Computers and Electronics in Agriculture 191 (2021) 106508

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

An improved obstacle separation method using deep learning for object detection and tracking in a hybrid visual control loop for fruit picking in clusters

Ya Xiong, Yuanyue Ge *, Pål Johan From

Faculty of Science and Technology, Norwegian University of Life Sciences, Ås, Norway

ARTICLE INFO

Keywords:
Agricultural robotics
Obstacle separation
Strawberry-harvesting robots
Hybrid visual loop
Deep learning

ABSTRACT

Selectively picking a target fruit surrounded by obstacles remains a challenge for fruit harvesting robots. This paper presents improvements to the active obstacle separation method for strawberry picking in clusters. A faster and more accurate vision system was developed that combined two neural networks and color thresholding for real-time detection, tracking and localization of strawberries. We propose an improved active obstacle separation method that used a push and a drag-push operation to separate the obstacles from the target in three stages. The push and drag vectors were simplified and precisely calculated based on the exact locations of obstacles. Also, different from many systems that only "looked" once for the entire picking process, the new system used a hybrid vision-based control method. In stage 1, the push operation was controlled by a simple closed-loop vision at two key points. In stages 2 and 3, the vision system re-perceived the environment to update the target and obstacle information before each round of drag-push movements. Field evaluation showed that the proposed method was more precise to separate the obstacles without reducing the speed, increasing the whole process success rate to 62.4% in clusters on the "Murano" strawberry cultivator that was 36.8% higher than the previous work.

1. Introduction

High-value crops such as strawberries, tomatoes, cucumbers, and sweet peppers ripen unevenly and require selective harvesting of only the marketable fruits, considering ripeness, size, and disease (Xiong et al., 2020a; Kootstra et al., 2021). Selective harvesting is still heavily reliant on human labor and the most labor-intensive task in the whole production (Anjom et al., 2018; Yu et al., 2020). Labor represents the highest cost and significant operational uncertainty for fruit growers (Yamamoto et al., 2014). This issue is emphasised during the COVID-19 pandemic, due to the shortage of seasonal workers (Wagner et al., 2021). Despite several attempts to develop a selective fruit-harvesting robot, a fully viable commercial system has yet to be established (Silwal et al., 2017; Xiong et al., 2020b; Arad et al., 2020). Both crop perception and manipulation are challenging tasks due to the diverse and unstructured environments and crop variations in size, shape, color, texture and pose (Tang et al., 2020; Lehnert et al., 2020; Bac et al., 2013).

Some fruits, such as strawberries and tomatoes, tend to grow in clusters. Selectively harvesting a ripe fruit that grows in clusters or is surrounded by obstacles, such as branches or leaves, while leaving the other fruits to remain undamaged on the plant, is one of the major challenges for fruit-harvesting systems (Xiong et al., 2020b; Yaguchi et al., 2016). Due to the presence of obstacles, many researchers tried to avoid them by generating a collision-free path (Lin et al., 2021) or finding a view with fewer occlusions (Lehnert et al., 2019). Unfortunately, passive obstacle avoidance does not work when the obstacles are not avoidable, especially in clusters where the obstacles may be extremely close to the target.

Our previous work introduced a strawberry-harvesting robot with a gripper that used opening fingers to surround a target and push the peduncle to the cutting area at the closing configuration, which was robust to positional errors (Xiong et al., 2020b). We also proposed using the outside of the fingers to actively push aside the obstacles close to the target based on the 3D point cloud (Xiong et al., 2020b). The active obstacle separation made it possible to pick a fruit that would otherwise be inaccessible to the robot by using obstacle avoidance methods. Based on our conception of active separation, Mghames et al. (2020) proposed a learning from demonstration (LfD) approach to generate the push movements using the same robot setup as this work. The proposed interactive primitive based planner learnt two movement primitives

* Corresponding author.
E-mail addresses: caucoexy@hotmail.com (Y. Xiong), geyuanyue@hotmail.com (Y. Ge), pal.johan.from@nmbu.no (P.J. From).

https://doi.org/10.1016/j.compag.2021.106508
Received 22 August 2021; Received in revised form 8 October 2021; Accepted 16 October 2021

from demonstrations, conditioned the resulting primitive to pass through selective obstacles and found the pushing directions. The method showed promising results in a simplified simulation environment with an ideal perception system, but was not verified in the field using a real robot. Also, this method only used push operations to clear the obstacles under the target but was not able to separate the obstacles around or above the target.

In a later work, we showed improvements to the active obstacle method (Xiong et al., 2020a). This included a sophisticated layout of point cloud blocks around the target with a new calculation method to generate more accurate separation paths, and two new polices (a zig-zag push and an in-hand drag) capable of separating the obstacles at all interest locations around the target. However, the method used predefined cuboid blocks with certain positions and numbers to represent the obstacles around the target, limiting the precision of push and drag paths. Also, using the number of points in one block to determine the presence of obstacles was inaccurate, especially for partially occluded obstacles that had few or even no points in the point cloud. It also happened that the points of several obstacles were mixed together in one block although their heights were different, or the points of one obstacle were located in two or more blocks, which led to failure picking. In this paper, both ripe and unripe strawberries were detected and localized in 3D for obstacle separation calculation. We redefined the layout of the region of interest (RoI) area and precisely separated the obstacles using push and drag-push movements based on their exact positions.

In addition, the previous work only perceived the environment once and generated the separation paths in all layers at the beginning of the picking. However, the positions of the target and the obstacles might change after the initial perception, which caused many failures. Therefore, this work proposed to use a hybrid visual loop to control the gripper for obstacle separation. A closed visual loop was used to control the push operation at two key points in the bottom layer, and a continuous "look-and-move" was used to determine a new round of drag-push operation in central and top layers. To make the control possible, we developed a faster vision system that combined YOLOv4 for object detection and Deep SORT for ripe berry tracking.

2. Materials and methods

2.1. Robot overview

As shown in Fig. 1, the field tests were performed on our previously developed U-shaped strawberry-harvesting robot (Xiong et al., 2020a). The robot mainly consists of a U-shaped platform and two independent picking systems mounted on either side of the arch, allowing strawberries to be picked on both sides of the table. Changing ambient illumination in the field is a challenge for image processing. The robot was designed to pass through the strawberry table to avoid ambient illumination, covering the entire plants and the picking systems. The picking system (Fig. 1(b)) included an RGB-D camera (D435; Intel, USA), a laboratory-developed SCARA-type arm, a previously developed gripper, and a LED panel (VT-2407; V-TAC, Bulgaria). The three degrees of freedom arm had two rotation joints and one linear vertical axis using the same motors and control strategy as the old Cartesian arm (Xiong et al., 2020b). In the 2020 picking season, we improved the structure of the arm links and redesigned arm's working space, making it possible to pick with both arms in a limited tunnel row space.

2.2. Perception

Our previous work used an instance segmentation convolutional neural network Mask Region-Convolution Neural Network (R-CNN) to identify and segment strawberries at pixel level (Ge et al., 2019). Compared with other popular object detection networks, such as You Only Look Once (YOLO) (Bochkovskiy et al., 2020), the instance segmentation was more accurate on 3D localization of the targets, since it generated the boundaries of the targets while detection networks only outputted the bounding boxes that might contain pixels from other objects. However, the Mask R-CNN method required ample computational resources and was too slow for a closed-loop control system. In our applications, the average processing time for one image frame, including running the detection network, coordinate transformation and other computations was 0.82s on GTX 1060 GPU (Ge et al., 2019). Recent development in the YOLO network (YOLO version 4, YOLOv4) has made significant progress in both detection accuracy and execution speed (Bochkovskiy et al., 2020). This paper presents a new vision system that combined YOLOv4, Deep Simple Online and Realtime Tracking (SORT),

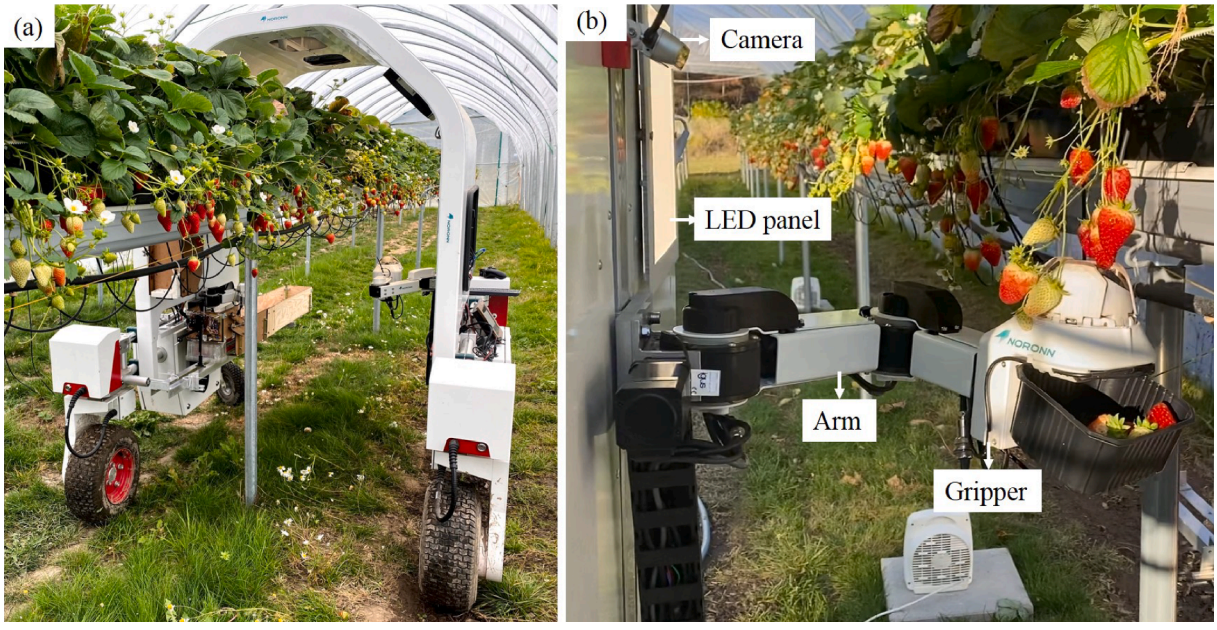

Fig. 1. The U-shaped strawberry-harvesting robot on a farm: (a) overview of the whole system; (b) inner view of the picking system.

and color thresholding for strawberry detection, tracking and localization in real-time.

