Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

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

Original papers

Accurate detection and depth estimation of table grapes and peduncles for robot harvesting, combining monocular depth estimation and CNN methods

Gabriel Coll-Ribes [a,1], Iván J. Torres-Rodríguez [a,1], Antoni Grau [b], Edmundo Guerra [b], Alberto Sanfeliu [a,b,*]

[a] Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Llorens i Artiga 4-6, Barcelona, 08028, Spain
[b] Universitat Politècnica de Catalunya, Pau Gargallo 5, Barcelona, 08028, Spain

ARTICLE INFO

Keywords:
Image segmentation
Monocular depth
Grape bunch and peduncle detection
Grape bunch and peduncle depth estimation
Robot harvesting

ABSTRACT

Precision agriculture is a growing field in the agricultural industry and it holds great potential in fruit and vegetable harvesting. In this work, we present a robust accurate method for the detection and localization of the peduncle of table grapes, with direct implementation in automatic grape harvesting with robots. The bunch and peduncle detection methods presented in this work rely on a combination of instance segmentation and monocular depth estimation using Convolutional Neural Networks (CNN). Regarding depth estimation, we propose a combination of different depth techniques that allow precise localization of the peduncle using traditional stereo cameras, even with the particular complexity of grape peduncles. The methods proposed in this work have been tested on the WGISD (Embrapa Wine Grape Instance Segmentation) dataset, improving the results of state-of-the-art techniques. Furthermore, within the context of the EU project CANOPIES, the methods have also been tested on a dataset of 1,326 RGB-D images of table grapes, recorded at the Corsira Agricultural Cooperative Society (Aprilia, Italy), using a Realsense D435i camera located at the arm of a CANOPIES two-manipulator robot developed in the project. The detection results on the WGISD dataset show that the use of RGB-D information ($mAP = 0.949$) leads to superior performance compared to the use of RGB data alone ($mAP = 0.891$). This trend is also evident in the CANOPIES Grape Bunch and Peduncle dataset, where the mAP for RGB-D images ($mAP = 0.767$) outperforms that of RGB data ($mAP = 0.725$). Regarding depth estimation, our method achieves a mean squared error of 2.66 cm within a distance of 1 m in the CANOPIES dataset.

1. Introduction

The agri-food sector is central to human society and faces several challenges. One challenge is how to feed the world's growing population without increasing the amount of land devoted to agriculture, which could lead to deforestation, additional Greenhouse Gas (GHG) emissions and reduced biodiversity. Another challenge is the shortage of agricultural workers, mainly due to the seasonal nature of the work and the migration of labor to more stable productive sectors, driven by better employment opportunities, working conditions, access to education, social protection, credit and markets (FAO, 2016). The latter challenge can be overcome by the full or partial automation of the various agri-food processes.

Precision agriculture or precision farming is a modern, whole-farm management concept that uses several technologies, from remote sensing and proximal data collection to automation and robotics. In our case, we are dealing with the harvesting of table grapes, which has to be done by a robot using perception systems and a robotic arm. This task has proven to be very challenging when it comes to the perception system, the mechanical design of the robot, the control methods, the navigation and the manipulation systems. In this article, we will only analyze the perception system to detect the table grapes and their peduncles, as well as the 3D location of the peduncles to be grasped.

The complexity of this perceptual task is reflected in the nature of the environment in which it is performed. The field is unstructured (e.g., the table grapes can be of different sizes and appear at different heights and locations), and the conditions (e.g., illumination or partial occlusion) are constantly changing. This poses several problems. For example, the bunch may be partially occluded by leaves or other bunches. Alternatively, the cluster may be visible, but the location on the image of the peduncle to be cut may not be obvious (more

* Corresponding author.
E-mail address: alberto.sanfeliu@upc.edu (A. Sanfeliu).
[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.compag.2023.108362
Received 1 April 2023; Received in revised form 12 September 2023; Accepted 23 October 2023

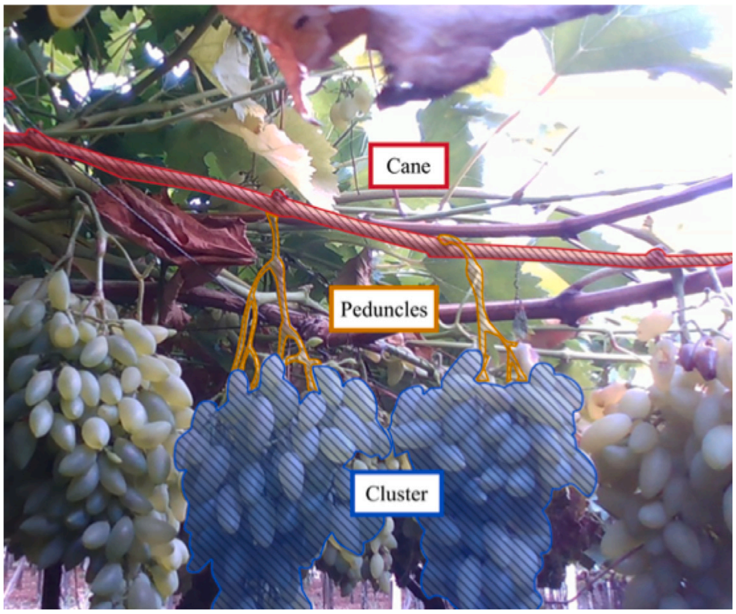

Fig. 1. Grapevine parts in the section of the plant closest to the grapes.

than one possible peduncle) or impossible (peduncle not visible due to occlusion). In any case, the result of the perception system must be the detection in the image of the bunches to be harvested, as well as the 3D localization of the corresponding peduncles. This information is then sent to a robotic arm equipped with a tool capable of grasping and cutting the peduncle.

Robotic harvesting is an emerging technology that has been tested for harvesting sweet peppers (Sa et al., 2017), strawberries (Han et al., 2012), and apples (De-An et al., 2011). Harvesting involves not only detecting the fruit, but also identifying the stalk or peduncle that connects the fruit to the tree branch or cane. The characteristics of the stalk or peduncle vary depending on the type of fruit. For sweet peppers and apples, they are usually wide and long. In the case of grapes, however, the peduncles are very thin (a few millimeters), branched, and sometimes short (see Fig. 1). Furthermore, the bunches we are dealing with are located in canopies, resulting in a very cluttered background (Fig. 1). This makes it very difficult to separate the grapes and peduncles from the background. In addition, the lighting conditions under the canopies are highly variable, changing with the density of the foliage, and can include cast shadows and variations in luminance. These factors contribute to the complexity of robotic harvesting, as it involves not only the difficulty of detecting the stems, but also navigating around obstacles in the environment. In this work, we present a set of methods that address these challenges, and our results, obtained from real images captured in the field, demonstrate that our methods outperform existing technologies. Moreover, it proves good generalization as it has been tested with plastic and real grapes of different type.

The motivation of this article is to provide a functional and field-tested (Figs. 2 and 3) solution for the perception needed to recognize bunches and peduncles by providing the cut-off point in 3D camera coordinates. This would allow to generalize the solution for other projects and robots.

In this paper, we first propose a precise method for the detection of table grapes and their corresponding peduncles, which takes as input an RGB-D image whose depth channel has been obtained by monocular depth estimation. Using this image as input, detection and segmentation are performed using a state-of-the-art method based on convolutional neural networks.

We also propose an exact estimation of the distance of each peduncle from the camera to obtain a precise 3D location of the cutting point.

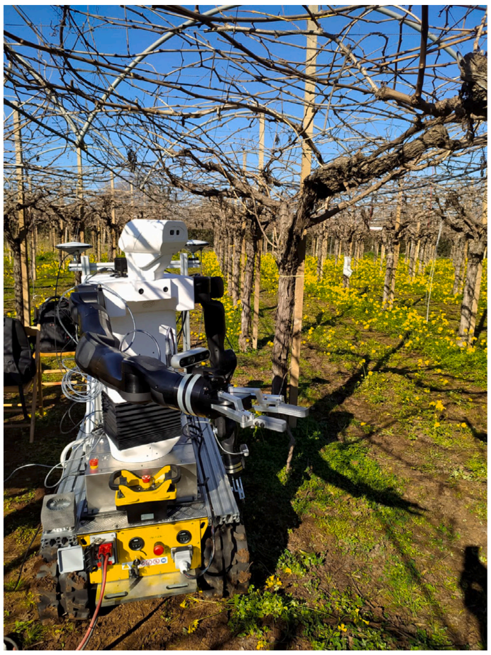

Fig. 2. CANOPIES robot in the field.

Due to the small dimensions of a grape peduncle (1–5 mm), traditional depth sensors (LIDAR's, stereo cameras) have proven incapable of providing robust depth measurements. We present a fusion of two methods to obtain an accurate estimate of the distance of the peduncle from the camera.

The article is structured in the following sections. Section 2 presents the related work on instance segmentation, monocular depth estimation, and computer vision techniques applied to harvesting of fruits and vegetables. Section 3 presents the system overview, the databases used for validation of the methods, the metrics, and the models developed for instance segmentation and depth estimation. Section 4 presents the experimental setup, the instance segmentation and depth estimation results. Finally, a discussion of the results and conclusions of the work are presented in Sections 5 and 6, respectively.

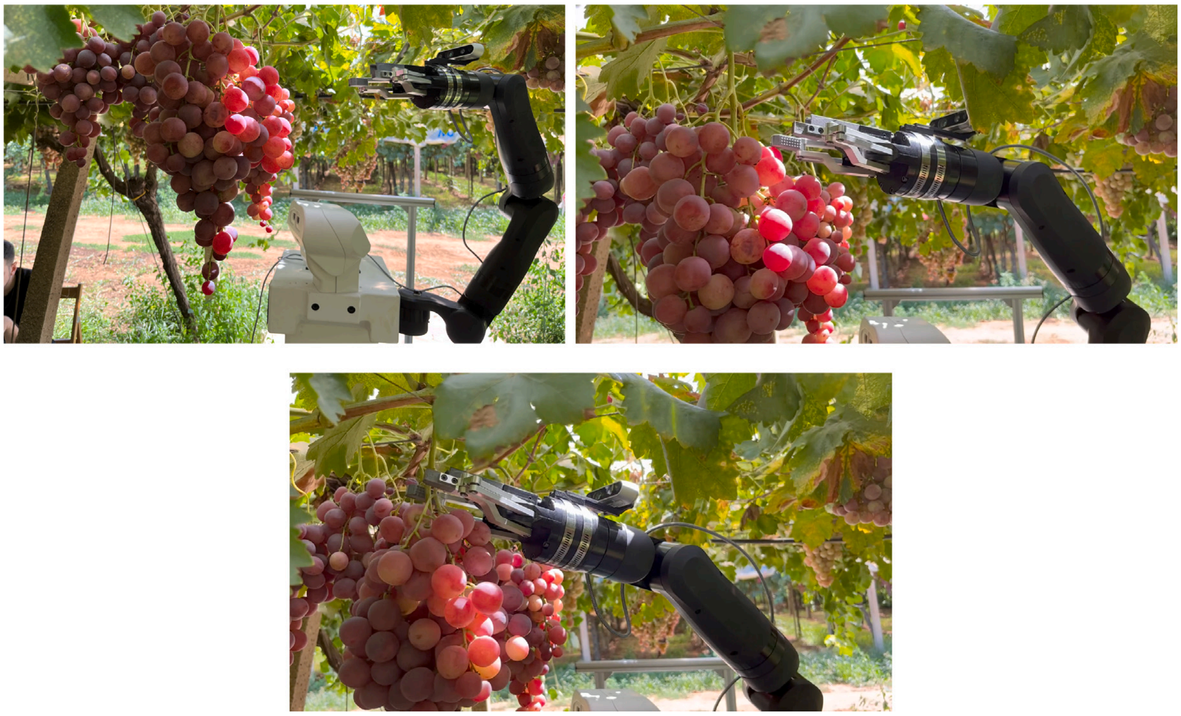

Fig. 3. CANOPIES robot in the field. Top-left: Robot arm with camera in the wrist; Top-right: Cutting tool griper with stereo camera approaching a grape; Bottom: Cutting tool griper touching the peduncle.

2. Related work

2.1. Instance segmentation

Instance segmentation is a computer vision technique that involves two main steps: object detection and semantic segmentation. In other words, an instance segmentation method must be able to detect individual instances of different classes of objects in an image, while also being able to produce binary masks for each of these detection's.

