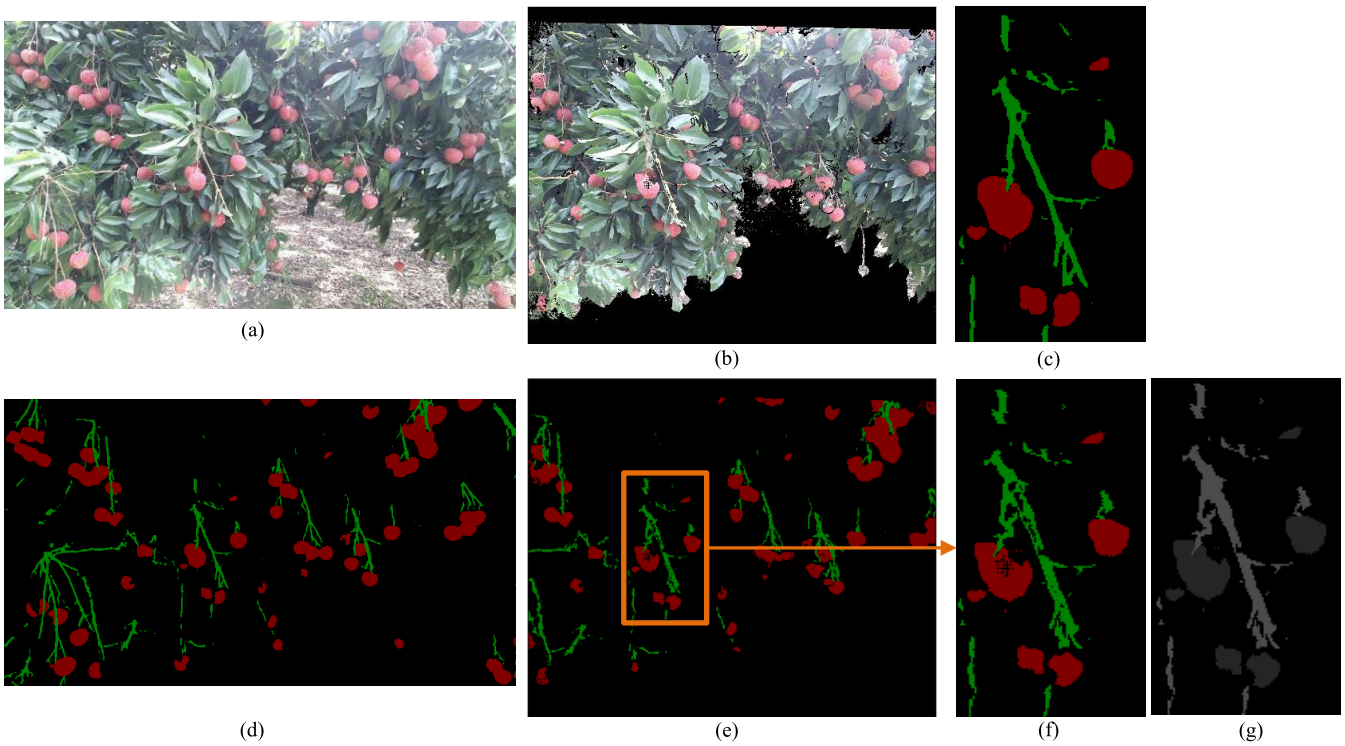


FIGURE 5. (a) RGB image; (b) Aligned RGB image; (c) Partial enlarged image of (d); (d) segmentation results of (a); (e) Aligned segmented image; (f) Partial enlarged image of (e) (marked orange); (g) Gray image after filling of the pixel holes (in segmented image, where red represents fruit and green represents twigs in groves).

algorithm for removal of fruitless twigs, their skeleton extraction, and pruning was developed to detect the effective picking branch information under a single pixel.

First, we converted the aligned segmented image to grayscale and filled the image holes using [45]; the image obtained after filling of the pixel holes is shown in **FIGURE 5-g**, Subsequently, we removed the fruitless twigs as per the following steps:

(1) Addition of a blank pixel border to the gray image of the twigs and marking of the eight connected regions.

(2) Checking the pixels in each isolated twig region; if there are pixels with the same gray value as the fruit in the 7×7 neighborhood of the pixel, such a twig region will be preserved, otherwise it will be excluded because it is considered fruitless.

(3) Repetition of (2) until all connected regions in the image have been traversed, followed by deletion of the blank pixel border.

Next, skeleton extraction and burr removal was conducted [45] to reduce the number of branch point clouds and the time complexity of the algorithm. Subsequently, a pruning algorithm was developed to remove the skeleton branches and retain the trunk/backbone of the fruit-bearing branch. As shown in **FIGURE 6-a**, the skeleton image includes three types of feature points: end points (A), corner points (B) and branch points (C), of which only branch points have branches. Therefore, in this study, the information of points A and B was preserved, and the C-type feature points were pruned to B-type feature points. For each branch point P, we defined two marker variables on its eight neighborhood, where *SUM* is the number of non-pixels on its eight neighborhood, and *FLAG* is the location of non-zero pixels. We determined the value of *FLAG* according to the location shown in **FIGURE 6-b**,

$\mathrm{P}\{FLAG\} = \{1, 2, 3, 4, 6, 7, 8, 9\}$. It is clear that $\mathrm{A}\{SUM\} = 1$, $\mathrm{B}\{SUM\} = 2$, $\mathrm{C}\{SUM\} \in [3, 8]$.