2.2.1. Detection

The training and validation datasets were collected from the Boxford Suffolk Farms (England) and a university experimental tunnel at the Norwegian University of Life Sciences in 2018 and 2019, respectively. As we used transfer learning, the total dataset contained 280 images for training and 55 images for validation. Around one-third of the images were captured by the integrated camera of a mobile phone (iPhone 6s, Apple, USA) at Boxford Farms, and the remaining were collected using an RGB-D camera (D435, Intel, USA) on the strawberry-harvesting robot at the university tunnel. The cultivators of the strawberries were "Lusa" from the Boxford Farms and "Murano" from the university tunnel. We defined two classes of strawberries for annotation, ripe and unripe. Most of the images contained a great number of strawberries. A total of 5,983 ripe strawberries and 3,500 unripe strawberries were annotated in the dataset, representing 28.3 berries per image. To reduce the annotation time, we used our previously trained YOLOv3 model to generate the bounding boxes of the strawberries and then manually refined them using an annotation software Lableme (Wada, 2016), including bounding box refinement and correction for undetected and incorrectly detected objects.

The strawberry detection model was obtained through transfer learning. The annotated strawberry dataset was used to fine tune the weight parameters of an official released YOLOv4 model (*yolov4.conv.137*) that was pretrained on the Common Objects in Context (COCO) image dataset (Lin et al., 2014; Bochkovskiy et al., 2020). The images were resized to 416 × 416 pixels for training and the learning rate was set to 0.001 with a momentum of 0.949. The model was then trained for 20,000 iterations in Darknet framework (Bochkovskiy et al., 2020), ending with a loss value of 2.24.

To avoid picking unripe strawberries, the confidence rate was set relatively high at 0.7 in real applications. The robot control system used ROS architecture, so we modified the official Darknet YOLOv4 package (Bochkovskiy et al., 2020) to a ROS YOLOv4 package. Initially, Realsense ROS package was used to grab the RGB and depth images from the camera and publish them as ROS topics that were subscribed by the YOLOv4 package. However, this method significantly reduced the cycling speed caused by the non-synchronized image publishing and image detection and many unusable functions, such as point cloud generation and publishing in the Realsense package. Therefore, the YOLOv4 ROS package was modified to call the camera and process the images directly using the Pyrealsense2 library. The images were acquired only when the network started a new round of detection, thus avoiding unnecessary image publishing. A resolution of 640 x 480 pixels was used for both RGB and depth images. The loop rate of this method achieved 22 frames per second (FPS) on the robot computer (GTX 1060 6 GB GPU). Fig. 2 shows the detection results of a complex situation on the images captured by the robot camera (D435) in the field, where the class name was shown at the top of the bounding box and the confidence rate was given at the bottom of the box. In general, the new model showed good performance for strawberry detection, with high confidence scores on most objects even on highly occluded strawberries, although some small and occluded unripe berries were not detected (Fig. 2(a)). Fig. 2(b) shows an example of the picking procedures, in which a target ripe berry was continuously detected, even those partially swallowed by the gripper.

2.2.2. Tracking

Deep SORT (Wojke et al., 2017) is an improved version of the classic Kalman filter-based tracking method SORT (Bewley et al., 2016) for multiple object tracking. The key improvement of Deep SORT was that it employed a two-layer convolutional neural network for appearance matching among different image frames, which reduced the failures of identify switches and increased the possibilities of maintaining identities through longer occlusions. This was useful for strawberry picking since the target berry might be occluded by other objects or even the gripper during the picking operation. Continuous re-identification of objects can help the robot pick on a specific target, avoiding capturing other obstacles. Deep SORT was mostly used for people tracking and re-identification. To apply it to strawberry picking, it was necessary to train the deep appearance descriptor.

The training dataset was required to have connected image frame sequences in videos where each instance of a moving object was identified for each frame. Annotating each object in a number of video frames was a laborious work. Using Deep SORT without appearance descriptor was also able to track other objects but with lower accuracy. We therefore used the YOLOv4 detection network combined with Deep SORT to roughly detect and track all the strawberries in videos that were recorded by the robot camera D435 during the previous picking. The picking operation moved many berries, which met the requirements of

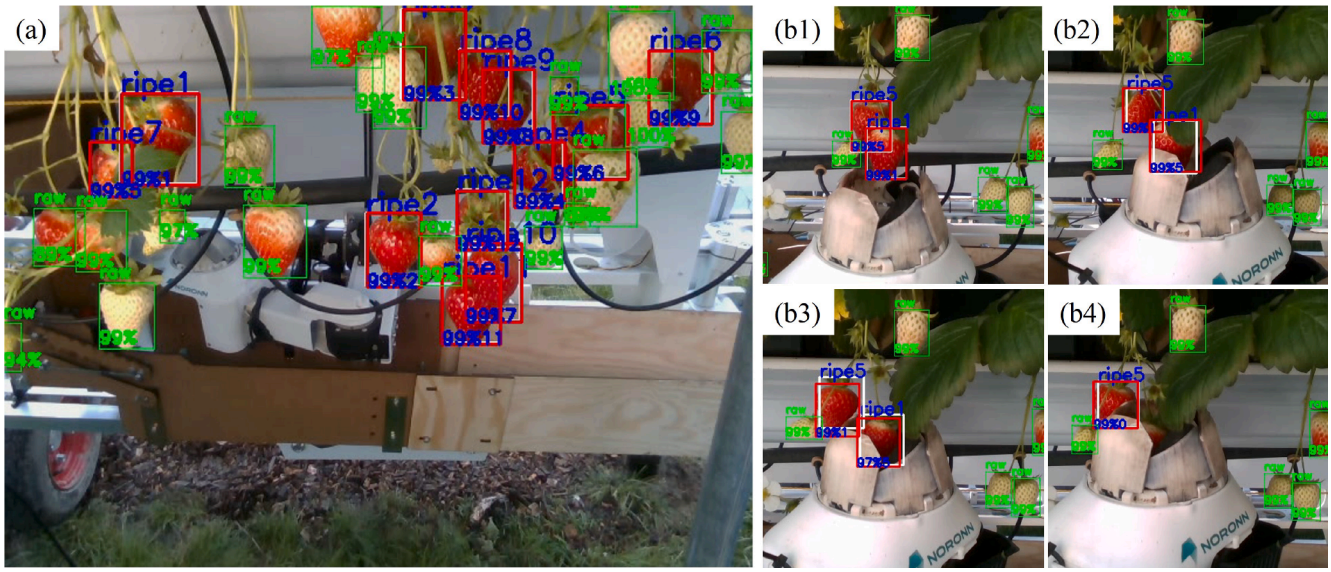

Fig. 2. Strawberry detection and tracking using YOLOv4 and Deep SORT, the numbers on the top of the ripe berry bounding boxes were the tracking IDs: (a) a complex situation with many occlusions, some small and occluded unripe berries were not detected; (b) picking procedures of ripe 1 using a drag-push operation, where the target was still detected and tracked in partially captured situations.

the tracking dataset and was the same situation in applications. The detected berries were cropped out from the video frames, and all the frames of a tracked target were placed into a folder forming an identity. Only correctly detected and tracked objects were manually selected for training. The total dataset contains 110 identities and 24189 cropped images, with an average of 220 images per identity. All the cropped images were resized to 128 × 256 pixels for training. We followed the steps in the metric learning paper for training (Wojke and Bewley, 2018). The model was trained to a loss value of 2.48 using a learning rate of 0.001.

The Deep SORT function was added to the YOLOv4 ROS package to combine detection and tracking, becoming a new Deep SORT YOLOv4 ROS package (Deepsort_yolov4). As the flowchart shows in Fig. 3, the detected bounding boxes of the objects from the YOLOv4 ROS module and the current RGB image were sent to the Deep SORT function for multiple object tracking. The Deep SORT function outputted the predicted bounding boxes and tracking ID numbers. The processing speed of the Deep SORT function was negatively correlated to the number of tracked objects. The robot only picked ripe strawberries, while the unripe strawberries were detected to calculate obstacle separation, so only ripe strawberries were selected for tracking. As shown in Fig. 2, the tracked ID numbers of ripe strawberries were placed at the top of the bounding boxes. In Fig. 2(b), the target was continuously tracked with ID 1 even it was partially swallowed by the gripper, allowing the system to focus on the same target during the picking procedures.