Overall, there is an extensive literature on image segmentation methods that do not rely on neural networks. Graph-based segmentation techniques are used for this task, such as the GrabCut method (Rother et al., 2004), the Random Walker method (Grady, 2006), and the Normalized Cut method (Shi and Malik, 2000). These methods use graph-based representations of images to segment objects. In addition, several traditional methods have also been applied to this task, such as thresholding, histogram-based clustering, region growing, watershed or k-means clustering. However, given the complexity of the instance segmentation task in unstructured and changing condition environments, most of the successful approaches are based on deep learning.

Different Deep Learning techniques have been developed in object detection using Convolutional Neural Networks (CNN), but a milestone on these techniques was the work on Region-based CNN (Girshick et al., 2014) in 2014. After this work, in 2015, it appeared Fast R-CNN (Girshick, 2015) and Faster R-CNN (Ren et al., 2015). However a great improvement was the work Mask R-CNN (He et al., 2017), which extended the Faster R-CNN method (Ren et al., 2015) with an object mask prediction step after the object detection branch. The validation of this technique in the COCO (Common Objects in Context) dataset (Lin et al., 2014), showed great results for instance segmentation. If we use the mean average precision (mAP) as a metric (refer to Section 3 for more details) the obtained results range from 33.1 to 37.1, depending on the backbone used. An extention of this technique was Mask scoring R-CNN (Huang et al., 2019) that was built upon Mask R-CNN, by adding another head containing the MaskIoU prediction network. This improved the performance, as it obtained an average precision (AP) in the COCO dataset of 35.4 while Mask R-CNN obtained 34.3. In addition, it improved the robustness of its output against different backbones and frameworks.

YOLACT (Bolya et al., 2019) is another CNN based approach for instance segmentation. It reports a lower mAP than Mask R-CNN (a mAP of 29.8 against 35.7), but its main advantage is that it runs in real-time, 30 frames per second (FPS), as opposed to Mask R-CNN, 5 FPS. One more Deep Learning approach was DetectoRS (Qiao et al., 2021). It is based on Recursive Feature Pyramids, a modification of Feature Pyramid Networks. This approach outperformed Mask R-CNN, obtaining an mAP of 44.4–48.5 (depending on the backbone) in the COCO test dataset. SCNet (Vu et al., 2021) also surpassed Mask R-CNN, obtaining a mAP of 40.2–42.7 (depending on the backbone) in the COCO test-dev dataset. QueryInst (Fang et al., 2021) introduced an architecture that implements query-based instance segmentation, also upgrading Mask R-CNN results (mAP of 39.9–49.1 in the COCO test dataset, depending on the backbone) at a similar speed (ranging between 3.3 FPS and 13.5 FPS, depending on the backbone).

2.2. Monocular depth estimation

Depth estimation is a computer vision task designed to estimate depth from a 2D image. First approaches to monocular depth estimation were based on MRF-based formulations (Saxena et al., 2009), followed by methods that used simple geometric assumptions or non-parametric methods. Significant advances were done by using the expressive power of convolutional neural networks to directly regress scene depth from the input image (Eigen et al., 2014). To enhance prediction accuracy, new architectural innovations have been proposed (Roy and Todorovic, 2016), which uses depth data from real sensors (like RGB-D cameras or LIDAR) for training. Recently, a powerful architecture denominated MiDaS (Ranftl et al., 2022) based on neural networks was trained by mixing several datasets for zero-shot cross-dataset transfer to obtain a robust monocular depth estimation. This model was enhanced by replacing the convolutional neural network by a vision-transformer model as a backbone for dense prediction tasks (Ranftl et al., 2021).

Another interesting network is the one presented in Miangoleh et al. (2021). This method is based on MiDaS, but its main contribution is to obtain relative depth maps with a higher resolution than MiDaS. The main idea of the work emerges from the observed changes in the MiDaS output, when the resolution of the input image changes. What is observed is that, at low resolution, details are lost in the depth map, but the structural consistency is good. However, when the resolution is increased, the relative depth map is more detailed, but the structural consistency deteriorates. From this information, the method tries to find the optimal relationship between depth map detail and structural consistency. This solution seems interesting because obtaining detailed depth maps could be a great advantage for segmentation and can help in the image processing stage. However, the results in terms of depth are more inconsistent than those from MiDaS. In the supplementary material (Hosseini Minagoleh et al., 2021) of the paper, in section A, the authors wrote “We notice a discrepancy between the apparent visual improvement gained from depth refinement techniques and numerical results based on common metrics”. This is clearly appreciated in some comparisons presented in the supplementary material. For instance, in the Ibims1 dataset (Koch et al., 2019) MiDaS obtains an absolute relative error of 2.0325 while Miangoleh et al. (2021) obtains 2.0510.

2.3. Agriculture works

Fruit and vegetable detection as well as their depth estimation using computer vision techniques is a growing field in the agricultural industry. In Yin et al. (2021), they present a method for grape detection and depth estimation method which is based on Mask R-CNN (He et al., 2017) and, once the grape is detected, a 3D point cloud is extracted and filtered. A Random Sample Consensus (RANSAC) technique is then applied to fit the point cloud to a cylinder and finally, the depth of the stalk is obtained.

Working with different fruits could lead to similar solutions for the detection and depth estimation process. However, it is important to take into account the particularities of the analyzed fruit. For instance, in Bac et al. (2017), their work focused on sweet peppers. In Han et al. (2012), the fruit used to perform the study was the strawberry. Whereas others such as De-An et al. (2011) worked with apples.

When proposing a solution for fruit and vegetable detection in a real-world agricultural environment, it is critical to consider and address the various challenges that arise. Some of these difficulties are related to the lighting, overlapping, or the different colors of the fruit. In fact, papers like Yu et al. (2019) study the non-structured environment in which the system needs to operate.

As stated before, CNNs are commonly preferred for solving the fruit and vegetable detection problem due to their effective feature extraction capabilities and generalization in unstructured environments. For example, in Wan and Goudos (2019), the architecture of Faster R-CNN (Girshick, 2015) is used for detecting different fruits: apples, oranges, and mangoes. In this case, the convolutional and pooling layers of the base architecture were modified to achieve better fruit detection. Another example is presented in Mai et al. (2020), where another version of Faster R-CNN was developed by merging multiple classifier fusion strategies. Several approaches were built around the YOLO (Bochkovskiy et al., 2020), such as Guo et al. (2023), Sozzi et al. (2022), Zhou et al. (2023), Pinheiro et al. (2023) and Xu et al. (2023), enabling robust, accurate and real-time detection of grape bunches in the vineyard, improving the results obtained with plain YOLO.

A problem with the methods described above is that they only perform object detection (bounding boxes), which may not provide the necessary information for many applications. Other works are able to provide individual masks for each of the detection’s. For example, in Majeed et al. (2018), a convolutional neural network called SegNet (Badrinarayanan et al., 2017) is used to segment branches and apple tree trunks from RGB-D images. Mask R-CNN is also used successfully in works such as Ghiani et al. (2021).

With regard to agriculture-related works that deal with 3D information, there are some focused on tracking and three-dimensional association such as Santos et al. (2020). However, there are others such as Sa et al. (2017), that work directly with the 3D information. In this case, they present methods for detecting sweet peppers. Using 3D information, they are able to cut out the stalk of the pepper. A key point to note is that the peduncles of peppers are wider than those of grapes. This makes it easier to obtain the location of the stalk directly from the RGB-D camera information.

Regarding the implementation of robotics for grape harvesting, recent research includes the dual-arm grape-harvesting robot introduced by Jiang et al. (2022), which is designed for horizontal trellis cultivation. A novel grape harvesting technique is presented in Xu et al. (2023), which uses vision and robotics to identify grapes, predict picking locations, and operate a cut-and-pick robot. In Vrochidou et al. (2021), the authors present an integrated system architecture of an autonomous robot designed for grape harvesting, green harvesting, and defoliation.

As mentioned in Section 1, our work focuses on developing a robot capable of detecting the peduncles of table grape bunches. These peduncles are characterized by being very thin (a few millimeters), branched, and short. In addition, the robot operates in a cluttered background with varying lighting conditions. Previous research in peduncle detection has focused primarily on crops such as peppers, strawberries, and apples. Although there are existing methods for grape bunch image segmentation, they have not specifically addressed peduncle detection and localization. In our work, we propose methods that not only detect grape bunches in complex environments with changing illumination, but also accurately localize the delicate peduncles and their branches. Our image segmentation approach uses both RGB and monocular depth images to separate foreground grapes and peduncles from the background in cluttered scenes. In particular, the monocular depth image allows relative depth estimation using only a single camera, eliminating the need for depth sensors during the segmentation stage. Finally, we propose a technique for estimating peduncle localization in cases where direct peduncle detection is difficult.

3. Materials and methods

3.1. System overview

A flowchart showing the general operation of the method presented in this paper is shown in Fig. 4. The key components of the method include monocular depth estimation, instance segmentation and computing grape and peduncle depths. The input is an RGB image and a point cloud of the scene. This is provided by a stereo camera located in the CANOPIES robot, this robot has been developed by PAL Robotics in the scope of the CANOPIES EU project (CANOPIES, 2021). An estimate of the depth map of the scene is then generated from the RGB image, using the Boosting Monocular Depth Estimation method presented in Miangoleh et al. (2021).

The RGB image and the estimated depth map are combined and used as an input to the instance segmentation method, in order to obtain the segmented individual masks of peduncles and bunches. Then, using the segmentation masks and the point cloud obtained from the camera, the depth (z) to the peduncle is computed. This depth information is used to derive the (x, y, z) coordinates of the picking point as it is shown in Section 3.7. The computation of the picking point coordinates involves the following steps: first, the z coordinate is obtained from the depth estimation, and second, the (x, y) coordinates are derived by using the intrinsic camera values in conjunction with the depth value. As discussed below, this depth estimation process can be performed using two techniques, one when the peduncle is not visible and the other when the peduncle is detected.

The system has been implemented in Python and in ROS (Robot Operating System) Melodic that works in Ubuntu 18.04. This facilitates the integration in other robotic platforms, since ROS is one of the most used frameworks for robotics.

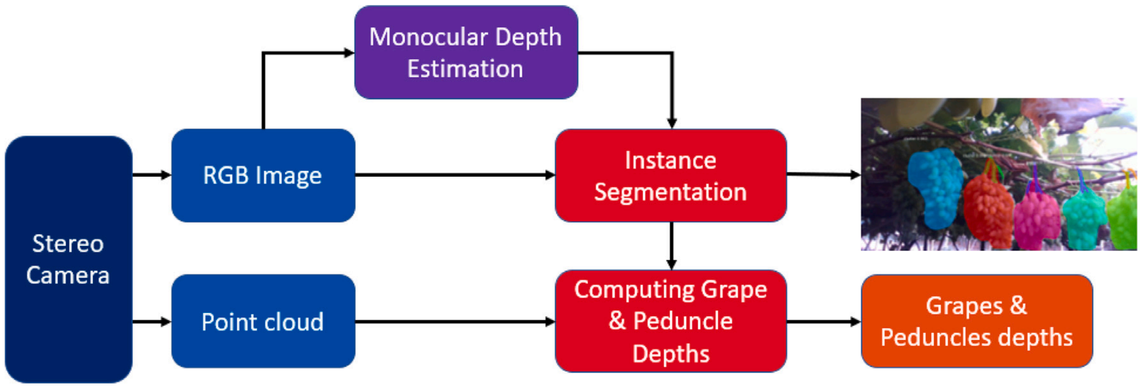
Monocular Depth Estimation
Stereo Camera
RGB Image
Instance Segmentation
Point cloud
Computing Grape & Peduncle Depths
Grapes & Peduncles depths

Fig. 4. General process flow of the method.

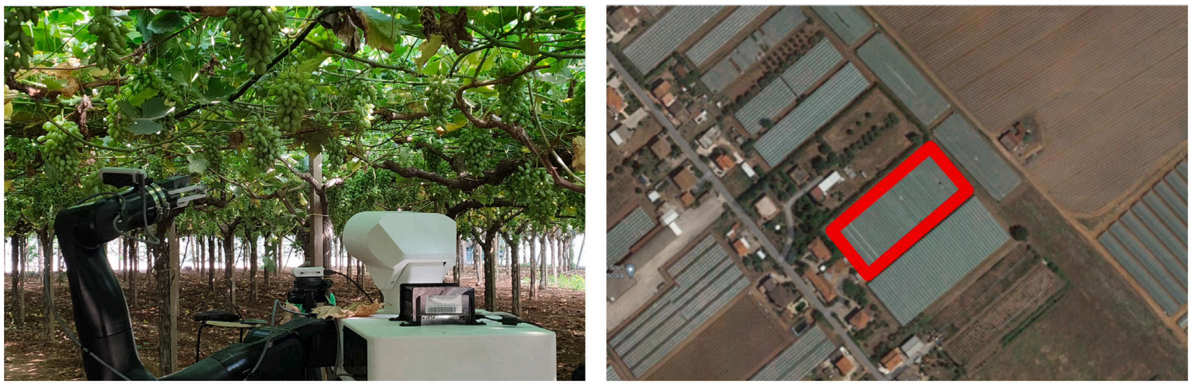

Fig. 5. On the left there is an image of the Corsira field and on the right there is a diagram showing the field location.

