Vision-based Vineyard Trunk Detection and its Integration into a Grapes Harvesting Robot

Eftichia Badeka, Theofanis Kalampokas, Eleni Vrochidou, Konstantinos Tziridis, George A. Papakostas, Theodore P. Pachidis, Vassilis G. Kaburlasos
Human-Machines Interaction Laboratory (HUMAIN-Lab), Department of Computer Science, International Hellenic University (IHU), Kavala, Greece
Email: {evbadek, theokala, evrochid, kenaaske, gpapak, pated, vgkabs}@cs.ihu.gr

***Abstract*—In this work, deep learning is employed for accurate and fast detection of vine trunks in vineyard images. More specifically, six well-known object detectors, Faster regions-convolutional neural network (Faster R-CNN), You Only Look Once version 3 (YOLOv3) and version 5 (YOLOv5), EfficientDet-D0, RetinaNet and MobilNet, are tested for real-time vine trunk detection. The models are trained with an in-house dataset designed for the needs of this study, containing 1927 manually annotated vine trunks in 899 different images. Comparative results indicate EfficientDet-D0 as the configuration that allows the faster and most accurate vine trunk detection, achieving Intersection over Union (IU) of 71% and overall Average Precision of 77.9% in 38 ms. The high precision combined with the fast runtime performance, indicate EfficientDet-D0 detector as the most suitable to be integrated into an autonomous harvesting robot for real-time vine trunk detection.**

***Index Terms*—object detection, harvesting robot, deep learning, trunk detection, computer vision, precision agriculture, Cyber-Physical System (CPS)**

I. Introduction

Wine industry has greatly developed in the last few decades [1]. In wine industry, eonologists seek to maximize the quality of the harvested grapes, while field managers try to minimize all operational costs. These two opposite objectives are met in the implementation of viticultural practices; on one hand, there are the annual canopy management practices aiming at maintaining and improving vineyards' health, leading to optimized wine quality, while on the other hand, there is the mechanization of these practices by agricultural robots, namely agrobots, aiming at reducing all labor costs [2].

Agrobots are capable of a longer duration of work, as an autonomous and automatic robot may outlast a human worker, increase productivity, application accuracy and operation safety [3]. In the aforementioned context, agrobots are adopted to perform a variety of vineyard management practices, including pruning, defoliation or green harvest [4]. Our interest here is in the development of an autonomous robot for grape harvest, namely ARG, able to support viticulture tasks such as harvest, cluster thinning (green harvest) and basel leaves removal (defoliation) [5]. ARG is designed as a Cyber-Physical system (CPS), integrating intelligence, communication and functionality, towards sensor awareness and decision making. In this context, ARG needs to navigate in the vineyard and to detect the vine trees so as to perform the selected viticulture tasks.

In vineyard, especially in those build in steep slope hills, there are several challenges regarding robots' navigation and localization, mainly due to terrain irregularities and inaccuracies of the signals emitted by the global navigation satellite system (GNSS), which is usually used for these purposes. Feature-based localization, i.e. extraction of reliable and persistent features or landmarks from vineyards, is therefore considered. Knowledge about the vineyards patterns is currently the most accurate, cheap and fast solution to facilitate agricultural tasks that need to be precise. Vine trunks can be selected as stable landmarks that exist in all vineyards. It makes sense to provide the robot with the ability to recognize vine trunks as high-level features of vineyards, to use in localization and mapping procedures. More analytically, detection of the vine trunks can help in building a precise vineyard map that the agricultural robot may rely on, to navigate safely and perform a wide range of agricultural tasks. Moreover, locating the vine trunk is the first step to automatically control the position and orientation of the robot in order to execute basel defoliation, and to center on the vine to perform harvest or green harvest evenly spaced on both sides. Therefore, vine trunks need to be located precisely for two main reasons: 1) to facilitate the navigation of ARG in the vineyard corridors and 2) to locate the working point of ARG regarding the performance of the selected viticulture tasks.

The problem of vine trunk detection is challenging due to the fact that during both basel defoliation and green harvest season, vineyard corridors and vine trunks are occluded by shoots and leaves [6], making it difficult to determine either the vineyard corridors or to discriminate the vine trunks (Fig. 1.).