We marked the connected regions of the deburred skeleton image and then conducted pruning for each region (each region also represents one branch), as **FIGURE 6-e** shows, where each small square in the graph represents a pixel, gray indicates that the grayscale value of that pixel point is not 0, and white indicates that the grayscale value of that pixel point is 0. The search is conducted first by row and then by column. The first branch point found in the graph was C1, and we could calculate $\mathrm{C1}\{SUM\} = 4$ and $\mathrm{C1}\{FLAG\} = \{3, 6, 7, 9\}$. We then set the pixel value of the image at FLAG(3) to 0; we found that this region was divided into two regions, and the ratio of the number of non-zero pixels in the two regions was RTO{FLAG(3)}=3/7. Similarly, the values for other positions were calculated to RTO{FLAG(6)}=0, RTO{FLAG(7)}=1/9, RTO{FLAG(9)}=2/8. To delete the redundant branch at branch point C1 and make it a corner point, point or connected regions with smaller RTO ratios around point C1 should be deleted here: accordingly, the point at the FLAG(6) position was deleted, and the region with a small number of pixels was deleted at the FLAG(7) position, as shown in steps ① and ② in **FIGURE 6-e**. It is clear that only the main branch information was retained. **FIGURE 6-d** shows **FIGURE 6-c** after the pruning process. The pseudocode routines of the branches pruning algorithm are presented in **FIGURE 7**.

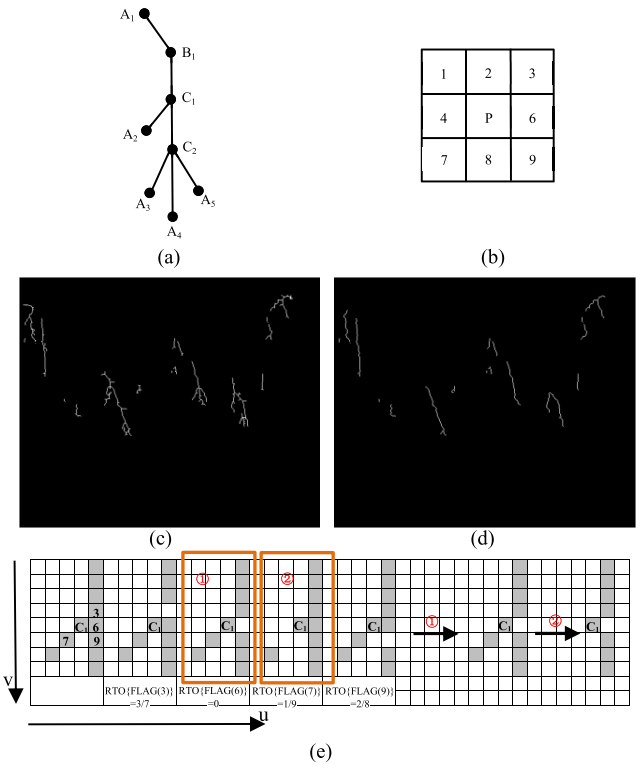


FIGURE 6. (a) Feature points of skeleton image; (b) Definition of *FLAG* value of branch point P; (c) Skeleton image; (d) Image after branches pruning; (e) Process of pruning one branch.

3) RDBSCAN CLUSTER

After the morphological processing, the image retained the main stem of litchi fruiting branch. In order to determine its spatial position, it was very important to process the 3D point clouds of the branch skeleton. In this study, the depth data between 550 mm and 1200 mm was converted to 3D coordinate according to (3).

$$\begin{cases} z_i = Dpt\,(v_i, u_i) \\ x_i = z_i(u_i - u_0)/f_x \\ y_i = z_i(v_i - v_0)/f_x \end{cases} \tag{3}$$

where (v_i, u_i) is pixel i's pixel coordinates, $Dpt\,(v_i, u_i)$ is the depth data of the corresponding coordinates, and (v_0, u_0) and (f_x, f_x) are the internal parameter of depth camera, included in *LR* matrix in Section B-1.

As shown in **FIGURE 9-c**, each branch is not a regular line in space but presents a complex curve shape, as it is difficult to directly fit a straight line. Therefore, this step was conducted to group the branch point clouds into a set of clusters, with each cluster representing one fruit-bearing

```
presented in
Algorithm bpruning (skeimg)
output=zeros(size(skeimg))
[bwlab, n]=bwlabel (skeimg)
For bwlab_i in bwlab   (i=1, 2,…, n)
   Img=bwlab_i
   For P_j in bwlab_i    (P_j!=0)
      [SUM, FLAG]=Mark_eightneighbor (P_j)
      If SUM>2
        temp=Img
        For k in SUM
          temp(FLAG(k))=0
          lab=bwlabel(temp)
          s=regionprops (lab, 'Area')
         RTO=min{s.Area(1), s.Area(2)}/max{s.Area(1), s.Area(2)}
        End For
        minpos=sort(RTO,'ascend');
        minpos(:,length(minpos)-1:length(minpos)) = [];
        For i=1:length(minpos)
          a=find(rto==minpos(i));
          Img (FLAG(a))=0;
        End For
      End If
   End For
   output=output+Img
End For
End
```

FIGURE 7. Pseudocode of pruning algorithm.

branch. Considering that the cluster is made up of a high density data cloud and some clusters are connected to each other, a revised non-parametric density-based DBSCAN clustering algorithm was adopted (inspired by [47]) to achieve more specific results. The algorithm flowchart is shown in **FIGURE 8**, where equation N =*Retrieve_Neighbors*(P, *Eps*) denotes that the data points within range *Eps* of point P are stored in N, *dist*(P_{border}, P'_{border}) denotes the distance between points P_{border} and P'_{border}, and an object whose number of points in its N set is greater than the value of *Minpts* is defined as a core object. According to this method, upon entering data set D and two important RDBSCAN parameters (scanning radius *Eps* and minimum number of included points *Minpts*), we were able to obtain the specific clusters into which this intensive data set D was aggregated as well as the noise points that did not belong to any cluster.