3.2. Data collection

In September 2021, data of grape bunches and peduncles were collected in the vineyard of the Corsira Agricultural Cooperative Society (Aprilia, Italy), where table grapes are grown. Fig. 5 shows an image of the field and its location. The left image shows a view of the vineyard field, while the right image highlights (red rectangle) the field used for data collection and experiments. A total of 45 sequences were recorded, with a total duration of 3660 s (61 min) and containing a total of 103,775 images and 80,489 point clouds. The images were acquired on different days and at different times, from 9 am to 5 pm. They had different lighting conditions due to weather conditions, density of foliage, or time of day. The images were also taken at different distances from the bunches, ranging from 30 cm to 2 m.

3.3. CANOPIES Grape Bunch and Peduncle Dataset (GBPD)

From the data recorded, ten sequences were selected in which the peduncles and bunches were fully or partially visible, discarding those taken from too far away to identify the peduncle. The CANOPIES dataset contains a total of 1326 RGB images and provides a total of 3770 binary masks for grape bunches and another 3770 for each of their peduncles. Some examples of the dataset with their corresponding instances segmentation are shown in Fig. 6. The dataset will be available for research purposes in the near future, but can be available on request.

The dataset was labeled using VGG Image Annotator (VIA) (Dutta and Zisserman, 2019; Dutta et al., 2016), a free software that can be used without any installation. The labeling work was divided among 8 people. Each of them was given strict guidelines on how to label the images in order to keep the labeling consistent.

Because the bunch and peduncle had complex geometry, it was indicated that they would be labeled using the polygonal tool. It was also specified that a bunch would only be labeled if its peduncle was visible in the image and could be labeled with the tool without zooming into the image. These criteria filtered out all bunches that were too far away from the camera or whose peduncle was too blurred.

Two complex cases occurred during the labeling process. In the case of overlapping bunches, the line separating the masks may not be clear. The labelers were instructed to separate the masks with a vertical line between the peduncles to maintain coherence. The other complex case was that of branching stems. If the ramification point appeared in the image, it should be considered as a single peduncle, as well as its bunch. On the other hand, if the ramification point appeared outside the image, it should be considered as two separate peduncles, and the bunches should be separated accordingly.

Fig. 6 shows ground truth examples from the dataset. As can be seen, the binary bunch mask contains not only the grapes, but also the peduncle. This was a deliberate decision during the creation of the dataset. The main reason for this choice is to increase the probability of detecting the peduncle. If a single peduncle was not detected, it would be possible to extrapolate it from the other mask.

3.4. WGISD dataset

For further analysis, we will compare the methods proposed here with another dataset from the literature, in addition to our own dataset. The Embrapa Wine Grape Instance Segmentation Dataset (Embrapa WGISD for short) (Santo et al., 2019) is a dataset containing RGB images of grape bunches in vineyards, providing bounding box information for all instances. Additionally, for a subset of the images, the mask of each bunch is also provided.

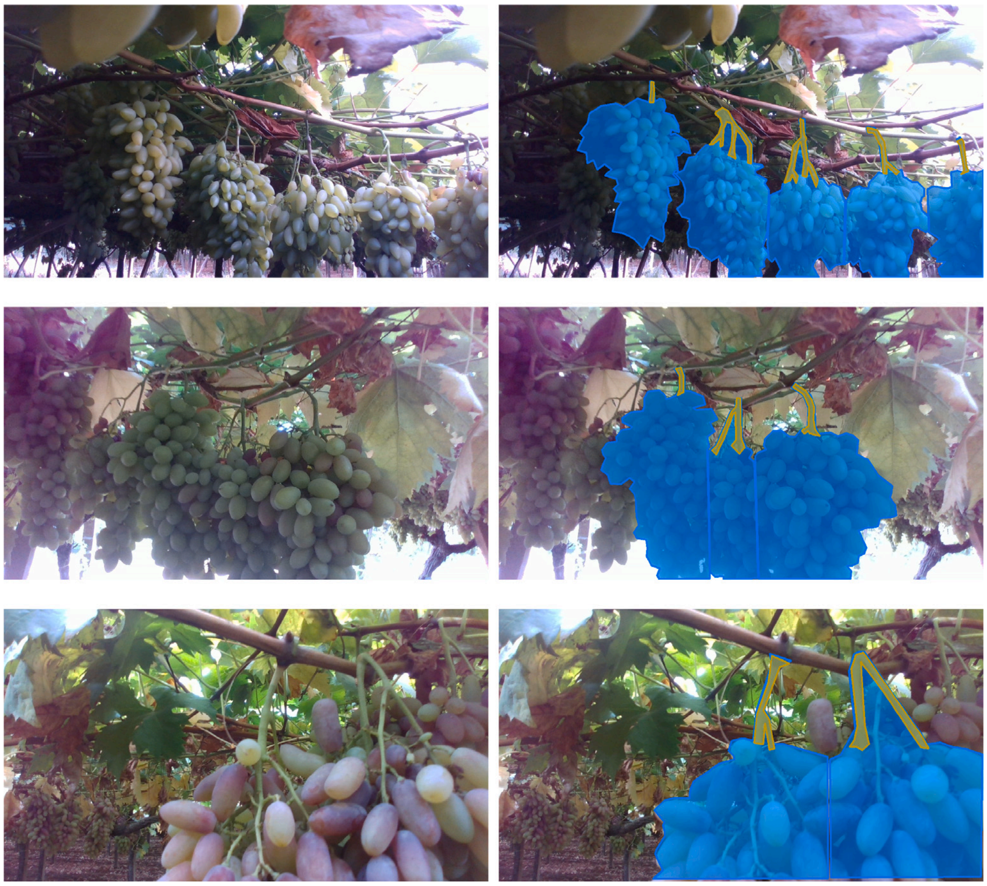

Fig. 6. Examples of ground truth images from the generated dataset. The left column shows the original RGB images. The right column shows the segmentation of the bunches (in blue) and peduncles (in orange) found in the dataset.

Unlike the CANOPIES GBPD dataset, the WGISD dataset does not contain masks or boxes for the peduncles. In addition, only a small subset of the dataset contains bunches masks. Moreover, the detection conditions are simpler than in our dataset, since the vine has been previously prepared and the bunch is in a less chaotic and element-loaded context than in our dataset. An example of an image from this dataset is shown in Fig. 7. In general, the metrics obtained in this dataset are expected to be higher than those obtained in the CANOPIES dataset.

3.5. Metrics

Segmentation results are evaluated using the following state-of-the-art metrics: precision (P), recall (R), F1 score and mean Average Precision (mAP). A description of the first three can be seen in Eq. (1). The mAP is the average of the Average Precision (AP) obtained in each class. The AP is obtained by calculating the area under the Precision-Recall curve. The formulation of mAP can be seen in Eq. (2).

As it can be seen in their respective formulations, these metrics are calculated using the numbers of true positives, false positives, and false negatives. In order to compute them, a matching between the predicted values and the ground truth elements must be established. This matching is affected by two pre-selected values. The first is the confidence score threshold, which is used to discard any detection generated by the network with a confidence score below it. For this implementation, a value of 0.8 has been used. The other value is the minimum IoU value that a detection must have with a ground truth element, in order to be considered a match. Like it is common in the literature, the metrics obtained by a model are presented for different IoU threshold values.

$$P = \frac{N_{TP}}{N_{TP} + N_{FP}} \qquad R = \frac{N_{TP}}{N_{TP} + N_{FN}} \qquad F1 = 2\frac{PR}{P+R} \tag{1}$$

$$AP_k = \frac{1}{11} \sum_{r\in\{0,0.1,\ldots,0.9,1\}} P(r)$$
$$mAP = \frac{1}{K}\sum_{k=1}^{K} AP_k \tag{2}$$

K: number of classes

To evaluate the peduncle depth estimation process, the method is tested on a dataset of known examples. For each peduncle, the depth estimation error is computed. The root mean square error (RMSE), the mean absolute error (MAE), and the standard deviation of the error are used for evaluation. Eq. (3) shows how the RMSE and the MAE are calculated, where z_i is the distance estimation for example i and $z_{\mathrm{gt},i}$ is the ground truth distance of example i. Furthermore, these metrics allow to check if the method is within the tolerance required for its implementation in the field. It is known that the gripper of the robot has a tolerance of ± 3 cm, so the error must be below this threshold.

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_i - z_{\mathrm{gt},i})^2}$$
$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|z_i - z_{\mathrm{gt},i}| \tag{3}$$

Additionally, although the speed of the algorithm will not be taken into account in this first analysis of the method, it is a relevant metric

Fig. 7. Sample from the WGISD dataset.

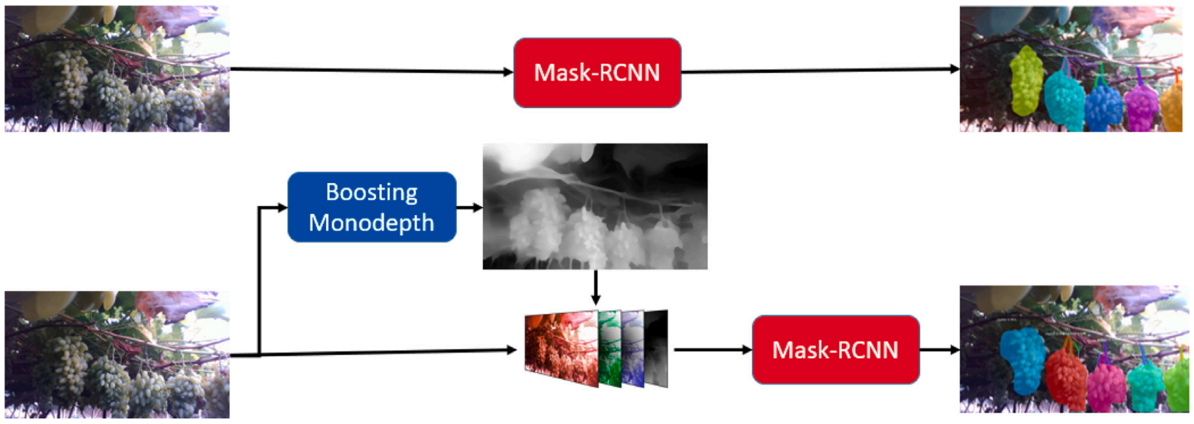

Fig. 8. Diagram of the segmentation process with RGB images (top) or estimated RGB-D images (bottom).

to consider for future implementations. The speed of the algorithm will be evaluated in terms of frames per second (FPS), in other words, how many image frames can be processed per second.

3.6. *Instance segmentation of table grapes and peduncles using CNN*

Object detection requires to locate a region where the object is detected and also compute an identification score. The region is obtained by using a set of bounding boxes. Moreover, when a single pixel mask is obtained for each bounding box detection, instance segmentation is performed. This is therefore a combined problem of object detection and semantic segmentation.

The CANOPIES dataset provides a set of RGB images and individual masks that can be used to train an instance segmentation model such as Mask R-CNN (He et al., 2017). However, in this paper we propose a technique to improve the instance segmentation and object detection, which instead of using the RGB image alone, it includes an extra depth channel which helps to improve the segmentation metrics of peduncles and bunches. Unlike other methods that also use RGB-D images, our method uses a depth channel estimated by monocular depth estimation. In Fig. 8 a diagram of the mentioned process can be seen.

3.6.1. *Models for monocular depth estimation*

The detection of peduncles in a cluttered environment, such as that found in vineyard fields, is very complex due to the number of leaves, branches, partial bunches of grapes with similar color and, above all, due to the shape and variety of peduncles. Using only the RGB image, it is not enough because too many ambiguities appear in the detection of peduncles, and for this reason the use of a depth map can help to eliminate a big part of the background and also separate the foreground grape bunches and peduncles from the rest. In this work, we include the depth as an essential cue that combined with the RGB image can add the extra information needed to improve the image segmentation and also the peduncle detection. The computation of the depth can be done using a stereo camera but again the point cloud obtained is not enough dense. We realized that using a monocular depth network, we can obtain a dense relative depth map that improves the instance segmentation and allows to separate the front objects from the back ones. We have used two different monocular depth methods, MiDaS (Ranftl et al., 2022) which is faster but the resolution is reduced, and Boosting (Miangoleh et al., 2021) that gave better results as can be seen in Fig. 9. It can be appreciated that the Boosting net gives more details, even the grapes of a cluster can be clearly identified. This information is added to the RGB image to improve the instance segmentation process. The architecture of the MiDaS network is based on ResNet. Whereas the Boosting method adopt the Pix2Pix architecture with a 10-layer U-net as generator, to get more detail see Ranftl et al. (2022) and Miangoleh et al. (2021).