2.2.3. Localization

The robot required 3D bounding boxes of both ripe and unripe strawberries for picking manipulation. Our previous work used instance network Mask R-CNN that outputted RGB masks of an object in pixels without other objects, so the 3D locations were easily obtained by matching the RGB masks with corresponding depth images (Ge et al., 2019). YOLOv4 outputted 2D bounding boxes of the targets, which may contain pixels from other objects with different depth values. To remove these noise pixels, a two-step color thresholding was implemented to filter non-red pixels for ripe berries and non-green pixels for unripe strawberries. For instance, in Fig. 4(a), ripe berries 2 and 4 were occluded by front green branches, leaves, and berries removed by non-red pixel thresholding. However, this method was unable to filter the occlusions with the same colors as the objects. Thereafter, the mean depth value of all the filtered pixels in the bounding box was used as the depth of the object. The top left and bottom right points of the 2D bounding box were de-projected into 3D points in the camera frame using the mean depth value, obtaining the length and height of the 3D bounding box of a target. In addition, since strawberries are mostly symmetrical, the length of the berry was also used as the width, thus obtaining a 3D bounding box in the camera frame (Fig. 4(b)). After that, the coordinates were transformed from the camera frame to the robot arm frame based on camera extrinsic calibration. More details about coordinate transformation can be found in our previous work (Ge et al., 2019).

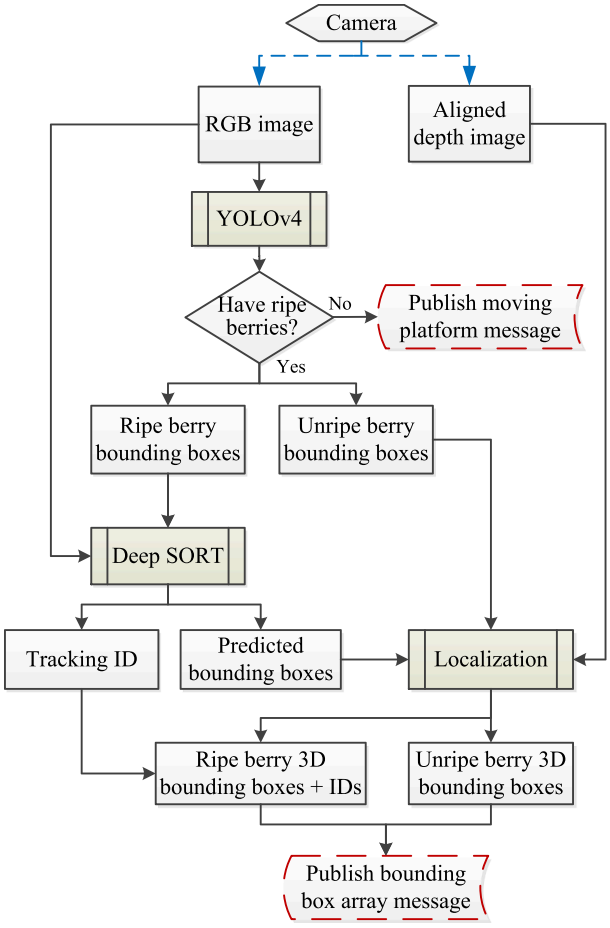

Fig. 3. Flowchart of the perception system.

2.3. Improved active obstacle separation

2.3.1. Region of interest and layout

The obstacle separation paths were generated according to the visual perception of the obstacle information around the target. Similar to the previous method, we considered that the obstacles located in a region of interest (RoI) area around the target might cause picking failure. These obstacles might be wrongly captured by the gripper during picking or prevented the target from being captured by the gripper. The RoI area was thus used to calculate the separation paths based on the distribution and number of the obstacles (Xiong et al., 2020a). The RoI comprised a volume of the 3D point cloud that contained the target fruit and potentially one or more obstacles. As shown in Fig. 5, the RoI area included a top layer above the target, a central layer encompassing the target and a bottom layer under the target. The gripper was instructed to separate obstacles in three stages in three layers accordingly. As the gripper picked from below, the gripper may push aside obstacles horizontally within the bottom layer during the first stage. During the second stage, the device moved upwards to swallow the target and might drag the target to avoid obstacles and then pushed the obstacles back within the central layer. Finally, the third stage performed a similar drag-push motion as the second stage but used a greater magnitude of the motion. The detailed separation policies will be elaborated in the below sections.

The radius of the layer was determined according to the opening radius r_{gri} of the gripper aperture for picking. This was because the gripper might wrongly capture the obstacles located within the opening size. Considering having a large space for pushing and dragging and the limitation of drag distance in the central layer, the radius of bottom and r_{bot} central layers r_{cen} were twice that of the gripper opening radius r_{gri}. In the central layer, as the gripper partially swallowed the target fruit, the drag distance was limited to prevent the target from slipping out of the gripper. In the top layer, the target was fully enclosed in the gripper, so it can be dragged to a further position if needed. Therefore, the radius of the top layer r_{top} was three times of gripper opening radius r_{gri}. The height of the central layer was equal to the strawberry bounding box height, while the heights of the bottom layer h_{bot} and top layer h_{top} were fixed, 80 mm and 40 mm, respectively.

2.3.2. Stage 1: push in the bottom layer

As the gripper swallowed berries from below, to avoid capturing the obstacles under the target, it was necessary to clear them before the gripper opening fingers to swallow the target. Our previous work

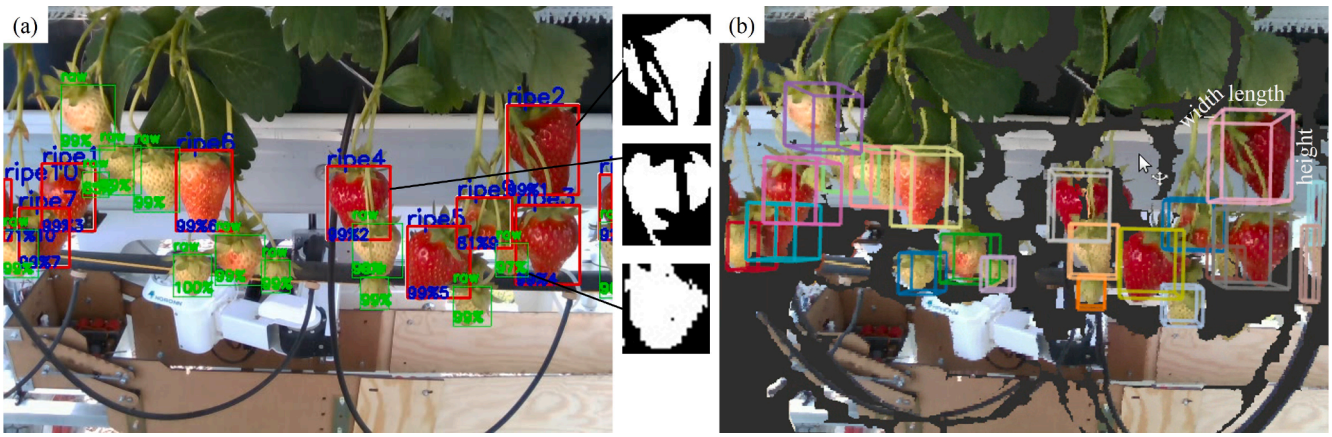

Fig. 4. Localization: (a) results of detection and tracking, the binary images show to use the two-step color thresholding to remove the noise pixels from the branches and other obstacles in different colors; (b) localization results in the 3D point cloud. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

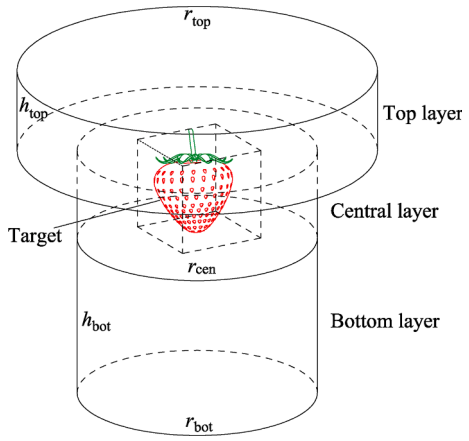

Fig. 5. Region of interest area around the target for obstacle separation.

proposed a single linear push to move the obstacles out of the way in a simple situation and a zig-zag push for a more complex situation (more obstacles), which was demonstrated to be effective (Xiong et al., 2020a). A zig-zag push was a motion where the gripper used a zig-zag movement that contained several linear motions to push the obstacles side to side. This work remained the pushing policies but proposed several improvements that made the method more precise and robust. One of the main improvements was that a closed visual loop was used to control the gripper for obstacle separation in the bottom layer, thanks to the implementation of object tracking. Another improvement was the calculation method of push direction and magnitude.

Fig. 6(a) and (b) show that a single push was used to push aside obstacle 1 under the target. Obstacle 1 might be captured if the gripper moved up directly to swallow the target. The single push first instructed the gripper to move to the right side of the target bottom P_0 (Fig. 6(b)). Second, the gripper pushed from P_0 to the center of the target bottom P_1. To reduce the possibility of the gripper touching the bottom of the target due to inaccurate localization, P_0 and P_1 were lower than the bottom of the target bounding box ($offset_{push}$, using 1 mm in this work). Until then, the fingers remained closed to avoid receiving any obstacles. The final action in the bottom layer was that the gripper opened fingers at the P_1 position (Fig. 6(b)).