On analyzing the point cloud data of the picture in this study, we found that more than 70% of the images contained only about 800 points; the data scale was small but required a high degree of clustering accuracy, so an adaptive analysis method based on the K-average nearest neighbor (K-ANN) and mathematical expectations was used to determine parameters *Eps* and *Minpts* for each image [48]. First, for each point in data set D, the K-NN distance between it and its K-NN data point was calculated, following which the K-NN distance was averaged over all data points, and the K-ANN distance

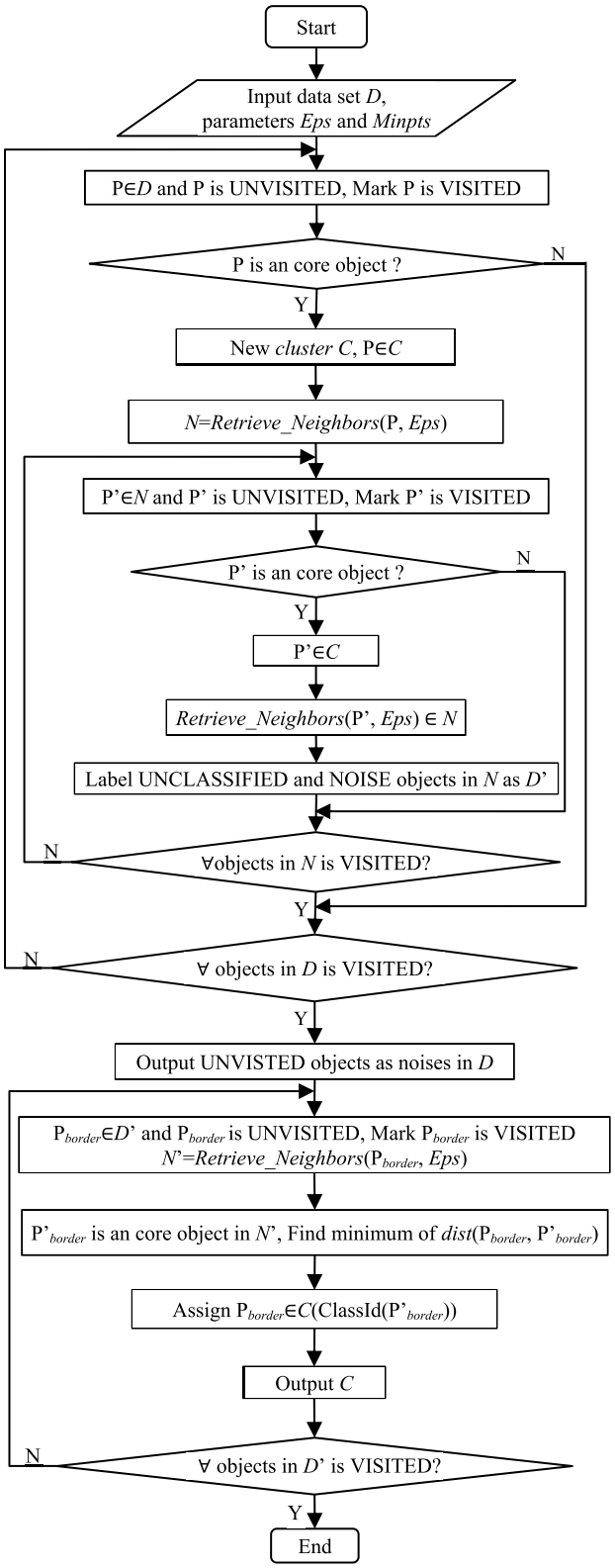


FIGURE 8. The algorithm flowchart of revised DBSCAN.

vector for all K values was calculated. This vector was the calculated list of *Eps* parameters for the RDBSCAN cluster. The calculation was performed as per the following steps:

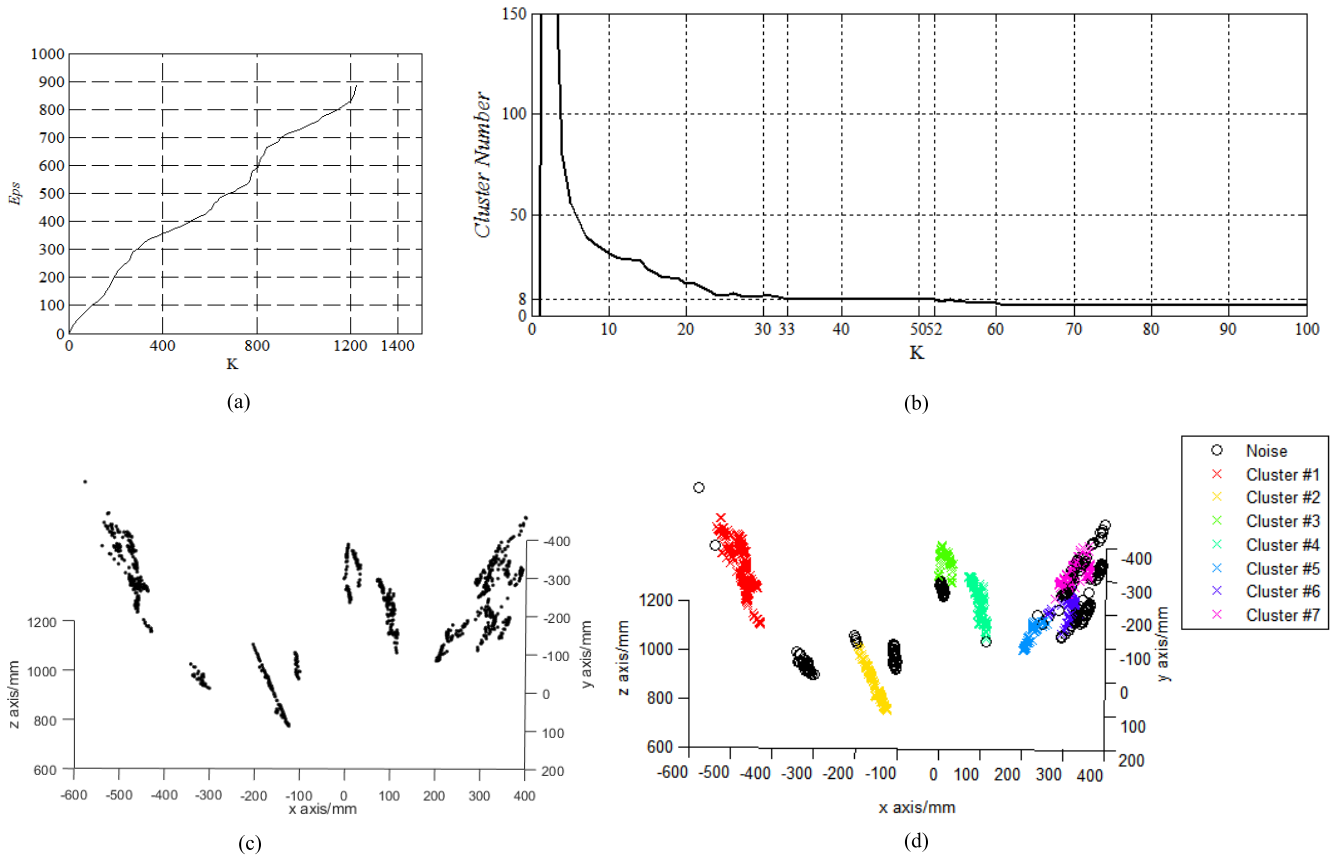


FIGURE 9. RDBSCAN cluster process and results. (a) Branch point cloud. (b) The K-Eps parameter relationship. (c) The K-Cluster Number relationship. (d) Cluster result, where each segment is marked with a random color.

Step 1: Calculation of the distance distribution matrix $D_{n\times n}$ for data set D.

$$D_{n\times n} = \{dist\ (i, j)\ |\ 1 \le i \le n, 1 \le j \le n\} \tag{4}$$

where n is the number of data points in the data set D, $D_{n\times n}$ is a real symmetry matrix of $n \times n$, and $dist\ (i, j)$ represents the distance from the i point to the j point in D.

Step 2: For each row of elements in $D_{n\times n}$ in ascending order, the elements in column K form the K-NN vector D_k for all data points.

Step 3: Calculation of the average value of the elements in D_k to determine the K-ANN distance $\bar{D_k}$ of vector D_k. For each K-value calculation, i.e., the average value of each column in D_k, the resulting vector is the list of *Eps* parameters. The K-*Eps* parameter relationship diagram is shown in **FIGURE 9-a**.

We obtained the neighborhood density threshold parameter, *Minpts*, using equation 5.

$$Minpts = \frac{1}{n}\sum_{i=1}^{n} P_i \tag{5}$$

where P is the number of *Eps* neighborhood objects of the i object.

From this, each K-value corresponds to a set of *Eps* and *Minpts* parameters, and we performed RDBSCAN clustering for each set of parameters to obtain the K-value in relation to the number of clusters.

When the generated *Cluster Number* was the same five times in a row, we assumed that the clustering result was stable, and we considered the clustering number N as the optimal clustering number, selected the maximum K value as the optimal K value, and found a set of parameters *Eps* and *Minpts* that corresponded to K. We assumed this set of parameters to be the optimal parameter pair. The K-*Cluster Number* relationship diagram is shown in **FIGURE 9-b**. As can be seen, when N = 8 and K was between 33 and 52, the clustering results tended to be stable, and we considered the parameter pair at the maximum K value of 52 as the optimal parameter, where *Eps* was 62.13 and *Minpts* was 60.

Despite the time complexity of the analysis process being relatively high, this method guarantees accuracy of the number of fruit clusters when compared to static parameters and is highly applicable to complex field environments. The cluster results for **FIGURE 9-c** are shown in **FIGURE 9-d**; the pixels in the pruned skeleton map were divided into seven clusters, with each color representing a cluster and the black hollow circle representing noise points. These seven clusters would be the premise for line fitting and location; that is, a total of seven fruit-bearing branches were extracted.

FIGURE 10. Examples of linear fitting results in aligned RGB images.

4) SPATIAL STRAIGHT LINE FITTING AND LOCALIZATION

After the clustering process, the 3D point cloud for each category represented a string of pre-harvested fruiting branches. Subsequently, via PCA, a 3D linear fit was performed for the point clouds in each cluster. The 3D data were projected into the 2D plane and then downscaled to a linear line to obtain the linear equation for the closest picking branch in space. According to [49], the data set matrix was diagonalized by a covariance matrix and projected to the dimension of principal component. For the $m \times 3$ data set D in this study, PCA straight line fitting was conducted as per the following steps:

1. Feature centralization. Subtraction of the average value of each column in data D to obtain matrix X.

2. Determination of the covariance matrix of X, $\mathrm{C} = \frac{1}{m} XX^{\mathrm{T}}$.

3. Determination of the eigenvalues and corresponding eigenvectors of covariance matrix C.

4. Arrangement of the eigenvectors into a matrix according to the descending order of the corresponding eigenvalues. The previous line is to be taken to form the matrix P (the first component). $\mathrm{Y} = PX$ is the data that is reduced to one dimension from three dimensions.

The P vector obtained via PCA is the direction vector of the space straight line in the 3D space, and the straight line is the fitted litchi fruit-bearing branch information. Through analysis and statistics, the average contribution rate of the first component of all spatial lines in the images was obtained as 78.66%, which is a reliable value for the data set containing noise. Therefore, it is reliable to use this method to locate the 3D lines. At the same time, the calculation was based on the existing functions of MATLAB, with small calculations and high practical application value. This completed the process of fitting and localization the spatial straight line. **FIGURE 10** shows some examples of linear fitting results. In the image, each line is marked with a random color, and each line contains information about the location of the fruit-bearing branch of the litchi to be picked.