The MiDaS implementation provides different models to be used depending on the available hardware resources. For this project, the

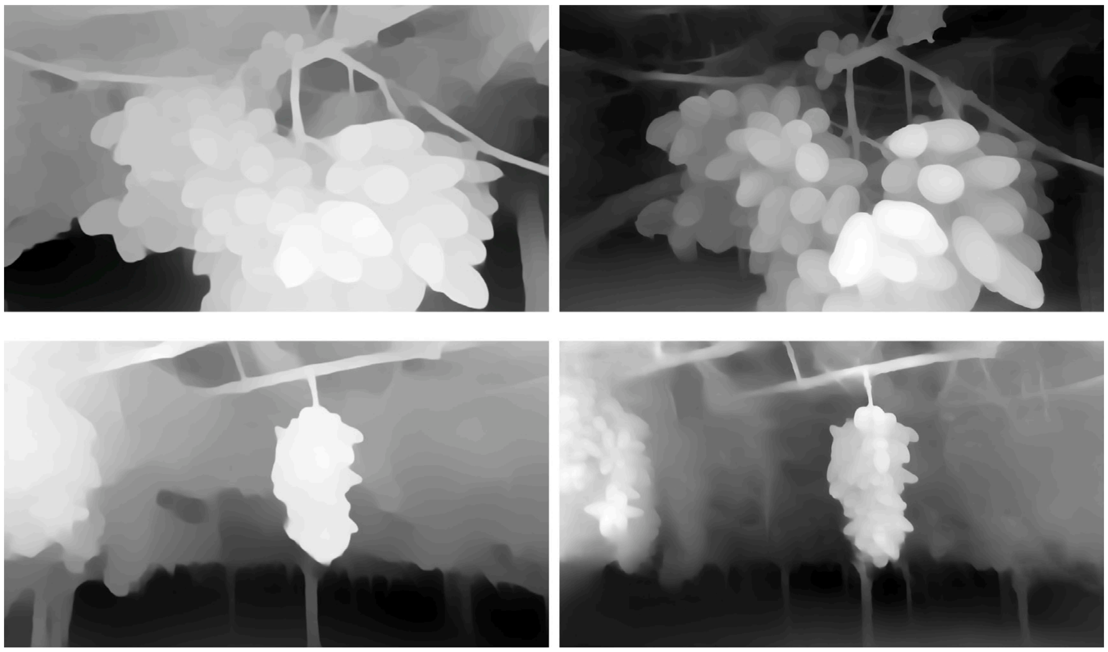

Fig. 9. Examples of application of monocular depth map estimation techniques in our dataset. The left column shows the results obtained by MiDaS (Ranftl et al., 2022). On the right, those obtained by Boosting (Miangoleh et al., 2021).

model "version 2 large" was selected, which yields an absolute relative error ranging from 0.1246 to 0.3270, depending on the selected dataset. The model was able to run at 50 FPS on a RTX3090 graphic card, enabling it to be used for real time applications. It is important to note that the generated depth maps use relative units, ranging from 0 (furthest point) to 1 (nearest point). Additionally, the wide variety of datasets used for training has allowed the method to generalize well to the case of vineyards.

3.6.2. *Improving instance segmentation performance through monocular depth estimation*

The instance detection and segmentation of the different grape peduncles and bunches were performed using Mask R-CNN. However, instead of using only the three channels (RGB) of the image, an additional channel containing the monocular depth estimation was included, as explained in this section.

Mask R-CNN (He et al., 2017) is the culmination of a series of works developed in object detection and instance segmentation (Girshick et al., 2014; Girshick, 2015; Ren et al., 2015). This network is composed of several key components. First, a convolutional backbone extracts features from the input image, capturing both low-level and high-level details. The backbone used was ResNet-101 (He et al., 2016). Next, a region proposal network (RPN) generates candidate object regions, called proposals. These proposals are then refined by a bounding box regression subnetwork, which localizes the objects with high precision. In addition, a mask prediction subnetwork generates pixel-level masks for each object that accurately delineate their shapes. The network also includes a classification subnetwork to classify the detected objects into different categories. More details about the architecture can be found in He et al. (2017) and Abdulla (2017). By combining these components, Mask R-CNN achieves state-of-the-art results in instance segmentation tasks, enabling advanced object detection and segmentation capabilities in various applications. Other instance segmentation networks were also considered. For example, YOLACT training was tested, but was not able to improve the metrics obtained by Mask R-CNN (mAP of 0.632 *vs.* 0.725). The metrics obtained can be seen in Section 4.2, where a comparison of the metrics obtained by different models is shown in Table 5. The performance of the Mask R-CNN in initial tests, along with its extensive implementation in the literature, led to its selection as the network of choice.

In previous works, fruits have been correctly segmented based only on the texture obtained from the RGB images. However, in those cases, there was a contrasting difference in texture between the fruit to be harvested and the rest of the surrounding elements (leaves, branches, soil, etc.). In this project, it is intended to obtain an accurate detection of the peduncles, which have many similarities with elements that should not be identified by the robot, such as branches or posts.

Based on the first results of the applied depth estimation techniques, as shown in Fig. 9, it was observed that these estimated depth maps offer potential for improving instance segmentation results. The results indicate that the estimated depth maps contribute to the creation of contrast between foreground and background elements. In the case of RGB images, the grape bunches exhibit noticeable differences in color and texture compared to the surrounding vineyard environment, making their segmentation less challenging. However, accurate detection of the grapes is more difficult due to their similarity to other elements in the scene, such as branches and leaves. Therefore, it is hypothesized that the incorporation of generated depth images can aid in the segmentation of peduncles. In addition, during the tests, it was found that the depth map of the stereo camera often had difficulties in correctly detecting the peduncles due to the size of them. Therefore, an additional advantage of using the estimated depth maps is their ability to reliably detect the presence of peduncles.

3.6.3. *Instance segmentation model training*

In order to train the Mask-RCNN model, we first expanded the dataset by augmenting the images with an additional channel containing the estimated depth map, resulting in 4-channel RGB-D images. Fig. 10 shows the 4-channels, three for RGB and one for the estimated depth map. Subsequently, a new set of models was trained using this augmented dataset to evaluate the impact of this technique on segmentation results. While MiDaS (Ranftl et al., 2022) was initially considered, the Boosting technique (Miangoleh et al., 2021) was employed for generating the estimated depth maps due to its superior performance, as stated in Miangoleh et al. (2021).

To determine that this combination technique improves the results, Mask R-CNN was trained using both scenarios: with RGB images only

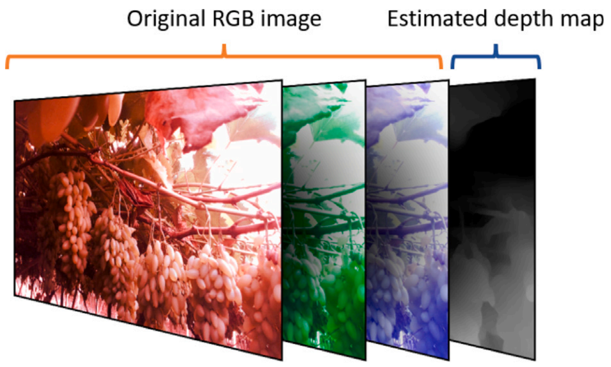

Fig. 10. Assembling the estimated RGB-D images.

Table 1
CANOPIES grape bunch and peduncle dataset split used for training.

	Images	Cluster masks	Peduncles masks	Total masks
Training	984	1434	1434	2868
Validation	274	342	342	684
Test	68	109	109	218

Table 2
WGISD dataset split used for training.

	Images	Cluster masks
Train	110	1612
Test	27	408

and with the estimated RGB-D images. Three different models were trained with each dataset, and the metrics obtained by each set of models (as seen in Eqs.(1), (2)) were compared.

The dataset was divided into 75% for training, 20% for validation and 5% for testing. Table 1 shows the dataset split for the CANOPIES Grape Bunch and Peduncle Dataset, displaying the number of images and instances of each class contained in each one of the sets. Additionally, Table 2 shows the dataset split used for the WGISD dataset. Note that the WGISD dataset does not provide the peduncle class.

For training the model with the RGB dataset, the weights of the network were initialized with a pretrained model on the COCO dataset. For training the model with the RGB-D dataset, the weights of the model that obtained the best metrics with the RGB dataset were used instead, and the model was fine-tuned for the RGB-D images. To do this, the weights of the first layer of the model had to be modified, since the input size was different (now the images had 4 channels instead of 3). Therefore, the last layer of the first convolutional layer has to be determined. It was decided to initialize it as the average of the other three channels. Another option that was considered was to randomly initialize the first layer of the network and keep the weights in the rest of the layers, but the former methodology showed much better results in a faster way.

To further enhance the results obtained by the network and to avoid over-fitting, data augmentation techniques were used. The techniques were selected such that the variations obtained are similar to those that could be seen in a real field scenario. Among the set of enhancements included there is gaussian blur, dropout (some pixels), image rotation (±20 degrees), image scaling (from 50 to 150 degrees), horizontal flip and multiplication (to lower or raise the brightness of the image). From these filters, 0 to 3 (uniform distribution) are applied to each image. In Fig. 11, some examples of the image variations obtained after the augmentation process are shown.

The Mask R-CNN implementation in Keras/Tensorflow developed by Matterport, Inc (Abdulla, 2017) was used. The input images were resized to a size of 1024 × 1024 by rescaling the image and cropping a square section of the image. The training was performed on an NVIDIA GeForce RTX 2080Ti GPU (12 GB memory), placed on a server with an Intel Xeon Silver 4214 CPU and a total of 128 GB RAM. The network weights were initialized with a pre-trained model on the COCO dataset. Three different training stages were used, each training a different set of network layers. In the first stage, only the heads (RPN and the branches for classification, box regression and mask generation) were trained, for 50 epochs. In the second stage, the heads and stage 4 and up of the Resnet backbone were trained for another 150 epochs. Finally, all layers were trained for another 200 epochs, reducing the learning rate by a factor of 10. The total training time following the presented schedule was approximately 13 h.

3.7. Depth estimation of table grapes and peduncles

As we commented previously, the depth estimation of the bunches and peduncles can be very complex, where for example, the bunch can be detected but not the peduncle. This is the case of Fig. 12, where the peduncle is partially occluded by a leaf.

Considering the above complexities, there is a clear need to apply different methods for peduncle depth estimation, taking into account whether the peduncle has been detected or not. For cases where it is not detected, a 3D reconstruction of the bunch is performed and an approximation of the peduncle location is obtained based on the size and pose of the bunch.

To tackle the problem of depth is also crucial to make a good selection of the sensor to use. The most commonly used sensors to address the problem of depth estimation in the real world are LIDAR sensors or stereo cameras. These sensors have different characteristics that will determine which one is the most suitable for this project.

Regarding LIDAR sensors, the first thing to take into account is that they are generally more expensive than stereo cameras. Among LIDARs there are different models in terms of resolution, usually having 4, 16, 64, 128, or 256 beams. For the detection of vine stalks, these resolutions are not sufficient. However, the range is up to 300 m.

Stereo cameras have more depth points than a 16-beam LIDAR. However, resolution remains a problem when dealing with peduncle depth measurement, due to its small size. In tests with a RealSense D435i camera, only a few depth points could be obtained from the peduncle. Another key point is the minimum measurement distance. For the RealSense D435i, the ideal range is 30 centimeters to 3 m.

After a preliminary field study with different sensors, the LIDAR sensor proved not to be a good solution, while the D435i stereo camera had good resolution and had the advantage of providing a dense RGB image for detection. The RealSense D435i was therefore chosen to address the depth problem.

3.7.1. Depth estimation of the peduncle based on grape bunch size

This section explains the method used to estimate the peduncle position and orientation. The method uses the depth map obtained by the stereo vision camera and the segmentation obtained with Mask R-CNN to perform a 3D reconstruction of the bunch and obtain an approximate position of the peduncle. As mentioned in the introduction, this method can be useful when the peduncle depth cannot be measured directly with the stereo camera, because the camera is far away from the cluster and no peduncle measurements are obtained or when the peduncle is partially occluded.