During this stage, the gripper was closed-loop controlled based on visual feedback to reach at P_0 and P_1, respectively. Fig. 7 shows the control diagram for reaching at position P_0. The system used the camera to perceive the current environment to detect, track and localize the target and obstacle strawberries. The 3D bounding boxes were sent to the obstacle separation algorithm to determine the newest and desired gripper position P_0. Meanwhile, the system read the current gripper position and subtracted it from the desired position obtaining an error or difference of the gripper position. If the positional error was greater than a threshold, it was further converted into arm joint errors for motor speed adjustment. Considering the relatively high tolerance of the gripper and slow reacting speed of the camera-arm system, a simple proportional controller was used. The strawberries were not always stable, and the depth sensing of the camera generated about 1–2 mm errors among different image frames, so we used relatively large thresholds for quick settlement. The method was less sensitive to the positional error at P_0 point as this was a push-starting point without strawberry capturing, so the threshold was set to 10 mm. The control of reaching at P_1 point included all the procedures in Fig. 7, except for the obstacle separation function, because a push motion was already executed. A smaller threshold of 5 mm was used for P_1 point for more accurate positioning, since the gripper opened fingers to capture the fruit at this point.

The closed-loop feature was helpful when the strawberries were moved by previous picking of other berries or by wind, which accounted for approximately 34% of the failures before manipulation for the variety of "Malling Centenary" (Xiong et al., 2020a). Also, the target position might change when pushing aside obstacles from P_0 to P_1, especially when they grew on the same stem. This made more than 50% of the failures in the bottom layer in the previous open-loop system (Xiong et al., 2020a). A closed visual loop enabled the system to find the newest P_1 position. It also increased the success rate in the bottom layer when the mobile platform was moving or shaking, thus improving the system's robustness.

A high tolerated solution without pushing was used for the situation where no obstacles were detected in the bottom layer. The gripper moved directly to a lower point P_{1_l} that was vertically under P_1 (30 mm was used in this work) and then opened fingers at P_{1_l}. This method gave a high tolerance to the z-direction of the target position. Furthermore, if the target was isolated, with no obstacles in all layers, the gripper moved to P_{1_l} as well but opened fingers to a larger angle, making it more tolerated to all directions of the localization errors.

2.3.3. Stage 2: Drag-push in the central layer

After stage 1, the gripper moved up to separate the obstacles from the target in the central layer. Our previous work introduced an upward zig-zag push operation to push the surrounded obstacles side to side while the gripper moved upwards (Xiong et al., 2020a). This method worked in many cases but had low precision, long operation time and may

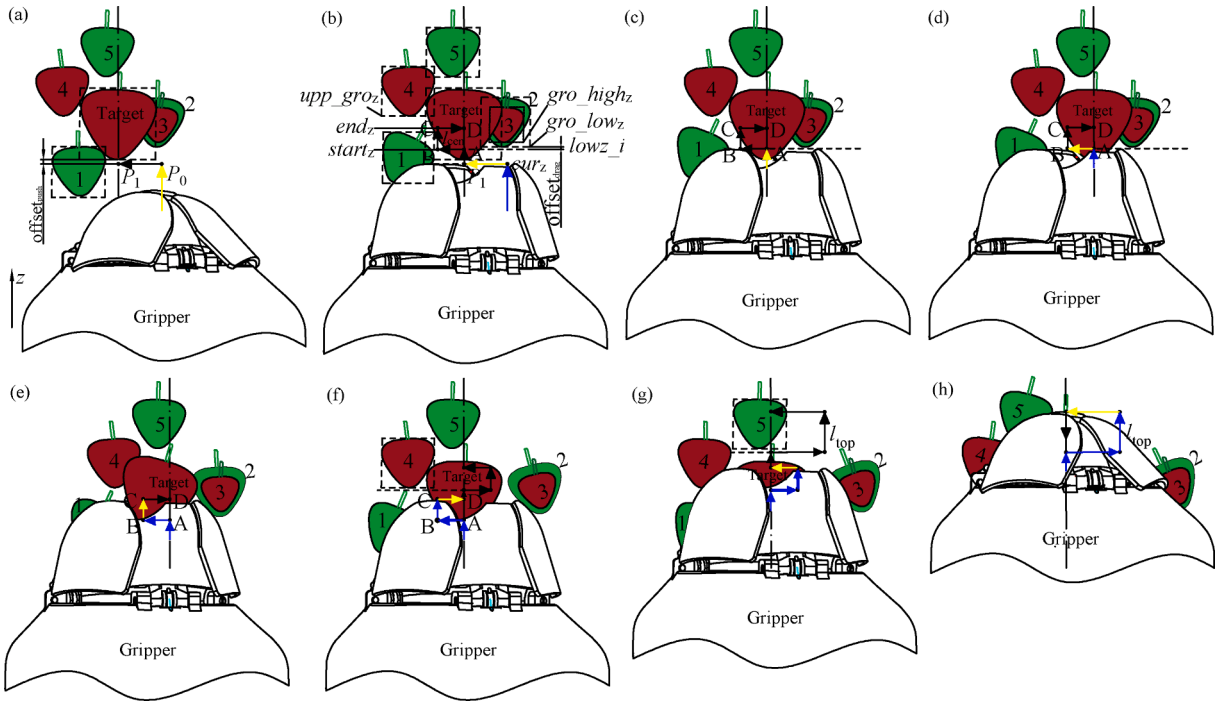

Fig. 6. Diagram of using push and drag-push movements to separate the obstacles from the target during picking: (a) the gripper is moving upwards to P_0 before pushing obstacles; (b) pushing aside obstacle 1 in the bottom layer, arriving at P_1, then opening fingers; (c) moving upwards to A; (d) dragging the target to B to avoid capturing obstacles 2 and 3 in the central layer; (e) moving upwards to C to enclose more part of the target; (f) pushing back to reduce the contact force generated from the inclined peduncle; (g) using the same drag-push motion to separate obstacle 4; (h) using drag-push motion to separate obstacle 5 in the top layer, closing fingers and moving down for fruit detachment.

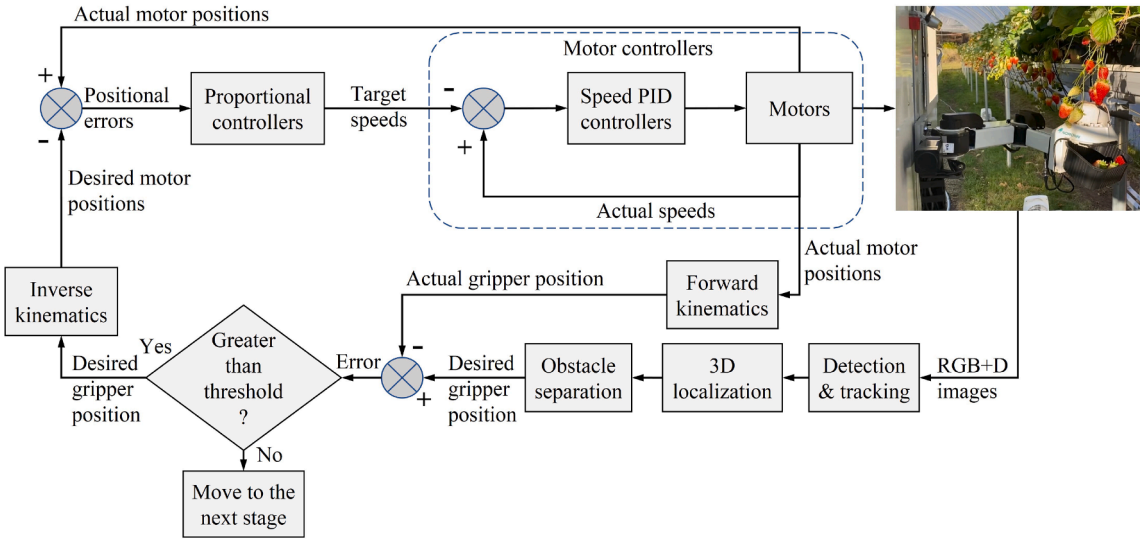

Fig. 7. Closed-loop position-based visual-servoing control in stage 1.

increase the damage to the fruit. The reason for the low precision was that the predefined point cloud blocks used to determine the presence of obstacles had certain positions and numbers, so the gripper may miss the obstacles if it was located at a higher position in a block. Also, the old system only perceived the environment and generated the separation paths in all layers at the beginning of the picking. The environment might be changed when operated in the central layer. In this work, the separation policies were simplified and unified in the central and top layers, using a drag-push operation to separate the obstacles from the target. As shown in Fig. 6(c), the drag-push started with a vertical upward motion to swallow the target from P_1 to the bottom of the lowest obstacle (2) at point A. After that, to avoid capturing obstacles 2 and 3, the gripper dragged the target horizontally to a position that contained fewer obstacles at point B (Fig. 6(d)). At this position, the gripper continued to move vertically upwards to enclose more of the target at point C (Fig. 6(e)). At point C, the peduncle might be inclined such that the target was difficult to fall due to the static contact force with the gripper fingers and easily be damaged when the gripper moved upwards further towards a cutting position. Hence, the gripper returned to the original central line of the target at point D while pushing the obstacles to the right (Fig. 6(f)). To avoid generating too many push and drag movements over a short distance, closely distributed obstacles were

regarded as a group (obstacles 2 and 3 were in one group in Fig. 6). The push-drag operation was implemented once for a group of obstacles. As the gripper picked from below, the bottom position *lowz* of an obstacle bounding box was used for the group determination. First, the obstacles in the central layer were sorted from bottom to top and labeled as $lowz_0$, $lowz_1$ and so on. The upper obstacles were grouped together if the vertical position was not higher than 10 mm of the nearest below obstacle:

$$lowz_{i+1} - lowz_i < 10, \quad i = 0, 1, \ldots, n \tag{1}$$

As shown in Fig. 6(b), the height of the pushing back motion (end_z, height of points C and D) was critical. In general, end_z should be very close to the lowest obstacle among the upper group upp_gro_z, so the gripper had the highest possibility to push aside all the obstacles in the current group. However, the gripper may move the obstacles unintentionally from the upper group due to the localization error in *z*-direction. Therefore, it was more tolerated to use a lower end_z when upp_gro_z was higher enough than the highest obstacle (gro_high_z) in the current group or there was only one group in the central layer. A threshold of 15 mm ($step_z$) was used to determine the value of end_z in this work. The abovementioned strategy can be concluded as follow:

$$end_z = \begin{cases} gro\_high_z + step_z/2, & upp\_gro_z - gro\_high_z > step_z \quad or \quad num_{\text{group}} = 1 \\ upp\_gro_z - offset_{\text{push}}, & upp\_gro_z - gro\_high_z <= step_z \end{cases} \tag{2}$$