III. RESULTS AND DISCUSSIONS

A. PERFORMANCE OF ALGORITHM

To evaluate the performance of the proposed method, the ground-truth location for each fruit-bearing branch needed to be measured in test dataset. We measured spatial information

of picking branches in the test image as per the following steps:

(1) Manual labelling of each target branch in the aligned RGB image using *LabelMe*; it should be noted that the fruitless and invisible branches were not included in our range.

(2) Conversion of those labelled pixels into 3D spatial coordinated, followed by PCA-based line fitting to determine the ground-truth location of the picking branches.

1) SEGMENTATION RESULTS

To measure the segmentation performance of Deeplabv3, mean intersection over union (MIOU) was used, the formula for MIOU is as follows:

$$\text{MIOU} = \frac{1}{n}\sum_{i=1}^{n}\frac{p_{ii}}{\sum_{j=1}^{n}p_{ij}+\sum_{j=1}^{n}p_{ji}-p_{ii}} \tag{6}$$

where n is the number of classes and is equals to 3 in our study. i and j are the ground truth and the predicted segmentation, respectively. p_{ij} represents the number of pixels belonging to class i but are predicted to be class j. The Deeplabv3 segmentation results from our test are shown in **TABLE 1**. Because of the small and scattered distribution of fruits and branches (particularly fruit branches) on litchi trees under natural harvesting environments, cumulative IOU could not yield a very good result regarding image pixel segmentation; the final result was 79.46%. This segmentation result was sufficient to describe the distribution and growth characteristics of branches scientifically and would not cause information loss for picking of fertile branches.

TABLE 1. Mean intersection over union (MIOU) of Deeplabv3 on our test dataset.

Method	IOU		MIOU
	Fruit	Branch	
Deeplabv3	81.27%	77.65%	79.46%

It facilitated reliable segmentation for the following localization experiments.

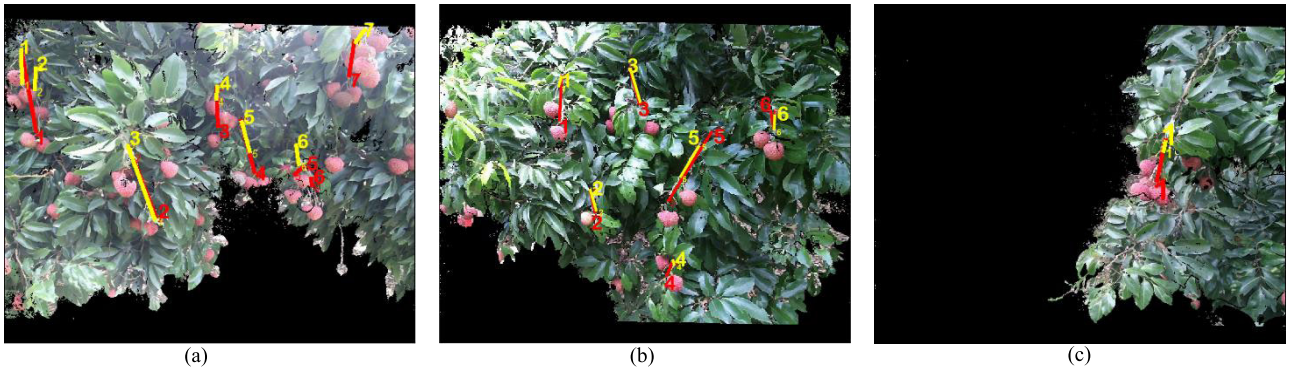
2) EXTRACTION AND LOCATION RESULTS

There are two indicators to measure the extraction and location of the picking mother branch. One is the error extraction rate, i.e., the error between the actual number of picking fruit strings and the calculated number of picking fruit strings. The other is the error between the actual spatial position of the picked fruit strings and the calculated position. There are two situations: one is that the actual picking fruit strings are lost in our calculations, which are called false positives, and the other is that the branches of the actual picking fruit strings are extracted, i.e., the branches where the actual picking fruit strings do not exist, which are called false negatives. Precision is the ratio of the true positives to the number of true fruit strings. For the second measurement, the location information of fertile branch was determined via the angle θ between the calculated line and ground-truth line. We measured this by calculating the angle between the direction vectors of two spatial straight lines, as per (7):

$$\theta = \arccos\frac{\vec{s_1}\cdot\vec{s_2}}{|\vec{s_1}|\,|\vec{s_2}|} \tag{7}$$

where S_1, S_2 represent the direction vector of the calculated line and the ground-truth line, respectively. The smaller is the value of θ, the smaller is the error. To overcome the influence of a few results with large errors, we used median and standard deviation to measure the angle deviation for the true positive and false negative spatial lines.

FIGURE 11 shows examples of fertile branch extraction printed over aligned RGB images, in which the yellow line and red line represent the true picking branches and extracted results based on the proposed algorithm. As mentioned above, it is clear that yellow line 2 depicts the false positives in the image (a). Initially, yellow line 6 also appears to be a false positive, but it is divided by red lines 5 and 6, which are false negatives, as well as the branches of fertile branch yellow line 6; therefore this line is included as a true positive. **TABLE 2** shows the experimental results of our algorithms



(a) (b) (c)

FIGURE 11. Examples of fertile branch extraction results printed to aligned RGB images. The yellow line represents the ground-truth picking branch, the red line represents the calculated picking branch by proposed method.

TABLE 2. Experiments results in test data sets.