Description of the method

The input data is a set of individual masks of each bunch and peduncle in the scene and associated depth map. Fig. 13 shows the starting image segmentation step, with the input image on the left and the segmentation obtained on the right. As it can be seen, the peduncle is not detected, being an example of a detection failure when the distance to the grapes exceeds the limitations of the detection method.

The diagram of the method is shown in Fig. 14, where all the main computational steps of the method are shown. The first step is to multiple pixel by pixel the mask results by the depth map values.

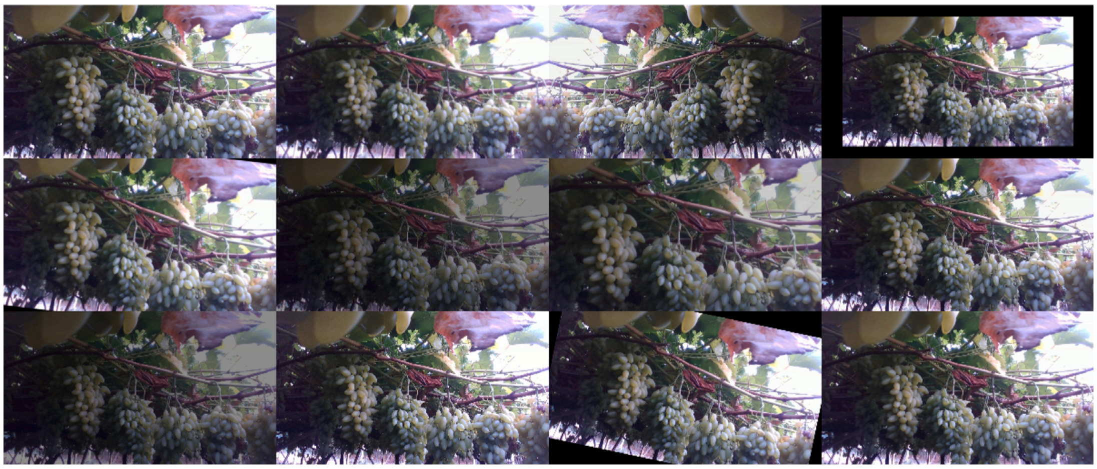

Fig. 11. Image variations obtained by a single dataset example by applying the augmentations used in training.

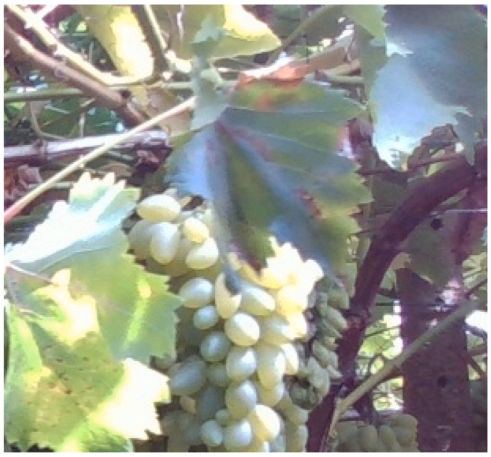

Fig. 12. Example of a bunch peduncle partially occluded by a leaf.

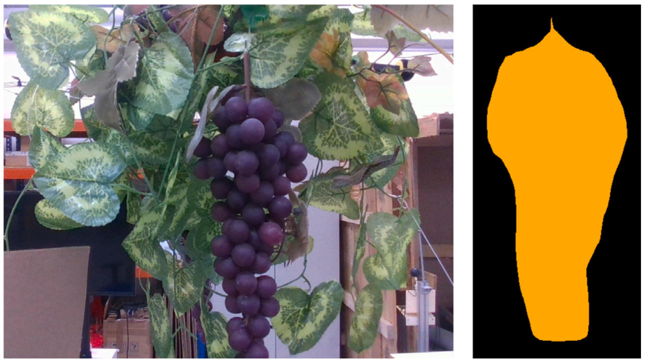

Fig. 13. Left, original RGB image. Right, segmentation obtained from the RGB image.

Although this operation is simple, some problems can occur. Usually, the mask does not fit perfectly the grape bunch, which produces the appearance of some close or distant points that are not part of the bunch. Moreover, it can occur that there is a small partial occlusion of a branch or a leaf in front of the bunch that also will affect the result. The result of this multiplication in the case that there are not problems is shown in Fig. 15 left.

Because of the mentioned problems, the next step is a filtering process. This process has different operations: (1) all points with a distance higher than 3 meters and lower than 25 centimetres are filtered out; and (2) to filter the noise in the bunch itself, all points that are more than 8 centimetres away from the median are eliminated. Doing this and keeping only the bunch information, the result obtained is shown in right side of Fig. 15.

As shown in Fig. 14, the next step is to estimate the bunch diameter. We define k as 1% of N, where N is the number of 3D points that compose the bunch. To calculate the diameter (D) of the bunch, the median of the x-coordinates of the rightmost k points (x_M) and the median of the x-coordinates of the leftmost k points (x_m) are calculated. The diameter is calculated as the difference of these two values, as shown in Eq. (4). The next step is to determine the point of the bunch closest to the camera. The z-coordinate (z_m) of this point is obtained by calculating the median of the k points with the lowest z, as shown in Eq. (5). Finally, the z-coordinate of the peduncle is obtained by adding the bunch radius ($D/2$) to the z-coordinate of the nearest bunch point, as shown in Eq. (6).

$$\begin{gathered}\text{Set of N 3D points: } x = [x_1, \dots, x_N],\ y = [y_1, \dots, y_N],\ z = [z_1, \dots, z_N] \\ k = \lfloor 0.01N \rfloor \\ x_M = \text{median}\,(\text{sort}(x, \text{descending})[1:k]) \\ x_m = \text{median}\,(\text{sort}(x)[1:k]) \\ D = x_M - x_m\end{gathered} \tag{4}$$

$$z_m = \text{median}\,(\text{sort}(z)[1:k]) \tag{5}$$

$$z = z_m + \frac{D}{2} \tag{6}$$

The final estimated position of the peduncle and its reconstruction can be seen in Fig. 16.

One good feature of this method is that it could be used for other fruits. In fact, the process is useful to know the depth of the centroid of symmetrical objects.

The detailed algorithm of this method can be seen in Algorithm 1.

3.7.2. Depth estimation of peduncle using the depth map

This section explains the peduncle depth estimation method using the depth map of the stereo camera.

Description of the method

Fig. 17 shows the diagram of the method, where the main computational steps are shown. The input data are the segmented image and the depth map. As it was done in the previous method, the first step is to multiply the mask by the depth map, so that only the depth information of the peduncle is preserved.

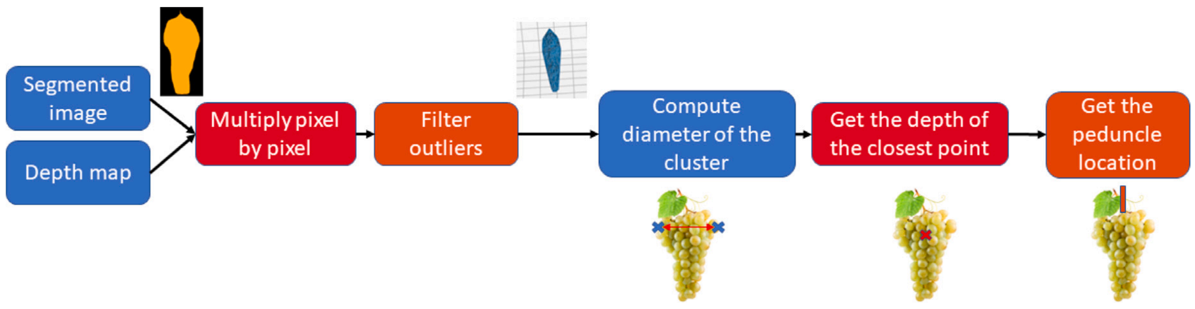
Segmented image
Depth map
Multiply pixel by pixel
Filter outliers
Compute diameter of the cluster
Get the depth of the closest point
Get the peduncle location

Fig. 14. Diagram of the peduncle depth estimation from bunch method.

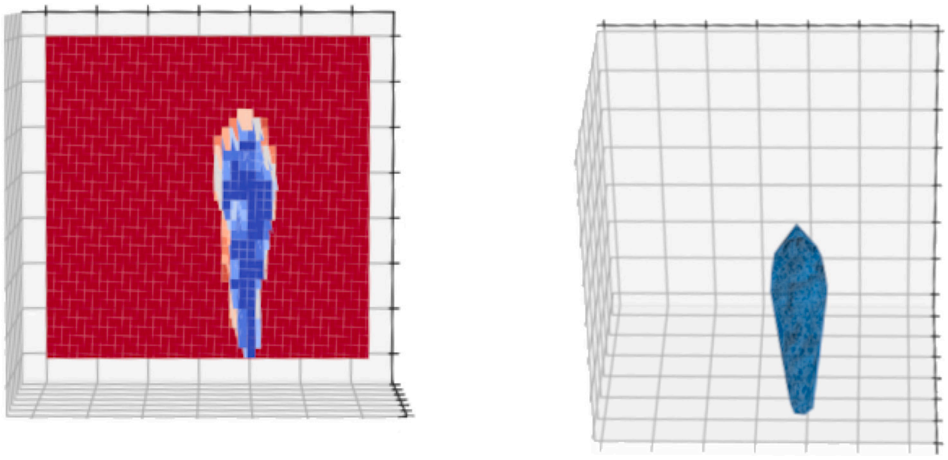

Fig. 15. Left, 3D points from a cluster. Right, 3D reconstruction after filtering.

Algorithm 1 Peduncle from cluster algorithm

```
Input: image_depth ← depth image from the Realsense
Input: imshape ← shape of the image
Input: masks ← (height × width × n) structure with the masks of the clusters and peduncles detected, being n the number of detected objects
Input: ids ← array of classes of the detected objects ▷ ids = 2 means that the mask corresponds to a cluster
Input: scores ← array of scores of the detected objects
Output: index ← array of indexes of the processed objects
Output: depths ← array of depths of the processed objects
for $i < len(scores)$ do
    if ids[i]==2 then
        index.append(i)
        roi ← image_depth * masks[i] ▷ Get the information of depth from the ROI
        roi_filtered ← filterByDistance(roi,25,300) ▷ Erase points closer than 25 cm and further than 3 meters
        median_value ← median(roi_filtered)
        roi_filtered ← filterByMedian(roi_filtered,8) ▷ Erase point that have a deviation of 8 cm with respect to the median
        (left_border,right_border) ← obtainBorders(roi_filtered,1) ▷ This function gets the borders of the mask using the percentage of points to take into account
        depth ← obtainDepth(roi_filtered,left_border,right_border) ▷ This function applies Equation (4)
        depths.append(depth)
    end
end
```

The next step is filtering the outliers of the resulting point cloud, by deleting the depth values that are not between 25 cm and 1 meter. The result of this process is that the majority of the depth values (75% of them) are from the background, due to segmentation errors and pointcloud errors of the stereo camera. The next step is to select a depth value that is in the 10% of the closest values, to be sure that is not a depth value of the background. Then finally, we obtain the z coordinate of the (x, y, z) point.

The detail of this method can be seen in Algorithm 2.

Algorithm 2 Direct measure algorithm

```
Input: image_depth ← depth image from the Realsense
Input: imshape ← shape of the image
Input: masks ← (height × width × n) structure with the masks of the clusters and peduncles detected, being n the number of detected objects
Input: ids ← array of classes of the detected objects ▷ ids = 1 means that the mask corresponds to a peduncle
Input: scores ← array of scores of the detected objects
Output: index ← array of indexes of the processed objects
Output: depths ← array of depths of the processed objects
for $i < len(scores)$ do
    if ids[i]==1 then
        index.append(i)
        roi ← image_depth * masks[i] ▷ Get the information of depth from the ROI
        roi_filtered ← filterByDistance(roi,25,100) ▷ Erase points closer than 25 cm and further than 1 meters
        median_value ← median(roi_filtered)
        depth ← percentile(roi_filtered,10)[z] ▷ Get z coordinate of the point corresponding to the tenth percentile of the roi_filtered
    end
end
```

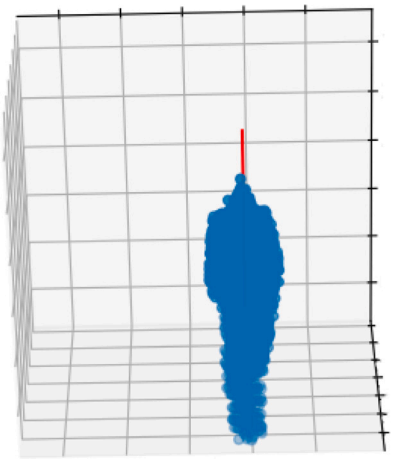

Fig. 16. Peduncle position estimation ant its reconstruction.