Fig. 8 displays the control process of the method in stage 2. The perception system grabbed the latest RGB-D images and sent the updated positions and IDs of the target and obstacles to the obstacle separation function. The obstacle separation function calculated and obtained a set of push-drag operation paths. Then, the manipulator was instructed to implement the drag-push operation, traveling from point A to B, C, and D, respectively. In the meanwhile, a parallel thread was used to check if the gripper received the target using the gripper internal IR sensors (see the gripper design in (Xiong et al., 2020b)). After completing the push-drag operation in one group, the system continued to separate the obstacles in the upper group if the gripper had not passed the central layer. The perception system updated the target and obstacle information for planning the separation of each obstacle group. For example, as shown in Fig. 6(f), after finishing the drag-push operation for the first group, at point D, the system recaptured the images and generated a new set of drag-push paths (black arrows) for the upper group (obstacle 4). Continuous "look-and-move" provided more accurate information compared to the old system. Also, in this stage, the target might not fall into the gripper smoothly when the gripper moved upwards, because the gripper might push up the target and obstacles if they grew on the same stem or the static contact force between the target and the connected obstacles was difficult to break (Xiong et al., 2020a). The current system was able to perceive the new situation and perform appropriate actions continuously until reaching the arm's working space limits, which increased the possibility of successful picking. If the gripper did not detect the target inside it for more than five rounds of the push-drag operation, the system would skip the current target and move to the next one. It also happened when the gripper failed to swallow the target due to localization errors.

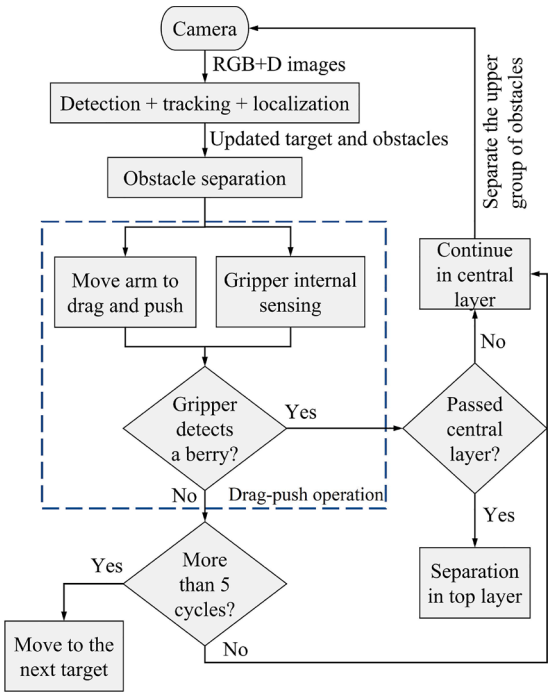

Fig. 8. Continuous "look-and-move" control in stage 2.

Unlike in stage 1, the control of the drag-push operation crossing points A, B, and C to D was a visual open loop. One reason was that points A, B, C, and D were determined according to the number and locations of the obstacles that might have obvious differences over image frames due to the picking operation. Strawberries swung quickly in the picking operation, which made the current system hard to estimate the position changes in a short time due to the latency of the image processing. The central layer might contain several rounds of drag-push operation, so using a visual closed-loop method to reach each point might significantly increase the operation time.

During stage 2, the target might be lost in detection and tracking when partially swallowed by the gripper. In this case, the latest target position from the previous image frame would be used for separation calculation, but the obstacle information was still updated in the current image frame. Thus, the target position was only updated when it was visible and trackable.

2.3.4. Stage 3: Drag-push in the top layer

After stage 2, the gripper continued moving upwards to provide more space for finger closure and fruit detachment in stage 3. Obstacle separation in stage 3 was similar to stage 2, using the drag-push operation but with a greater drag magnitude. As shown in Fig. 6(g), after finishing the obstacle separation in stage 2, the system re-perceived the scene and determined to use a drag-push operation to move aside the top layer of obstacle 5 to avoid capturing it. In stage 3, the target was fully enclosed in the gripper, so it could be dragged to a farther position without slipping out from the gripper. In addition, dragging the target to a farther location could increase the likelihood of finding a position with fewer obstacles. When one round of drag-push operation was finished, the gripper closed fingers at the final position and then moved down 30 mm for fruit detachment (Fig. 6(h)). The newly added moving down motion was simple, but a robust mechanical method to keep the peduncle that remained on the fruit short and neat.

It was also worth reminding that to avoid cutting the fruit body, the gripper internal sensors were used to check whether the target had

passed the cutter when the target was not visible from the camera in stage 3. This was because that in some cases, the fruit did not fall further for detachment when the static contact force between the target and the gripper was greater than the fruit weight or when the gripper pushed up the branch that connected to the target. On the other hand, a continuous upward movement may enable the fruit to fall for detachment. Therefore, the system would perform another drag-push round until the fruit fell or reached the arm limits. The control diagram of stage 3 was similar to Fig. 8 but replaced "gripper detects a berry" and "more than 5 cycles" with "gripper checks whether the target has passed the cutter" and "reach arm limits", respectively.

2.3.5. Calculation of push and drag vectors

This work simplified the calculation of push and drag vectors, making it possible to use the same formulas to derive both push and drag vectors in all stages. We used the latest distribution of obstacles in the current image frame to generate the push or drag motions, without the need of tracking the obstacles. The push vector $\boldsymbol{D}_{\text{push}}$ referred to the movement of $\boldsymbol{P_0P_1}$ in stage 1, and the drag vector $\boldsymbol{D}_{\text{drag}}$ referred to $\boldsymbol{AB}$ in stages 2 and 3 (Fig. 6). Fig. 9 shows the diagram of the calculation of the push and drag vectors. The central red circle represented the target that was surrounded by four obstacles, circle B_1, B_2, B_3 and B_4. The bold black circle was the ring formed by the gripper fingers at the opening position. In this work, a two-step calculation was used. First, we defined a risk circle to determine if the target was isolated at the current layer or group. That was, if the distance between the obstacle and the target $dis_{\text{ob\_tar}}$ was shorter than the radius of the risk circle r_{risk}, the target was considered not isolated so a push or drag was required. This was because that obstacles located within the risk circle might be captured by the gripper or stopped the gripper from enclosing the target. The radius of the risk circle r_{risk} equaled the gripper opening radius r_{gri} plus a tolerance Δ (8 mm was used in this work). The outer dashed circles show the top view of the bottom, central and top layers. Second, if the target was not isolated in the current layer/group, all the obstacles located within the layered circle were used for push or drag vector calculation. To increase the possibility that the gripper pushed all the obstacles out of the way and had less impact on the target, the gripper should move from an "entrance" that contains fewer obstacles towards the target center. Drag operation, on the other hand, took the target from the original center of the target to a place with fewer obstacles. As shown in Fig. 9(a), obstacles B_1 to B_4 were located inside of the bottom layer and the angle between the nearest obstacles around the target was labeled as θ_1, θ_2, θ_3 and θ_4 in clockwise. The "entrance" was the maximum angle θ_{max} among these obstacle angles (θ_1 in Fig. 9). The normalized drag vector $\boldsymbol{D}_{\text{drag\_N}}$ can be expressed as:

$$\boldsymbol{D}_{\text{drag\_N}} = \frac{\boldsymbol{OB}_L}{|\boldsymbol{OB}_L|}\left(\frac{\theta_{max}}{2}\right) = \begin{bmatrix} \cos\frac{\theta_{max}}{2} & -\sin\frac{\theta_{max}}{2} \\ \sin\frac{\theta_{max}}{2} & \cos\frac{\theta_{max}}{2} \end{bmatrix} \frac{\boldsymbol{OB}_L}{|\boldsymbol{OB}_L|}, \quad \theta_{max} = \max\left(\theta_1, \theta_2, \ldots, \theta_i\right) \tag{3}$$

where, B_L was the centroid of the end obstacle in a clockwise direction at the "entrance" (B_1 in Fig. 9). The normalized drag vector $\boldsymbol{D}_{\text{drag\_N}}$ was obtained by rotating the normalized vector of $\boldsymbol{OB}_L$ counterclockwise by half of θ_{max}. The push vector $\boldsymbol{D}_{\text{push}}$ was opposite to the normalized drag vector $\boldsymbol{D}_{\text{drag\_N}}$ and had a fixed magnitude of r_{bot} so that it can be expressed as:

$$\boldsymbol{D}_{\text{push}} = -r_{\text{bot}}\boldsymbol{D}_{\text{drag\_N}} \tag{4}$$