Images	Fruit Strings	True Positives	False Negatives	Precision
90	462	385	46	83.33%

TABLE 3. Median and standard deviation of localization errors.

Images	Calculated Fruit strings	Median(degree)	Standard deviation(degree)
90	431	17.29	24.57

TABLE 4. Executed time for the proposed algorithm over test dataset.

Phase	Functions	Average times (s)
Segmentation	Deeplabv3	0.6
Pre-processing	Images align and fruitless strings removal	0.2
Post-processing	Morphological processing	1
	RDBSCAN cluster and parameters analyze	1
	Spatial straight line fitting and localization	0.1
	Total	2.9

for extraction and spatial localization errors in test data sets with a precision of 83.33%. **TABLE 3** shows the median and standard deviation of θ; it is clear that the localization error of the fertile branches for test images was 17.29°±24.57°. This location error indicates that the extraction and localization system is suitable for a litchi harvesting robot if the end-effector can tolerate this angle error.

The causes for failure were grouped into two categories.

(a) When the pruning algorithm removed the skeleton branches, it also erroneously removed the fertile branches' trunk owing to the branch pixels being greater than the number of trunk pixels. If two fertile branches are too close to each other, it is hard to separate them via RDBSCAN, so two fruit strings were observed as one branch. As shown in **FIGURE 11-a**, yellow lines 1 and 2 were clubbed with red line 1.

(b) When the fruit-bearing branch of one string fruit was very thin or invisible, Deeplabv3 is unlikely to recognize it. Hence, the branch of this fruit string could not be detected. To address these problems, the following solutions can be considered:

(1) Cultivating a new variety with fewer leaves, which would be more suitable for robotic harvesting or improving the algorithm performance;

(2) Utilizing prior knowledge of the fruit string; for example, considering the fact that the mother branch tends to be above the fruit.

3) TIME EFFICIENCY ANALYSIS

In order to ensure real-time performance of the picking process and to reduce the time complexity of the algorithm, we encapsulated the code as functions as far as possible and deployed the algorithm to run on a GPU. We measured the time elapsed for different execution steps for each image, and the results as shown in **TABLE 4**.

On combining the average time of every phase, we conclude that for an average of 6.25 fruit strings in a scene, the average execution time was 2.9 s; i.e., the processing time for a single string was 0.464 s. It takes several seconds for the robot to finish picking a fruit string, so the time performance of the proposed algorithm can ensure continuous picking operations. Therefore, this method can be applied for real-time picking of litchi strings.

B. DISCUSSIONS

While theoretical analyses and practice experiments show that the algorithm can satisfy the requirements for real-time picking in terms of accuracy and time complexity, there is still some room for improvement.

(a) Because of the random distribution of litchi fruit clusters in the field environment, several clusters close to each other would have an impact on the extraction accuracy.

(b) When fertile branches are seriously occluded and the lighting conditions change significantly (particularly in the case of strong light), the final results would be unsatisfactory. Under the shade of litchi trees, the influence of light would be considerably weakened; the interactions among many fruit strings extracted from the same scene would also be effectively reduced. The information of fruit strings that are occluded in some scenes can be extracted from the scene captured from another angle.

(c) In addition, in the field experiments, we found that obstacles such as branches and fruits may lead to the failure of the picking task on the premise that the fruit-bearing branches have been accurately localized. Therefore, it is important to study methods to avoid such obstacles during the picking process.

IV. CONCLUSION

Localization of litchi strings is crucial to complete the picking task successfully using a robot. Therefore, the extraction of fertile branches and 3D spatial localization based on RGB-D images was proposed in this study, which mainly involved the following steps:

(a) Semantic segmentation based on Deeplabv3 to segment the fruits, branches, and background in RGB images.

(b) Morphological process analysis to obtain information about the fertile branches.

(c) Revised DBSCAN clustering-based branch extraction and optimal clustering parameter analysis.

(d) Spatial straight line fitting and localization of fertile branches.

The algorithm achieved a mean intersection over union of 79.46% on using Deeplabv3. In the image, although the pixel information of the litchi fruits and twigs is small and scattered, a favorable segmentation result can still be obtained. An 83.33% extraction precision regarding the test data sets was yielded. It can therefore be concluded that the proposed method is significantly robust for spatial extraction of litchi strings. The localization error of the fruit-bearing branches was 17.29°±24.57°, which suggests that the spatial localization method was suitable for locating the fruit-bearing branches. The average execute time was 0.464 s for a single litchi string, indicating that this algorithm can be applied to practical scenarios involving a litchi-harvesting robot.

In conclusion, the proposed method can effectively extract and locate litchi fruit strings and help guide the robot to conduct the picking task continuously. A future study will focus on the success rate of the picking task.

REFERENCES

[1] T.-H. Liu, X.-R. Zeng, and Z.-H. Ke, "Design and prototyping a harvester for litchi picking," in *Proc. 4th Int. Conf. Intell. Comput. Technol. Autom.*, Mar. 2011, pp. 39–42.

[2] R. C. Harrell, P. D. Adsit, T. A. Pool, and R. Hoffman, "The Florida robotic grove-lab," *Trans ASAE*, vol. 33, no. 2, pp. 391–399, 1988.

[3] K. Tanigaki, T. Fujiura, A. Akase, and J. Imagawa, "Cherry-harvesting robot," *Comput. Electron. Agricult.*, vol. 63, no. 1, pp. 65–72, Aug. 2008.

[4] Z. De-An, L. Jidong, J. Wei, Z. Ying, and C. Yu, "Design and control of an apple harvesting robot," *Biosyst. Eng.*, vol. 110, no. 2, pp. 112–122, Oct. 2011, doi: 10.1016/j.biosystemseng.2011.07.005.