Table 3
Instance Segmentation results for the best model trained with the RGB images, obtained for the validation and test subsets (confidence score threshold set to 0.8).

IoU	P	R	F1	mAP
0.3	0.593	0.676	0.631	0.725
0.4	0.582	0.667	0.622	0.701
0.5	0.561	0.651	0.602	0.654
0.6	0.498	0.598	0.543	0.542
0.7	0.366	0.509	0.415	0.365
0.8	0.263	0.433	0.288	0.230
0.9	0.037	0.269	0.058	0.027

IoU	$P_{cluster}$	$R_{cluster}$	$F1_{cluster}$	$AP_{cluster}$	P_{ped}	R_{ped}	$F1_{ped}$	AP_{ped}
0.3	0.582	0.683	0.629	0.647	0.603	0.668	0.634	0.802
0.4	0.564	0.667	0.611	0.608	0.600	0.667	0.632	0.793
0.5	0.523	0.635	0.574	0.537	0.598	0.666	0.630	0.771
0.6	0.414	0.547	0.471	0.387	0.583	0.649	0.614	0.696
0.7	0.164	0.379	0.229	0.141	0.567	0.639	0.601	0.589
0.8	0.015	0.274	0.029	0.009	0.511	0.591	0.548	0.450
0.9	0.000	0.269	0.000	0.000	0.075	0.268	0.117	0.054

4. Results

4.1. Experimental setup

The experiments performed for this article has been done in a field located at Aprilia, Italy. The field uses as structure the double-roof system as can be seen on Fig. 18.

The varieties of the grapes used to carry out the experiments are the “Pizzutello Bianco” and the “Red Globe” shown in Fig. 20.

The robot used is an evolution of the Tiago++ robot manufactured by PAL Robotics. This robot was specifically designed for the CANOPIES European project. It has one Realsense D435i stereo camera placed on the wrist and another one placed in the head. For these experiments the most used one is the one on the wrist.

The Realsense D435i camera sensor was configured with the following parameters. For the RGB sensor, the resolution was set to 1280×720 pixels, and frames were captured at a rate of 30 frames per second. The camera used a rolling shutter mechanism, while the RGB sensor had a field of view (FOV) of 69 degrees horizontally and 42 degrees vertically. In terms of depth settings, the camera used stereoscopic depth technology, providing a depth field of view of 87 degrees horizontally and 58 degrees vertically. At maximum resolution, the minimum depth distance (Min-Z) was set to 28 cm. The depth output resolution was configured to be 1280×720 pixels, the same as the RGB camera, and the depth images were captured at a rate of 30 frames per second. Additionally, as an alternative to the camera-based depth measurements, a STANLEY TLM130i laser range finder was used to obtain ground truth depth information. This laser system provided a resolution of ± 2 mm for accurate depth measurements.

Fig. 19 shows the process of obtaining ground truth depth values using a laser meter located in one of the arms of the CANOPIES robot.

4.2. Instance segmentation results

In this section, we present the instance segmentation metrics obtained with the Mask R-CNN models trained with the images obtained by combining the RGB images with the monodepth, following the method specified in Section 3.6.2 To analyze whether the proposed method improves the segmentation as hypothesized, the metrics obtained by training only with the RGB images are also presented. The dataset used for training was the CANOPIES Grape Bunch and Peduncle Dataset presented in Section 3.3.

Due to the possibility that the training can bring the network to a local minimum, comparing the results obtained based on a single example can lead to incorrect conclusions. Instead, we trained Mask R-CNN three times with RGB images and three times with RGB images combined with monodepth, obtaining two groups with three distinct models each. As it was mentioned in Section 3.6.2, the dataset was divided into 75% for training, 20% for validation and 5% for testing (see Table 1).

Table 3 presents the instance segmentation results for the model that obtained a higher mAP from the group of models trained with RGB images alone. The upper table shows the overall metrics (P, R, F1 and mAP) of the method for different $IoUs$. The lower table shows the metrics for each class, also for different $IoUs$. Table 4 presents the same set of metrics but for the model that obtained a higher mAP from the group of models trained with RGB-D images. Fig. 21 shows the comparison of the mAP and AP of each class obtained by all models trained with RGB images and with RGB-D images. As can be seen in them, the use of estimated RGB-D images produces an increase of the model performance. The AP for the cluster class ($IoU = 0.3$) increases its value from 0.647 for RGB, to 0.688 for RGB-D (6.3% better). For the peduncle class, the AP ($IoU = 0.3$), increases its value from 0.802 for RGB, to 0.845 for RGB-D (5.4% better). Overall, the mAP value increases from 0.725 for RGB to 0.767 for RGB-D (5.8% better). This trend continues for other IoU values. After comparing, it can be seen that the proposed method has succeeded in increasing the metrics.

Fig. 22, shows the segmentation obtained with the Mask-RCNN model trained with RGB-D images. Only detections with a confidence score above 0.95 are shown, discarding those with a lower score. As explained in Section 3.3, bunches whose peduncle was not visible were not labeled in the dataset. As a consequence of this condition, the model tends (as intended) to detect cluster/peduncle pairs that are in the foreground and with a visible peduncle.

Table 5 compares the performance of Mask R-CNN and YOLACT using the instance segmentation metrics used in the previous examples. The metrics are those obtained by training the models only on the RGB images (without RGB-D augmentation) in the Canopies dataset. Two different backbones were used in the YOLACT tests: ResNet-101 and ResNet-50 (abbreviated in the table as R-101 and R-50, respectively). A confidence threshold of 0.8 was used for comparison, and the table shows the results for an IoU threshold of 0.3. As can be seen, Mask R-CNN outperforms YOLACT in each of the metrics. A point worth noting is the low performance when the backbone was ResNet-101. The performance is significantly lower than ResNet-50 for this dataset, contrary to what would be expected with more parameters in the backbone. As this is beyond the scope of this paper, this fact was not analyzed further.

4.2.1. WGISD dataset

After analyzing in our dataset how the method of combining the RGB images with the monodepth depth map affects the segmentation,

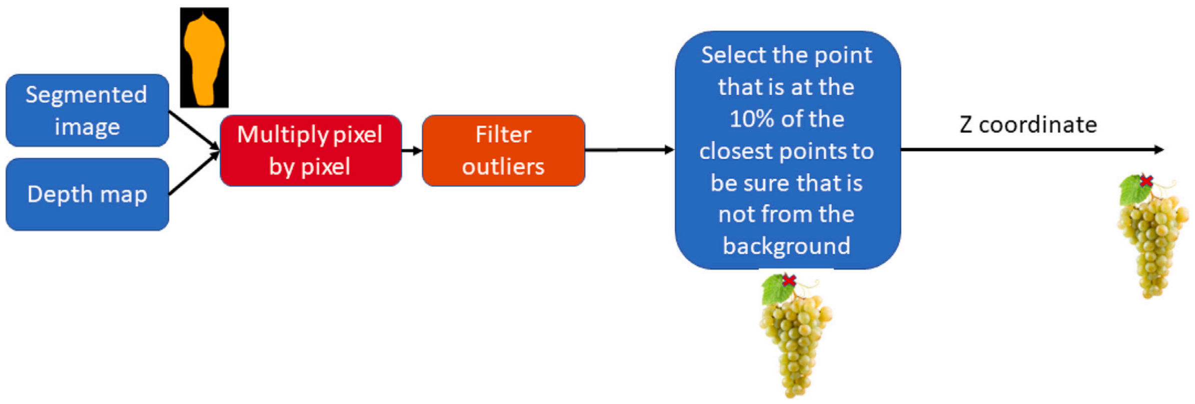
Segmented image
Depth map
Multiply pixel by pixel
Filter outliers
Select the point that is at the 10% of the closest points to be sure that is not from the background
Z coordinate

Fig. 17. Diagram of the direct measure method.

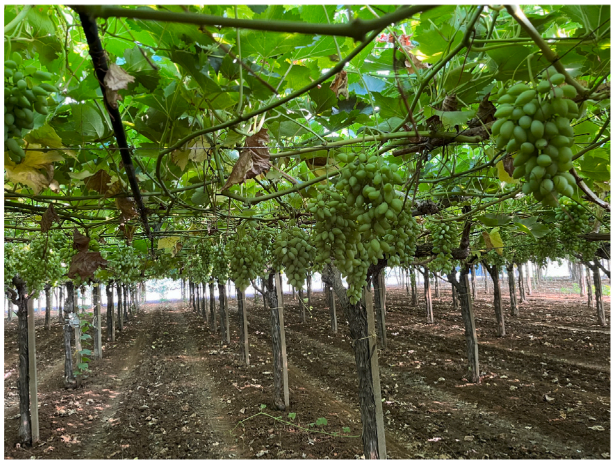

Fig. 18. Double-roof system.

Table 4
Instance Segmentation results for the best model trained with the RGB-D images. Obtained for the validation and test subsets (confidence score threshold set to 0.8).

IoU	P	R	F1	mAP
0.3	0.663	0.575	0.616	0.767
0.4	0.639	0.551	0.592	0.722
0.5	0.580	0.505	0.540	0.619
0.6	0.481	0.435	0.456	0.473
0.7	0.367	0.372	0.358	0.338
0.8	0.277	0.311	0.256	0.231
0.9	0.019	0.145	0.028	0.012

IoU	$P_{cluster}$	$R_{cluster}$	$F1_{cluster}$	$AP_{cluster}$	P_{ped}	R_{ped}	$F1_{ped}$	AP_{ped}
0.3	0.649	0.587	0.617	0.688	0.678	0.563	0.615	0.845
0.4	0.605	0.543	0.572	0.614	0.674	0.558	0.611	0.829
0.5	0.500	0.460	0.479	0.451	0.659	0.551	0.600	0.788
0.6	0.322	0.338	0.329	0.245	0.640	0.533	0.582	0.701
0.7	0.111	0.222	0.148	0.063	0.623	0.521	0.567	0.612
0.8	0.014	0.181	0.026	0.007	0.539	0.442	0.486	0.455
0.9	0.000	0.175	0.000	0.000	0.037	0.114	0.056	0.025

Fig. 19. CANOPIES robot handled by a human.

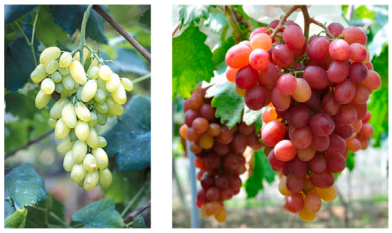

Fig. 20. Left, Pizzutello Bianco variety. Right, Red Globe variety.

this method was also studied in the WGISD dataset. In order to evaluate whether the proposed method improves the segmentation or not, it was used the same procedure explained before. Three models were trained on the RGB images and three others on the combined images (RGB-D). We used the same dataset split for training and test of the original work, as is shown in Table 2.

Table 6 shows the metrics obtained by the model that obtained a higher mAP from the group of models trained with RGB images alone (left table) and the one that obtained a higher mAP from the group of models traind with RGB-D images (right table). Note that the WGISD dataset does not contain peduncle information, so only bunch class metrics are available. As can be seen, the proposed method also produces an increase in the performance, increasing the mAP (IoU = 0.3) from 0.891 using RGB images to 0.949 when using RGB-D images (6.5% better). For other IoU values, the results present the same trend. Fig. 23 shows the comparison of the metrics obtained by all models trained on the WGISD dataset.

In conclusion, the proposed method has been successfully applied to two distinct scenarios: the WGISD dataset, characterized by isolated vineyards where grape bunches hang freely from the vineyard trees, and the CANOPIES Grape Bunch and Peduncle dataset, where grape bunches are constrained by the canopy. These scenarios exhibit significant variations in terms of bunch height, distribution, illumination conditions, and background characteristics. Remarkably, the method outperformed state-of-the-art techniques in both scenarios, demonstrating its effectiveness and robustness across diverse grape harvesting settings.