As for the drag vector, the magnitude m_{drag} was calculated based on the obstacle locations. As shown in Fig. 9(b), all the obstacles in the central layer were added with a risk circle. The obstacle risk circle may have one or two intersection points with the line segment OT along the drag vector. The intersection point was marked as E_1, E_2, …, and E_i. The gripper dragged the target to the farthest intersection point to O (OE_{max}) to reduce the risk of capturing obstacles. However, considering that the target may slip out of the gripper for long-distance drag, the magnitude m_{drag} was limited to r_{gri} in the central layer and $2r_{\text{gri}}$ in the top layer.

$$m_{\text{drag}} = \begin{cases} \overline{OE}_{\max}, & \overline{OE}_{\max} <= \text{kr}_{\text{gri}} \\ \text{kr}_{\text{gri}}, & \overline{OE}_{\max} > \text{kr}_{\text{gri}} \end{cases}, \quad \overline{OE}_{\max} = \max\left(\overline{OE}_1, \overline{OE}_2, \ldots, \overline{OE}_i\right) \tag{5}$$

where, k was 1 in the central layer and 2 in the top layer, respectively. The drag vector D_{drag} was then obtained by:

$$D_{\text{drag}} = m_{\text{drag}}D_{\text{drag\_N}} \tag{6}$$

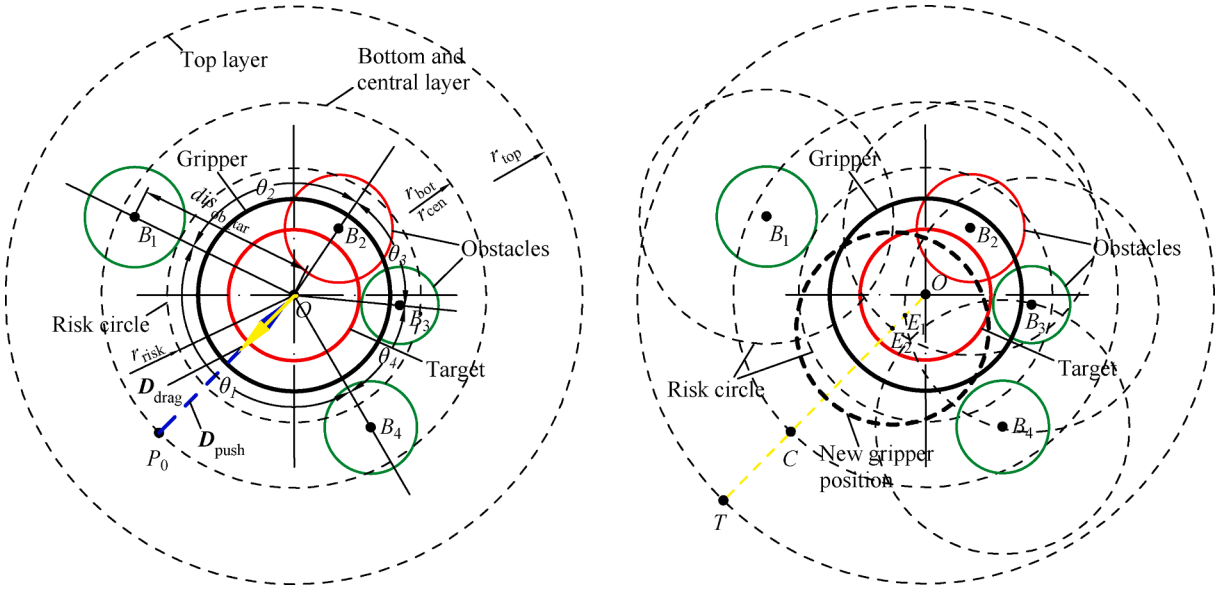

Fig. 9. Diagram of calculating the push and drag vectors: (a) top view with layers and obstacles to show the calculation of push or drag directions; (b) calculation of drag magnitude.

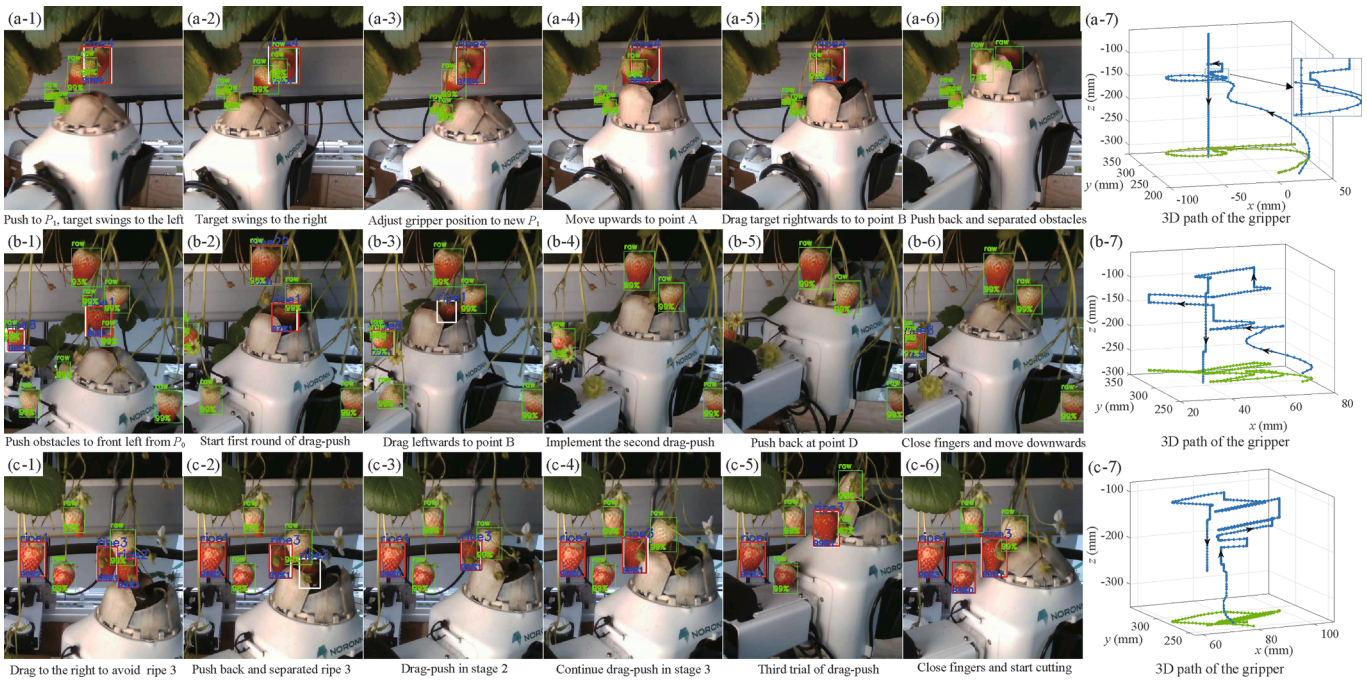

Fig. 10. Examples of the proposed method in the field test: each row of images represents a picking case, where the left six images that were captured by the robot camera show the picking procedures and the last image (right) displays the corresponding 3D trajectory of the gripper (blue line, and the green line is the trajectory projection on the *xy* plane). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. Results and discussions

3.1. Results

3.1.1. Application demonstrations of the proposed method

Fig. 10 demonstrates three picking examples of using the proposed method under different situations. The robot recorded these images during picking, showing both detection and tracking in a dynamic environment. In Fig. 10(a), the target ripe berry 4 and several green berries that grew on the same branch swung from left to right. First, the gripper moved to push aside the obstacles under the target to P_1 where the branch of berries was on the left (Fig. 10(a-1)). After that, the branch of berries swung to the right quickly, but the gripper remained in the left (Fig. 10(a-2)). In a short period, the system used the closed vision loop to adjust the gripper to the newest P_1 position (Fig. 10(a-3)). In the meanwhile, the berries stopped swinging due to the contact with the gripper. In Fig. 10(a-4), the gripper started to use a drag-push to separate the green obstacle in the central layer. The gripper dragged the target rightwards to point B to avoid the left green berry (Fig. 10)) and pushed back to the left at point D for further fruit enclosing (Fig. 10(a-6)). Fig. 10(a-7) shows the 3D trajectory (blue line) of the gripper tip position during the picking, in which the enlarged part shows the position adjustment of the closed-loop controlled gripper in the first stage.