[5] J. Hemming, C. W. Bac, B. A. J. van Tuijl, R. Barth, J. Bontsema, and E. Pekkeriet, "A robot for harvesting sweet-pepper in greenhouses," in *Proc. Int. Conf. Agricult. Eng.*, Jul. 2014.

[6] C. W. Bac, E. J. van Henten, J. Hemming, and Y. Edan, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *J. Field Robot.*, vol. 31, no. 6, pp. 888–911, Nov. 2014, doi: 10.1002/rob.21525.

[7] H. Yaguchi, K. Nagahama, T. Hasegawa, and M. Inaba, "Development of an autonomous tomato harvesting robot with rotational plucking gripper," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 652–657.

[8] A. Gongal, S. Amatya, M. Karkee, Q. Zhang, and K. Lewis, "Sensors and systems for fruit detection and localization: A review," *Comput. Electron. Agricult.*, vol. 116, pp. 8–19, Aug. 2015, doi: 10.1016/.j.compag.2015.05.021.

[9] A. Koirala, K. B. Walsh, Z. Wang, and C. McCarthy, "Deep learning—Method overview and review of use for fruit detection and yield estimation," *Comput. Electron. Agricult.*, vol. 162, pp. 219–234, Jul. 2019, doi: 10.1016/j.compag.2019.04.017.

[10] Y. Tang, M. Chen, C. Wang, L. Luo, J. Li, G. Lian, and X. Zou, "Recognition and localization methods for vision-based fruit picking robots: A review," *Frontiers Plant Sci.*, vol. 11, p. 510, May 2020, doi: 10.3389/fpls.2020.00510.

[11] G. Lin, Y. Tang, X. Zou, J. Cheng, and J. Xiong, "Fruit detection in natural environment using partial shape matching and probabilistic Hough transform," *Precis. Agricult.*, vol. 21, no. 1, pp. 160–177, Feb. 2020, doi: 10.1007/s11119-019-09662-w.

[12] D. M. Bulanon and T. Kataoka, "Fruit detection system and an end effector for robotic harvesting of Fuji apples," *Agricult. Eng. Int., CIGR E-J.*, vol. 12, no. 1, pp. 203–210, 2010.

[13] W. Ji, D. Zhao, F. Cheng, B. Xu, Y. Zhang, and J. Wang, "Automatic recognition vision system guided for apple harvesting robot," *Comput. Elect. Eng.*, vol. 38, no. 5, pp. 1186–1195, Sep. 2012, doi: 10.1016/j.compeleceng.2011.11.005.

[14] T. T. Nguyen, K. Vandevoorde, N. Wouters, E. Kayacan, J. G. De Baerdemaeker, and W. Saeys, "Detection of red and bicoloured apples on tree with an RGB-D camera," *Biosystems Eng.*, vol. 146, pp. 33–44, Jun. 2016, doi: 10.1016/j.biosystemseng.2016.01.007.

[15] S. S. Mehta and T. F. Burks, "Vision-based control of robotic manipulator for citrus harvesting," *Comput. Electron. Agricult.*, vol. 102, pp. 146–158, Mar. 2014, doi: 10.1016/j.compag.2014.01.003.

[16] L. Fu, E. Tola, A. Al-Mallahi, R. Li, and Y. Cui, "A novel image processing algorithm to separate linearly clustered kiwifruits," *Biosystems Eng.*, vol. 183, pp. 184–195, Jul. 2019, doi: 10.1016/j.biosystemseng.2019.04.024.

[17] C. Wang, T. Luo, L. Zhao, Y. Tang, and X. Zou, "Window Zooming–based localization algorithm of fruit and vegetable for harvesting robot," *IEEE Access*, vol. 7, pp. 103639–103649, 2019, doi: 10.1109/ACCESS.2019.2925812.

[18] T. Botterill, R. Green, and S. Mills, "Finding a vine's structure by bottom-up parsing of cane edges," in *Proc. 28th Int. Conf. Image Vis. Comput. New Zealand (IVCNZ)*, Nov. 2013, pp. 112–117.

[19] M. Chen, Y. Tang, X. Zou, K. Huang, Z. Huang, H. Zhou, C. Wang, and G. Lian, "Three-dimensional perception of orchard banana central stock enhanced by adaptive multi-vision technology," *Comput. Electron. Agricult.*, vol. 174, Jul. 2020, Art. no. 105508, doi: 10.1016/j.compag.2020.105508.

[20] Q. Zhang and G. Gao, "Grasping point detection of randomly placed fruit cluster using adaptive morphology segmentation and principal component classification of multiple features," *IEEE Access*, vol. 7, pp. 158035–158050, 2019, doi: 10.1109/ACCESS.2019.2946267.

[21] N. M. Elfiky, S. A. Akbar, J. Sun, J. Park, and A. Kak, "Automation of dormant pruning in specialty crop production: An adaptive framework for automatic reconstruction and modeling of apple trees," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2015, pp. 65–73.

[22] Y. Majeed, J. Zhang, X. Zhang, L. Fu, M. Karkee, Q. Zhang, and M. D. Whiting, "Apple tree trunk and branch segmentation for automatic trellis training using convolutional neural network based semantic segmentation," *IFAC-PapersOnLine*, vol. 51, no. 17, pp. 75–80, 2018, doi: 10.1016/j.ifacol.2018.08.064.

[23] L. Luo, Y. Tang, Q. Lu, X. Chen, P. Zhang, and X. Zou, "A vision methodology for harvesting robot to detect cutting points on peduncles of double overlapping grape clusters in a vineyard," *Comput. Ind.*, vol. 99, pp. 130–139, Aug. 2018, doi: 10.1016/j.compind.2018.03.017.

[24] S. Amatya, M. Karkee, A. Gongal, Q. Zhang, and M. D. Whiting, "Detection of cherry tree branches with full foliage in planar architecture for automated sweet-cherry harvesting," *Biosyst. Eng.*, vol. 146, pp. 3–15, 2016, doi: 10.1016/j.biosystemseng.2015.10.00.