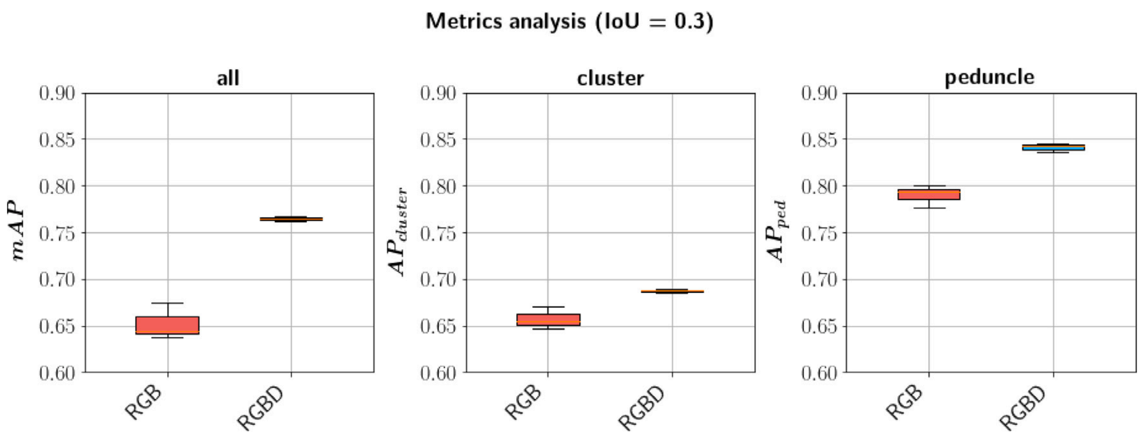

Fig. 21. Metrics comparison.

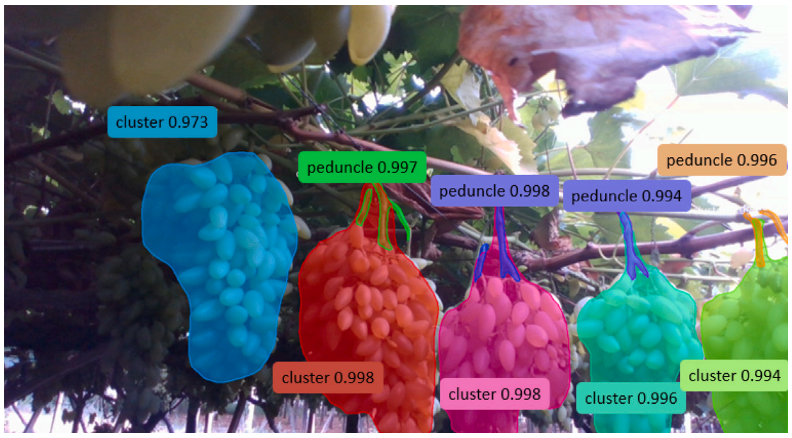

Fig. 22. Segmentation obtained with the Mask-RCNN model trained with RGB-D images. It is only shown the detections with a confidence score above 0.95.

Table 5
Instance Segmentation metrics comparison between Mask R-CNN and YOLACT (with two different backbones). Results obtained by training only with the RGB images of the CANOPIES dataset. A confidence threshold of 0.8 and an IoU threshold of 0.3 have been used.

Model	P	R	F1	mAP
Mask R-CNN	0.593	0.676	0.631	0.725
YOLACT (R-50)	0.517	0.579	0.546	0.632
YOLACT (R-101)	0.177	0.202	0.188	0.216

Model	$P_{cluster}$	$R_{cluster}$	$F1_{cluster}$	$AP_{cluster}$	P_{ped}	R_{ped}	$F1_{ped}$	AP_{ped}
Mask R-CNN	0.582	0.683	0.629	0.647	0.603	0.668	0.634	0.802
YOLACT (R-50)	0.502	0.580	0.538	0.565	0.537	0.578	0.553	0.699
YOLACT (R-101)	0.174	0.204	0.187	0.193	0.180	0.199	0.189	0.239

Table 6
Instance Segmentation results for the model trained with the WGISD dataset.

(a) Metrics WGISD (RGB)					(b) Metrics WGISD (RGB-D)				
IoU	P	R	F1	mAP	IoU	P	R	F1	mAP
0.3	0.971	0.875	0.920	0.891	0.3	0.956	0.938	0.947	0.949
0.4	0.964	0.868	0.913	0.887	0.4	0.956	0.938	0.947	0.949
0.5	0.931	0.839	0.882	0.859	0.5	0.933	0.814	0.924	0.925
0.6	0.887	0.799	0.841	0.816	0.6	0.906	0.888	0.897	0.895
0.7	0.799	0.720	0.758	0.728	0.7	0.839	0.822	0.830	0.823
0.8	0.515	0.464	0.488	0.434	0.8	0.564	0.553	0.558	0.511
0.9	0.104	0.102	0.103	0.076	0.9	0.073	0.066	0.069	0.039

4.3. Depth estimation results

The results of the depth estimation were evaluated using different data sequences of the real vineyard, containing images and pointclouds. The real depth of the peduncle was known for each sequence. These measurements were done using a laser meter and they ranged between 35–100 cm. The total number of measurements used for the experiments were approximately 600. A scheme of the interface for displaying the results of the methods presented in this work can be seen in Fig. 24. On the left image is shown the original image and in the left image, the segmented image with the depth distance.

On Table 7, it can be seen a summary of the depth results obtained with the peduncle from cluster and from direct measure methods. It was a project requirement that the estimated depth error falls inside the range of ± 3 cm.

The precision of the direct measure method is superior, as it incorporates information about the actual peduncle. Nonetheless, this method requires the robot to be in close proximity to the bunches in order to identify the peduncle. In instances where this condition is not met, the peduncle from bunch method provides an accurate approximation of the peduncle location.

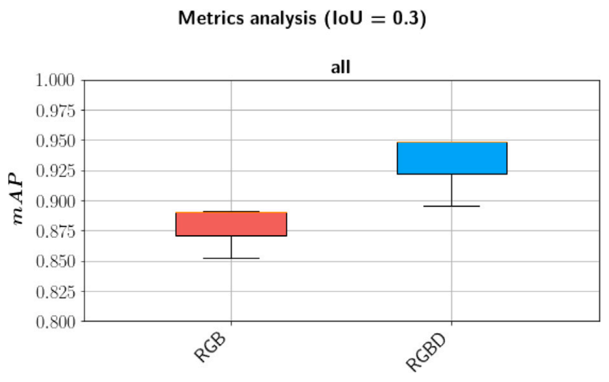

Fig. 23. Comparison of the mAP obtained in the WGISD dataset.

Table 7
Results of the depth estimation.

Method	Valid measurements (%)	Mean absolute error (cm)	Root mean squared error (cm)
Peduncle from cluster	74.7	2.18	2.66
Direct measure	85.0	1.82	2.25

To see how the error of the method is distributed over the distances, a plot of the deviations for different distances has been made. The plot corresponding to the clustering method can be seen in Fig. 25. Note that the horizontal axis shows the ground truth distance measured using the laser range sensor, and the vertical axis shows the distance measured by the peduncle from cluster method.

The results obtained show a substantial level of accuracy and precision, with negligible deviations, except for the limited range where the stereo camera performs suboptimally. The plot corresponding to the direct measurement method can be seen in Fig. 26, where the vertical axis shows the results of the direct measure method. The results in this case show the same problem as in the peduncle from bunch method. Although we have only shown the experiments to obtain the depth to the peduncle, we also did experiments to obtain the (x, y, z) coordinates of the peduncle, because this data is required for the robot to reach the picking peduncle point. In this case, the ground truth was to reach the peduncle with the CANOPIES robot, and the experiments that we did shown that the robot was able to reach the peduncle, within the error interval.

Regarding computational efficiency, the entire process of segmentation and depth estimation was executed in 1.42 s, with a standard deviation of 0.31 s, on a laptop equipped with an Intel Core i7-10750H processor, 16 GB of RAM memory, and a Nvidia RTX2060 graphics card.

5. Discussion

Regarding the bunch and peduncle segmentation, it can be seen from Fig. 21 that the proposed method of combining RGB and monodepth images improves the segmentation. A similar performance was obtained for the peduncle class. On the other hand, using the RGB-D images improves the mAP both in general (maximum mAP of 0.767 instead of 0.725) and by class (from 0.647 to 0.688 in the bunch class and from 0.802 to 0.845 in the peduncle class).

As shown, the proposed method also improves the results on the WGISD dataset. A maximum mAP of 0.949 was obtained in the model trained with this technique, in contrast to the maximum mAP of 0.891 obtained in the model trained with RGB images only (for an IoU threshold of 0.3). The only metric that has a small decrease is the precision, from 0.971 to 0.956 (for an IoU threshold of 0.3).

Table 8
Computation time (in seconds) and frames per second (FPS).

		Mask	Mask+Boost
Time (s)	Mean	0.2646	1.9249
	StDev	0.0188	1.7831
FPS	Mean	3.7792	0.5195
	StDev	0.0188	1.8161

However, this performance improvement is not without a cost. Table 8 shows the computational time required to perform the different techniques. It was found that it took an average of 1.6603 s per image (with a standard deviation of 0.6783 s) to estimate the depth map using Boosting Monodepth. This brings the total computation time (depth map estimation and Mask R-CNN segmentation) to 1.9249 s per image.

The proposed method has certain limitations. In real-world conditions, the detection step works well up to a distance of 2 m. However, between 70 centimeters and 2 m, it becomes difficult to accurately detect the peduncle due to its small size. In terms of depth estimation, the method achieves reliable results in the range of 30 centimeters to 1 meter, with an error of less than 3 centimeters. This level of accuracy meets the required accuracy for the project.

Moreover, the methods developed in this project enable the estimation of depth even when the peduncle is not directly visible due to either distance or occlusions. These represent the primary challenges encountered in accurately estimating the depth of the peduncle.

The methods have been tested with different grape varieties in different growth moments. According to the different experiments, the most accurate measure to do the approach to the peduncle is the one obtained by the direct method.

In terms of computation time, it is observed that the depth estimation process introduces a significant increase in the standard deviation, about 15 times higher compared to the segmentation step. This variation is due to the direct dependence of the computation time on the number of detected bunches and peduncles in an image. As the complexity of the scene increases with more instances to process, the computation time for depth estimation tends to become more variable.

The improvement in instance segmentation of grape bunches and peduncles in cluttered backgrounds is attributed to the effective combination of RGB and monodepth images. It should be noted that the computation of the monodepth map significantly contributes to the overall computational time. However, the continuous advancements in monodepth algorithms have resulted in reduced computational overhead and enhanced performance. This progress paves the way for more accurate measurements using cost-effective hardware. Another way to improve the performance is using more precise and smaller RGB-D sensors with smaller minimum depth distance.

These techniques, initially developed for grape bunches and peduncles, can be readily applied to a wide range of fruits, vegetables, flowers, and other produce. The precise detection of peduncles is crucial in various harvesting and pruning operations, especially in cluttered environments characterized by densely intertwined branches.

6. Conclusions

In conclusion, this paper provides a complete functional solution for the perception needed to perform a grape harvesting process. By combining instance segmentation and monocular depth estimation techniques, we have obtained a method that is able to detect grape bunches and peduncles in cluttered environments, with better results than other state-of-the-art methods. Regarding the estimation of the distance to the peduncle, a technique was developed that was able to locate the 3D coordinates of the peduncle using an inexpensive stereo camera, thus allowing the implementation of this technique for precision agriculture techniques. The proposed technique perfectly met the accuracy specifications required by the project.

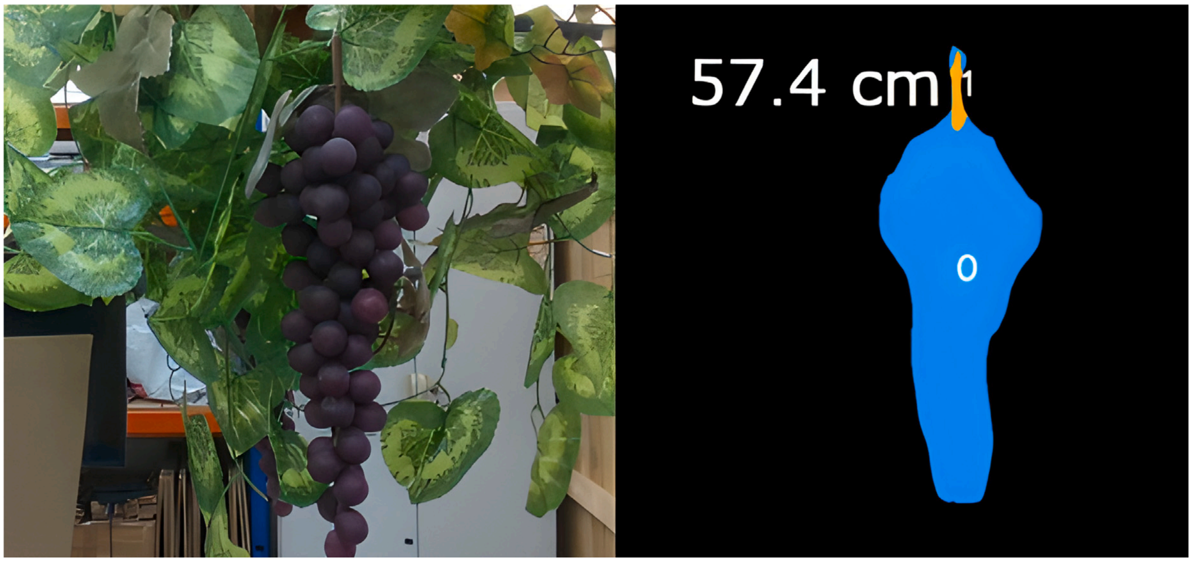

Fig. 24. Scheme of the display used to show the results.