Fig. 10(b) demonstrates a case where the separation method was used in all stages/layers. In stage 1, the gripper moved to P_0 position and pushed aside the bottom obstacles to the front left (Fig. 10(b-1)). Thereafter, a drag-push motion was used to separate the obstacle in the central layer (Fig. 10(b-2) and (b-3)). Then the gripper continued to separate the obstacle in the top layer using another drag-push (Fig. 10(b-4) and (b-5)). After obstacle separation, the gripper closed its fingers and moved downward for fruit detachment. Fig. 10(c) mainly illustrates a case of using continuous drag-push to separate an obstacle that did not fall. The gripper first used a drag-push to separate the obstacle ripe 3 in the left (Fig. 10(c-1) and (c-2)). After that, the system performed another drag-push to avoid the unripe berry in the central layer (Fig. 10(c-3)). However, the unripe berry did not fall and got stuck on the tip of the

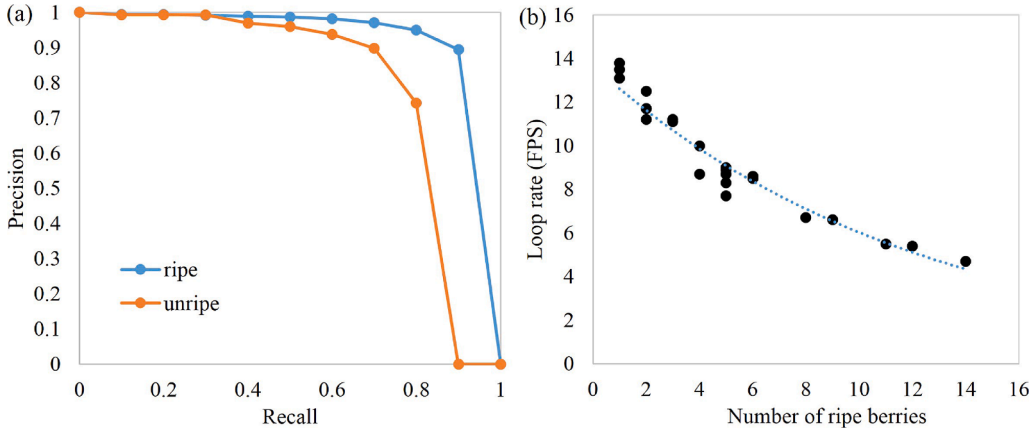

Fig. 11. Performance of the detection method: (a) precision-recall curves of the detection method; (b) loop rate - number of ripe berries of the perception system.

gripper (Fig. 10(c-4)). The system recaptured the environment and performed the second trial of drag-push on the same obstacle, although it moved to the top layer. In the third stage, the primary goal was to enable the target to fall inside the gripper that received sufficient long peduncle for fruit detachment, and in the meanwhile, the gripper did not capture obstacles. However, in this case, the target was pushed up together with the obstacle as they grew on the same stem. The gripper sensors detected insufficient length of peduncle and instructed the gripper to move upwards further. The system then carried out the third trial of the push-drag, and finally, the gripper received sufficient long peduncle and the obstacle fell at the same time (Fig. 10(c-5) and (c-6)).

3.1.2. Performance of the detection method

Fig. 11(a) shows the precision-recall curves of ripe and unripe classes of the YOLOv4 detection method. The intersection over union (IOU) threshold for the evaluation was 0.5. The ripe and unripe classes' average precision (AP) were 91.8% and 81.5%, respectively. It is clear that the detection network performed better on the ripe strawberries. This was because that the unripe strawberries varied in size and shape. As mentioned before, the detection network YOLOv4 ran at a speed of 22 FPS on the robot computer (GTX 1060 6 GB GPU). However, the loop rate of the whole perception system (including detection, tracking, and localization) was reduced to a medium value of around 9 FPS and was dependent on the number of ripe strawberries (Fig. 11(b)), since only ripe strawberries were tracked.

We also performed an experiment to evaluate the positional accuracy of the system in the field. Accurate measuring the ground truth positions of strawberries is difficult. Same to the previous method (Xiong et al., 2019), a 3D printed pointer with a sharp tip was mounted on the gripper for easier measurement. As shown in Fig. 12(a), the distance between the strawberry tip (roughly used as the bottom center point P_1) and the pointer tip was regarded as the absolute positional error, which was measured by a caliper. Fig. 12(b) shows the distributions of the positional errors for both ripe and unripe strawberries. The mean positional errors of reaching the ripe and unripe berries were 7.3 mm and 9.1 mm, respectively. The smaller sizes of unripe berries made the positioning less accurate. Compared to other picking devices, the gripper used in this paper was high tolerated to positional errors (Xiong et al., 2019). The gripper opened fingers forming a closed ring to surround the target strawberry from below, which usually started with the sharp strawberry bottom tip. Once a part of the strawberry entered the gripper, the gripper was likely to fully captured it. The tolerance of the gripper was roughly the half of the aperture diameter of the gripper fingers (35 mm used in this work). However, if the strawberry was highly tilted to the ground, it might be difficult to capture the fruit, as the P_1 position was the bottom center of the bounding box, not the tip position.

Table 1
Performance of the proposed method in field tests and its comparison with the old method (Xiong et al., 2020a).

	Performance indicators	Results of this work	Results of the old method
Success rate	Before separation	88.1%	N/A
	Stage 1	85.3%	81.8%
	Stage 2	80%	69%
	Stage 3	84.6%	75.4%
	Detachment	95.5%	N/A
	Whole process	62.4%	45.6%
Cycle time	Failure cases	7.6 s	6.9 s
	Successful cases	6.8 s	7.6 s

3.1.3. Picking performance tests

The performance experiment was carried out on a strawberry cultivator of "Murano" in a strawberry tunnel at the Norwegian University of Life Sciences, which used the same settings as in the previous work (Xiong et al., 2020a). Since this work focused on obstacle separation, only the targets surrounded by obstacles were counted for the performance evaluation. This may result in a lower success rate because the robot performed well on isolated strawberries (Xiong et al., 2020b). A total of 109 attempts on detected ripe strawberries were recorded for the performance tests. As shown in Table 1, we divided the picking process into five steps: before separation, stage 1, stage 2, stage 3, and detachment. Failures caused by localization error from the vision system before the separation were independently recorded, because it happened before the manipulation started. The result of each separation stage only included the situation that one or more obstacles were located in the corresponding layer, while the whole process might contain an isolated situation in one or two layers/stages, but at least one stage had obstacles. The success in the whole process means that all the five steps were successful. The success rate of the whole process was 62.4% using the proposed method, which increased 36.8% from the previous work. Overall, the system was improved in all separation stages, in which the drag-push using continuous "look-and-move" in stage 2 reduced many failures compared to the old system.

In terms of speed, we recorded the execution time for both failure and successful cases. The cycle time of picking one target included perception, manipulation, and fruit detachment. In general, the speed of the improved system was not reduced even using a more complex vision control system. One reason was using a more precise drag-push instead of using multiple zig-zag pushes in the central layer. Another reason was

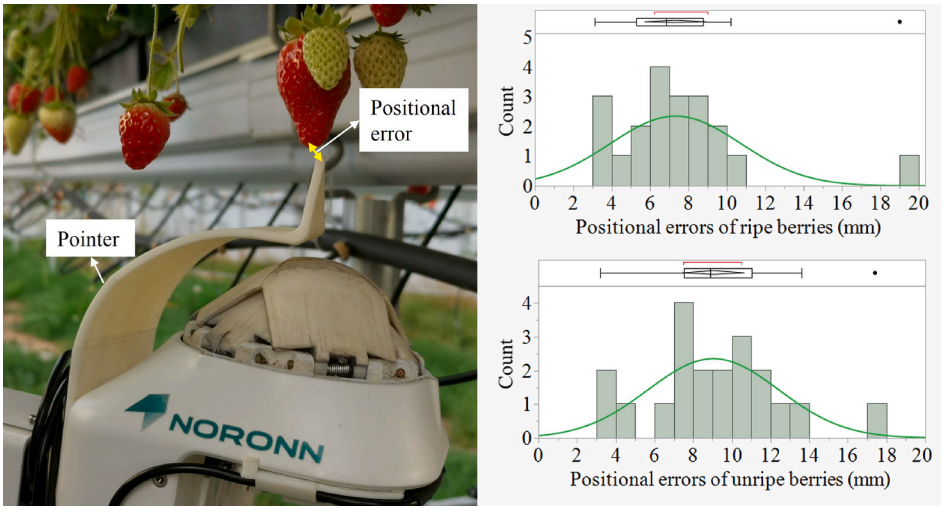

Fig. 12. Positional accuracy test: (a) measurement of positional errors; (b) positional error distributions of ripe and unripe berries.

Table 2
Failure analysis of the proposed method.

Stages and failures	Failure reasons
Before separation:	
a. 76.9%	a. Inaccurate localization due to occlusions.
b. 23.1%	b. Inaccurate localization due to arm-camera calibration.
Stage 1:	1. Difficult: unable to separate some obstacles that were too close to the target or were in dense clusters.
1. 12.5%	2. Pushed up: the target was pushed up but did not fall further into the gripper because: a) obstacles did not fall on outside of the fingers, resulting in the target not
3. 37.5%	falling as they grew on the same branch; b) gripper was covered by branches and not able to swallow the target; c) the target peduncle was too short, and the
4. 12.5%	gripper pushed up the connected branches.
5. 12.5%	3. Latency: failed to capture the target when reaching because berries swung quickly, and the closed-loop control did not work well due to latency.
6. 25%	4. One push: some new obstacles appeared after pushing and were wrongly captured.
Stage 2:	5. Undetected: did not detect small or occluded unripe berries or flowers that were picked together with the target.
1. 16.7%	6. Not updated: the target was moved, but the position was not updated and resulted in failure capturing, because: a) target was initially detected and localized
2. 16.7%	correctly but was not detected or incorrectly localized due to the new occlusions caused by separation; b) target was initially detected as a ripe berry but was
5. 16.7%	changed to unripe when gripper was approaching because of the illumination changes (shadow).
7. 25%	7. Disconnected detection: some obstacles were initially detected but not detected when they became occluded during separation or dropped suddenly that were
8. 25%	unstable to be detected, and thus no drag-push was carried out to separate them from the target.
Stage 3:	8. Inaccurate drag: insufficient or excessive drag distance, resulting in the target slipping out from or obstacles falling into the gripper.
2. 33.3%	
5. 16.7%	
6. 16.7%	
7. 16.7%	
8. 16.7%	
Detachment:	
c. 60%	c. Gripper fingers clamped other peduncles or branches when closing.
d. 40%	d. Failed to cut the target peduncle.

the faster detection system, which could even be ignored compared with the time used for manipulation. A noticeable difference was that the failure cases used more time than the successful cases. In some failure cases, the system used several drag-pushes for continuously detected target but did not fall for detachment.