[25] W. Ji, X. Meng, Z. Qian, B. Xu, and D. Zhao, "Branch localization method based on the skeleton feature extraction and stereo matching for apple harvesting robot," *Int. J. Adv. Robot. Syst.*, vol. 14, no. 3, 2017, Art. no. 256010465, doi: 10.1177/17298814.17705276.

[26] A. Milella, R. Marani, A. Petitti, and G. Reina, "In-field high throughput grapevine phenotyping with a consumer-grade depth camera," *Comput. Electron. Agr.*, vol. 156, pp. 293–306, 2019, doi: 10.1016/j.compag.2018.11.026.

[27] C. W. Bac, J. Hemming, and E. J. van Henten, "Stem localization of sweet-pepper plants using the support wire as a visual cue," *Comput. Electron. Agricult.*, vol. 105, pp. 111–120, Jul. 2014, doi: 10.1016/j.compa.g.2014.04.011.

[28] Q. Feng, W. Zou, P. Fan, C. Zhang, and X. Wang, "Design and test of robotic harvesting system for cherry tomato," *Int. J. Agricult. Biol. Eng.*, vol. 11, no. 1, pp. 96–100, 2018, doi: 10.25165/j.ijab.e.20181101.2853.

[29] J. Zhuang, C. Hou, Y. Tang, Y. He, Q. Guo, Z. Zhong, and S. Luo, "Computer vision-based localisation of picking points for automatic litchi harvesting applications towards natural scenarios," *Biosyst. Eng.*, vol. 187, pp. 1–20, Nov. 2019, doi: 10.1016/j.biosystemseng.2019.08.016.

[30] J. Deng, J. Li, and X. Zou, "Extraction of litchi stem based on computer vision under natural scene," in *Proc. Int. Conf. Comput. Distrib. Control Intell. Environ. Monitor.*, Feb. 2011, pp. 832–835.

[31] C. Wang, X. Zou, Y. Tang, L. Luo, and W. Feng, "Localisation of litchi in an unstructured environment using binocular stereo vision," *Biosyst. Eng*, vol. 145, pp. 39–51, May 2016, doi: 10.1016/j.biosystemseng.2016.02.004.

[32] C. Wang, Y. Tang, X. Zou, L. Luo, and X. Chen, "Recognition and matching of clustered mature litchi fruits using binocular charge-coupled device (CCD) color cameras," *Sensors*, vol. 17, no. 11, p. 2564, Nov. 2017, doi: 10.3390/s17112564.
[33] X. Zou, M. Ye, C. Luo, J. Xiong, L. Luo, H. Wang, and Y. Chen, "Fault-tolerant design of a limited universal fruit-picking end-effector based on vision-positioning error," *Appl. Eng. Agric.*, vol. 32, no. 1, pp. 5–18, 2016, doi: 10.13031/aea.32.10701.
[34] X. Zou, H. Zou, and J. Lu, "Virtual manipulator-based binocular stereo vision positioning system and errors modelling," *Mach. Vis. Appl.*, vol. 23, no. 1, pp. 43–63, Jan. 2012, doi: 10.1007/s00138-010-0291-y.
[35] J. Xiong, R. Lin, Z. Liu, Z. He, L. Tang, Z. Yang, and X. Zou, "The recognition of litchi clusters and the calculation of picking point in a nocturnal natural environment," *Biosyst. Eng.*, vol. 166, pp. 44–57, Feb. 2018, doi: 10.1016/j.biosystemseng.2017.11.005.
[36] J. Xiong, Z. He, R. Lin, Z. Liu, R. Bu, Z. Yang, H. Peng, and X. Zou, "Visual positioning technology of picking robots for dynamic litchi clusters with disturbance," *Comput. Electron. Agricult.*, vol. 151, pp. 226–237, Aug. 2018, doi: 10.1016/j.compag.2018.06.007.
[37] Q. Zhang and G. Gao, "Prioritizing robotic grasping of stacked fruit clusters based on stalk location in RGB-D images," *Comput. Electron. Agricult.*, vol. 172, May 2020, Art. no. 105359.
[38] 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, Aug. 2016, doi: 10.3390/s16081222.
[39] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: https://arxiv.org/abs/1706.05587
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.
[41] Github. (2018). *Deeplabv3*. [Online]. Available: https://github.com/tensorflow/models/tree/master/research/deeplab
[42] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.
[43] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. (2012). *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. [Online]. Available: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
[44] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000, doi: 10.1109/34.888718.
[45] C. R. Gonzalez, E. R. Woods, and L. S. Eddins, "Morphological image processing," in *Digital Image Processing Using MATLAB*. Beijing, China: Publishing House of Electronics Industry, (in Chinese), 2012, ch. 9, pp. 255–270.
[46] G. Lin, Y. Tang, X. Zou, J. Xiong, and J. Li, "Guava detection and pose estimation using a low-cost RGB-D sensor in the field," *Sensors*, vol. 19, no. 2, p. 428, Jan. 2019, doi: 10.3390/s19020428.
[47] T. N. Tran, K. Drab, and M. Daszykowski, "Revised DBSCAN algorithm to cluster data with dense adjacent clusters," *Chemometric Intell. Lab. Syst.*, vol. 120, pp. 92–96, Jan. 2013, doi: 10.1016/j.chemolab.2012.11.006.
[48] S. Yan and Y. Jiang, "Research on method of self-adaptive determination of DBSCAN algorithm parameters," *Comput. Eng. Appl.*, vol. 55, no. 5, pp. 7–13 and 154, 2019, doi: 10.3778/j.issn.1002-8331.1809-0018.
[49] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. Cham, Switzerland: Springer, 2002.