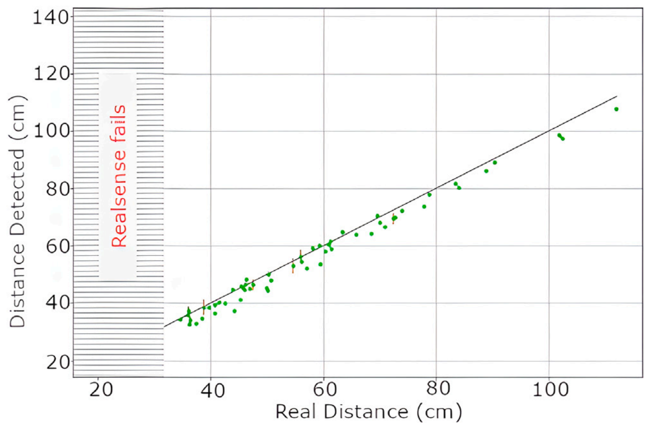

Fig. 25. Peduncle from bunch results.

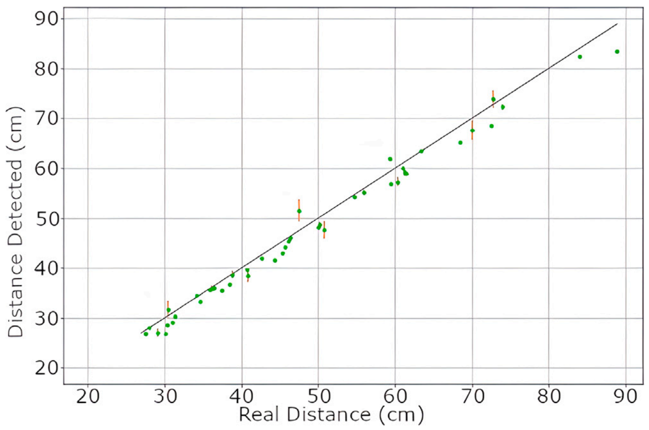

Fig. 26. Direct measure results.

This solution has been designed to work with a common robotics framework such as ROS, making it suitable for use with any robotic platform designed for vineyard harvesting. The solution has not only been developed and tested in the laboratory, but has also been tested and optimized to work in real fields with different grape varieties. The results are analyzed in terms of accuracy, but also include performance information related to computing time.

Although the solution is fully functional, there are things that can be improved as future work. One of them is to migrate the code to C++ for better performance. In addition, it will be essential to expand the datasets for segmentation and distance measurement in order to obtain better models and also to have more data to evaluate the results. In addition to enriching the dataset with more positive detection cases, it is crucial to add more negative cases that should not be falsely detected as bunches or peduncles: humans, machines, animals, etc.. This is crucial to ensure the robustness of the method.

Declaration of competing interest

The work presented in this work was partially supported by the European Commission under the grant (H2020-ICT-2020-2-101016906).

Data availability

We have used two datasets: (1) “Grape vineyard detection and localization of the peduncles” created in the EU Canopies project; (2) “The Embrapa Wine Grape Instance Segmentation Dataset”.

Acknowledgments

Work supported under the European project CANOPIES with the grant (H2020- ICT-2020-2-101016906). The authors want to acknowledge the help and support of the Sapienza Universitá di Roma and the Universitá degli Studi Roma Tre researchers of the CANOPIES project in the data collection of the grapes images and ground truth data.

References

Abdulla, W., 2017. Mask R-CNN for Object Detection and Instance Segmentation on Keras and TensorFlow. GitHub repository.

Bac, C.W., Hemming, J., Tuijl, B., Barth, R., Wais, E., Van Henten, E., 2017. Performance evaluation of a harvesting robot for sweet pepper. J. Field Robotics 34, http://dx.doi.org/10.1002/rob.21709.

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39 (12), 2481–2495.

Bochkovskiy, A., Wang, C., Liao, H.M., 2020. YOLOv4: Optimal speed and accuracy of object detection. CoRR abs/2004.10934. arXiv:2004.10934. URL https://arxiv.org/abs/2004.10934.

Bolya, D., Zhou, C., Xiao, F., Lee, Y.J., 2019. Yolact: Real-time instance segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9157–9166.

CANOPIES, 2021. CANOPIES EU project website. https://cordis.europa.eu/project/id/101016906/es. Accessed: 18-04-2022.

De-An, Z., Jidong, L., Wei, J., Ying, Z., Yu, C., 2011. Design and control of an apple harvesting robot. Biosyst. Eng. 110 (2), 112–122.

Dutta, A., Gupta, A., Zissermann, A., 2016. VGG image annotator (VIA). http://www.robots.ox.ac.uk/~vgg/software/via/.

Dutta, A., Zisserman, A., 2019. The VIA annotation software for images, audio and video. In: Proceedings of the 27th ACM International Conference on Multimedia. pp. 2276–2279.

Eigen, D., Puhrsch, C., Fergus, R., 2014. Depth map prediction from a single image using a multi-scale deep network. Adv. Neural Inf. Process. Syst. 27.

Fang, Y., Yang, S., Wang, X., Li, Y., Fang, C., Shan, Y., Feng, B., Liu, W., 2021. Instances as queries. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6910–6919.

FAO, Food and Agriculture Organization of the United Nations (FAO), 2016. Migration, Agriculture and Rural Development: Addressing the Root Causes of Migration and Harnessing its Potential for Development. Tech. Rep., Food and Agriculture Organization, URL http://www.fao.org/3/a-i6064e.pdf.

Ghiani, L., Sassu, A., Palumbo, F., Mercenaro, L., Gambella, F., 2021. In-field automatic detection of grape bunches under a totally uncontrolled environment. Sensors 21 (11), 3908. http://dx.doi.org/10.3390/s21113908.

Girshick, R., 2015. Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1440–1448.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 580–587.

Grady, L., 2006. Random walks for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 28 (11), 1768–1783. http://dx.doi.org/10.1109/TPAMI.2006.233.

Guo, C., Zheng, S., Cheng, G., Zhang, Y., Ding, J., 2023. An improved YOLO v4 used for grape detection in unstructured environment. Front. Plant Sci. 14, http://dx.doi.org/10.3389/fpls.2023.1209910.

Han, K.-S., Kim, S.-C., Lee, Y.-B., Kim, S.-C., Im, D.-H., Choi, H.-K., Hwang, H., 2012. Strawberry harvesting robot for bench-type cultivation. J. Biosyst. Eng. 37 (1), 65–74.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2961–2969.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Hosseini Minagoleh, S.M., Dille, S., Mai, L., Paris, S., Aksoy, Y., 2021. Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9680–9689. http://dx.doi.org/10.1109/CVPR46437.2021.00956.

Huang, Z., Huang, L., Gong, Y., Huang, C., Wang, X., 2019. Mask scoring r-cnn. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6409–6418.

Jiang, Y., Liu, J., Wang, J., Li, W., Peng, Y., Shan, H., 2022. Development of a dual-arm rapid grape-harvesting robot for horizontal trellis cultivation. Front. Plant Sci. 13.

Koch, T., Liebel, L., Fraundorfer, F., Körner, M., 2019. Evaluation of CNN-based single-image depth estimation methods. In: Computer Vision – ECCV 2018 Workshops. Springer International Publishing, pp. 331–348.

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: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. Springer, pp. 740–755.

Mai, X., Zhang, H., Jia, X., Meng, M., 2020. Faster R-CNN with classifier fusion for automatic detection of small fruits. IEEE Trans. Autom. Sci. Eng. PP, 1–15. http://dx.doi.org/10.1109/TASE.2020.2964289.

Majeed, Y., Zhang, J., Zhang, X., Fu, L., Karkee, M., Zhang, Q., Whiting, M., 2018. Apple tree trunk and branch segmentation for automatic trellis training using convolutional neural network based semantic segmentation. IFAC-PapersOnLine 51, 75–80. http://dx.doi.org/10.1016/j.ifacol.2018.08.064.

Miangoleh, S.M.H., Dille, S., Mai, L., Paris, S., Aksoy, Y., 2021. Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9685–9694.

Pinheiro, I., Moreira, G., Silva, D., Magalhães, S., Valente, A., Moura Oliveira, P., Cunha, M., Neves Dos Santos, F., 2023. Deep learning YOLO-based solution for grape bunch detection and assessment of biophysical lesions. Agronomy 13, 1120. http://dx.doi.org/10.3390/agronomy13041120.

Qiao, S., Chen, L.-C., Yuille, A., 2021. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10213–10224.

Ranftl, R., Bochkovskiy, A., Koltun, V., 2021. Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12179–12188.

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V., 2022. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Trans. Pattern Anal. Mach. Intell. 44 (3).

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

Rother, C., Kolmogorov, V., Blake, A., 2004. GrabCut: Interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. 23, 309–314. http://dx.doi.org/10.1145/1186562.1015720.

Roy, A., Todorovic, S., 2016. Monocular depth estimation using neural regression forest. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5506–5514.

Sa, I., Lehnert, C., English, A., McCool, C., Dayoub, F., Upcroft, B., Perez, T., 2017. Peduncle detection of sweet pepper for autonomous crop harvesting—combined color and 3-D information. IEEE Robot. Autom. Lett. 2 (2), 765–772.

Santo, T., de Souza, L., dos Santos, A., Avila, S., 2019. Embrapa wine grape instance segmentation dataset – embrapa WGISD. http://dx.doi.org/10.5281/zenodo.3361735.

Santos, T.T., de Souza, L.L., dos Santos, A.A., Avila, S., 2020. Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association. Comput. Electron. Agric. 170, 105247.

Saxena, A., Sun, M., Ng, A., 2009. Learning 3D scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. 31 (5), 824–840.

Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22 (8), 888–905. http://dx.doi.org/10.1109/34.868688.

Sozzi, M., Cantalamessa, S., Cogato, A., Kayad, A., Marinello, F., 2022. Automatic bunch detection in white grape varieties using YOLOv3, YOLOv4, and YOLOv5 deep learning algorithms. Agronomy 12, http://dx.doi.org/10.3390/agronomy12020319.

Vrochidou, E., Tziridis, K., Nicolaou, A., Kalampokas, T., Papakostas, G., Pachidis, T., Mamalis, S., Koundouras, S., Kaburlasos, V., 2021. An autonomous grape-harvester robot: Integrated system architecture. Electronics 10, 1056. http://dx.doi.org/10.3390/electronics10091056.

Vu, T., Kang, H., Yoo, C.D., 2021. Scnet: Training inference sample consistency for instance segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 2701–2709.

Wan, S., Goudos, S., 2019. Faster R-CNN for multi-class fruit detection using a robotic vision system. Comput. Netw. 168, 107036. http://dx.doi.org/10.1016/j.comnet.2019.107036.

Xu, Z., Liu, J., Wang, J., Cai, L., Jin, Y., Zhao, S., Xie, B., 2023. Realtime picking point decision algorithm of trellis grape for high-speed robotic cut-and-catch harvesting. Agronomy 13, 1618. http://dx.doi.org/10.3390/agronomy13061618.

Yin, W., Wen, H., Ning, Z., Ye, J., Dong, Z., Luo, L., 2021. Fruit detection and pose estimation for grape cluster–harvesting robot using binocular imagery based on deep neural networks. Front. Robot. AI 8, 626989.

Yu, Y., Zhang, K., Yang, L., Zhang, D., 2019. Fruit detection for strawberry harvesting robot in non-structural environment based on mask-RCNN. Comput. Electron. Agric. 163, 104846.

Zhou, X., Zou, X., Tang, W., Yan, Z., Meng, H., Luo, X., 2023. Unstructured road extraction and roadside fruit recognition in grape orchards based on a synchronous detection algorithm. Front. Plant Sci. 14, http://dx.doi.org/10.3389/fpls.2023.1103276.