3.2. Failure cases and discussions

Table 2 reports the failure reasons and the percentages of the failures in each step. Inaccurate localization of a target was the main failure reason before separation, especially in partially occluded situations where the depth sensing might be affected by front obstacles. Also, when the bottom of a target was occluded, the separation in stage 1 was hard to use because the gripper may push the bottom part of the target. Future work will consider completing the occluded bottom part of the strawberries.

A common failure (5) in all three separation stages was the wrongly capturing of undetected berries and flowers. The current single-view vision system was not able to detect the rear obstacles that were fully occluded by the front objects. However, the proposed obstacle separation required full obstacle information around the target to generate accurate motions. The undetected obstacles might be wrongly captured by the gripper. Therefore, future work should utilize the fusion of one or more cameras to provide multiple views, which can also improve the localization accuracy in partially occluded situations. Also, the class of unripe berries in the current vision system was trained on relatively large green berries, but did not include very small unripe berries and flowers. The small unripe berries have different shapes and features than typically-sized unripe berries, so it is necessary to set a new class to train the small unripe berries.

In stage 1, when the berries swung quickly, the system might not capture the target due to the latency of the closed-loop vision control system (failure 3). A faster perception manipulation system with motion prediction capacities may help address this issue. Another common failure in stage 1 was “not updated” (failure 6). Such failure would require the detection and localization system to work in a more complex environment, especially in highly occluded situation. Also, in stage 1, the closed-loop control was only performed once at two points, P_0 and P_1. However, occasionally, some new obstacles moved to the bottom of the target after pushing and were captured by the gripper in stage 2 (failure 4). This was because that the push may break the static contact force that held the berries together. To avoid this failure, the system should recheck the obstacles in the bottom layer and, if necessary, re-perform the push in stage 1 before moving on to stage 2.

In stage 2, the frequent failures were “disconnected detection” (failure 7) and “inaccurate drag” (failure 8). Inaccurate drag was because the drag-push process was controlled using an open vision loop. The predetermined drag distance might be inaccurate when the obstacles moved obviously or were inaccurately localized. Future work should consider using closed-loop vision for all stages, although it may make the system more complicated and slower. “Pushed up” (failure 2) frequently occurred in stage 3 and also can be seen in stage 2. The control system only used the obstacles above the gripper tip for separation in stages 2 and 3 and could not recognize these failures. The future system may need to detect the obstacles below the gripper tip and have the ability to identify and handle any failure cases. Also, an additional vision system inside of the gripper may help identify wrongly captured obstacles and instruct the gripper to get rid of them and recapture the target.

4. Conclusions

This paper presents the improvements in perception, obstacle separation and control method to the active obstacle separation method for strawberry picking in clusters. A faster and more accurate vision system was developed that combined YOLOv4, Deep SORT and color thresholding for strawberry detection, tracking and localization in real-time. We proposed an improved version of the active obstacle separation method that used a push operation in the bottom layer and a drag-push in the central and top layers. The push and drag vectors were simplified and precisely calculated based on the locations of obstacles. Most importantly, different from the old system that only “looked” once for the entire obstacle separation stages, the new system used a hybrid vision-based control system for the separation process. In stage 1, a simple closed-loop vision was used to control the push operation at two key points. In stages 2 and 3, the vision system re-perceived the

environment to update the information for each round of drag-push movements. Field evaluation showed that the improved picking method was more precise to separate the obstacles, increasing the whole process success rate to 62.4% in clusters on the "Murano" strawberry cultivator that was 36.8% higher than the previous method. Meanwhile, the picking did not slow down on successful cases despite using a more complex vision-guided system. Future work may include full-process closed visual loop, failure case handling, and multi-view perception.

CRediT authorship contribution statement

Ya Xiong: Conceptualization, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Yuanyue Ge:** Data curation, Writing – review & editing, Visualization. **Pål Johan From:** Resources, Funding acquisition.

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.

Acknowledgments

This work was supported by the Research Council of Norway [project title: SHAPE - strawberry harvester for polytunnels and open fields, Grant No. 303607].

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.compag.2021.106508.

References

Anjom, F.K., Vougioukas, S.G., Slaughter, D.C., 2018. Development of a linear mixed model to predict the picking time in strawberry harvesting processes. Biosyst. Eng. 166, 76–89. https://doi.org/10.1016/j.biosystemseng.2017.10.006.

Arad, B., Balendonck, J., Barth, R., Ben-Shahar, O., Edan, Y., Hellström, T., Hemming, J., Kurtser, P., Ringdahl, O., Tielen, T., van Tuijl, B., 2020. Development of a sweet pepper harvesting robot. J. Field Robot. 37, 1027–1039. https://doi.org/10.1002/rob.21937.

Bac, C., Hemming, J., Van Henten, E., 2013. Robust pixel-based classification of obstacles for robotic harvesting of sweet-pepper. Comput. Electron. Agric. 96, 148–162. https://doi.org/10.1016/j.compag.2013.05.004.

Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B., 2016. Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). https://doi.org/10.1109/icip.2016.7533003.

Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M., 2020. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 https://arxiv.org/abs/2004.10934.

Ge, Y., Xiong, Y., Tenorio, G.L., From, P.J., 2019. Fruit localization and environment perception for strawberry harvesting robots. IEEE Access 7, 147642–147652. https://doi.org/10.1109/ACCESS.2019.2946369.

Kootstra, G., Wang, X., Blok, P.M., Hemming, J., Van Henten, E., 2021. Selective harvesting robotics: current research, trends, and future directions. Curr. Robot. Rep. 2, 95–104. https://doi.org/10.1007/s43154-020-00034-1.

Lehnert, C., McCool, C., Sa, I., Perez, T., 2020. Performance improvements of a sweet pepper harvesting robot in protected cropping environments. J. Field Robot. 37, 1197–1223. https://doi.org/10.1002/rob.21973.

Lehnert, C., Tsai, D., Eriksson, A., McCool, C., 2019. 3d move to see: Multi-perspective visual servoing towards the next best view within unstructured and occluded environments. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3890–3897. https://doi.org/10.1109/IROS40897.2019.8967918.

Lin, G., Zhu, L., Li, J., Zou, X., Tang, Y., 2021. Collision-free path planning for a guava-harvesting robot based on recurrent deep reinforcement learning. Comput. Electron. Agric. 188, 106350. https://doi.org/10.1016/j.compag.2021.106350.

Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L., 2014. Microsoft coco: Common objects in context. In: European conference on computer vision. Springer, pp. 740–755.

Mghames, S., Hanheide, M., Ghalamzan, A., 2020. Interactive movement primitives: Planning to push occluding pieces for fruit picking. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp. 2616–2623. https://doi.org/10.1109/IROS45743.2020.9341728.

Silwal, A., Davidson, J.R., Karkee, M., Mo, C., Zhang, Q., Lewis, K., 2017. Design, integration, and field evaluation of a robotic apple harvester. J. Field Robot. 34, 1140–1159. https://doi.org/10.1002/rob.21715.

Tang, Y., Chen, M., Wang, C., Luo, L., Li, J., Lian, G., Zou, X., 2020. Recognition and localization methods for vision-based fruit picking robots: A review. Front. Plant Sci. 11, 510. https://doi.org/10.3389/fpls.2020.00510.

Wada, K., 2016. labelme: Image Polygonal Annotation with Python. URL: https://github.com/wkentaro/labelme.

Wagner, N., Kirk, R., Hanheide, M., Cielniak, G., et al., 2021. Efficient and robust orientation estimation of strawberries for fruit picking applications. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE. URL: https://eprints.lincoln.ac.uk/id/eprint/44426/.

Wojke, N., Bewley, A., 2018. Deep cosine metric learning for person re-identification. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE. pp. 748–756. https://doi.org/10.1109/WACV.2018.00087.

Wojke, N., Bewley, A., Paulus, D., 2017. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3645–3649. doi: https://doi.org/10.1109/ICIP.2017.8296962.

Xiong, Y., Ge, Y., From, P.J., 2020a. An obstacle separation method for robotic picking of fruits in clusters. Comput. Electron. Agric. 175, 105397. https://doi.org/10.1016/j.compag.2020.105397.

Xiong, Y., Ge, Y., Grimstad, L., From, P.J., 2020b. An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation. J. Field Robot. 37, 202–224. https://doi.org/10.1002/rob.21889.

Yaguchi, H., Nagahama, K., Hasegawa, T., Inaba, M., 2016. Development of an autonomous tomato harvesting robot with rotational plucking gripper. In: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE. pp. 652–657. doi: https://doi.org/10.1109/IROS.2016.7759122.

Xiong, Y., Peng, C., Grimstad, L., From, P., Isler, V., 2019. Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper. Comput. Electron. Agric. 157, 392–402. https://doi.org/10.1016/j.compag.2019.01.009.

Yamamoto, S., Hayashi, S., Yoshida, H., Kobayashi, K., 2014. Development of a stationary robotic strawberry harvester with a picking mechanism that approaches the target fruit from below. Jpn. Agric. Res. Quart. JARQ 48, 261–269. https://doi.org/10.6090/jarq.48.261.

Yu, Y., Zhang, K., Liu, H., Yang, L., Zhang, D., 2020. Real-time visual localization of the picking points for a ridge-planting strawberry harvesting robot. IEEE Access 8, 116556–116568. https://doi.org/10.1109/ACCESS.2020.3003034.