Manuscript received August 10, 2020; revised January 12, 2021.

doi: 10.18178/ijmerr.10.7.374-385

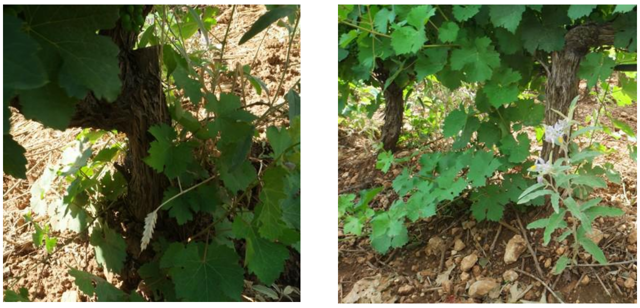


Figure 1. Vine trunk detection challenge due to foliage occlusions.

Toward this end, methods for reliable visual vine trunks detection are currently under investigation. Vine trunk detection is performed in [7] by adopting a new methodology, software and procedure of data acquisition, using 3D point clouds taken by laser scanning equipment. In [8], two laser sensors are used to detect vine trunks and provide their position and size measurements. A vision-based detector for natural feature detection is proposed in [9]. The proposed algorithm includes local binary pattern (LBP) image extraction and support vector machine (SVM) classification of extracted descriptors. In a later research [10], the same authors suggest LPB and hue, saturation and value (HSV) image extraction and SVM classification, exploring parallelization capabilities of pressing units in order to accelerate the processing time of the algorithm.

Deep learning-based techniques have demonstrated their ability to learn higher-level features and detect objects with higher accuracies than traditional machine vision systems [11].

Deep learning models [12] are widely used for object detection for agricultural-related tasks such as fruit detection [13], [14], leaves detection [13], plant disease detection [15], weed detection [16] and roots detection [17]. However, there are hardly no references in the literature for the use of deep-models in vine trunk detection. Deep learning has been used for vine trunks detection only recently, in [18], where pre-trained versions MobilNet V1, MobilNet V2 and Tiny YOLOv3 were examined. Results indicated MobilNet V2 as the most fast and accurate vine trunk detector, achieving an overall Average Precision of 52.98%. In their later work [19], the same authors used MobileNets, Inception, and lite version of YOLO to detect vine trunks in real-time. Results, once again, pointed out MobilNet V2 as the most effective model among them with the same 52.98% overall average precision. An other team of researchers also presented two versions of their work on vine trunk detection. The study in [6], presented a deep learning-based approach which first employs deep residual network (ResNet)-based Faster region-based convolutional neural network (Faster R-CNN) network to detect the visible segments of grapevine canopies. Then, position information of the detected visible segments of grapevine canopies are used, to estimate the cordon trajectories. Each canopy included two cordons coming from the trunk, growing along the trellis wires in to two opposite sides. Detecting visible parts of the trunk provided the reference point to differentiate right and left sides of the cordons. To eliminate overlapping bounding boxes and selecting only the strongest bounding boxes with high level of confidence, a non-maximal suppression (NMS) algorithm was used. The average precision of trunk detection for R-CNN was 26%, and improved to 76% with NMS. In their later research [2], the authors again tried to accurately determine the cordon shapes using deep learning networks. A color camera was used to acquire canopy images, and two different deep learning-based semantic segmentation techniques, segmentation network (SegNet) and fully convolutional network (FCN), were used for cordon detection and determination. Reported average pixel classification accuracy on trunk detection for both SegNet and FCN was about 90%. Three deep-learning models, Faster R-CNN, YOLOv3 and YOLOv5 are tested for real-time trunk detection in [20]. Results indicate YOLOv5 as the detector that outperforms the rest in terms of inference accuracy and runtime performance, achieving an overall Average Precision of 73.2% in 29.6 ms.

Based on the encouraging results of [20], this work comes as a continuation, adding to the investigation of the optimal vine-trunk detector for ARG, three more well-known deep learning models; EfficientDet-D0, RetinaNet and MobilNet. The models are trained in the same in-house designed dataset containing approximately 1927 annotated vine trunks in 899 different images. Comparative results are presented for all six models. Experimental results indicate EfficientDet-D0 as the configuration that allows the most accurate vine trunk detection, achieving an overall Average Precision of 77.9%. Going one step further, this work uses the detection results in combination with depth information acquired from a stereoscopic camera to determine the movement of ARG towards the detected vine trunk.

The aim of this work is to test well-known deep learning models that are employed for the first time to resolve the specific problem of vine trunk detection. The proposed approach is accurate and fast and it is considered suitable to be integrated to an autonomous harvesting robot to facilitate navigation and precise agricultural tasks implementation. The rest of the paper is structured as follows. In Section II the image dataset acquisition and annotation are described and the examined models are presented. Experimental results are discussed in Section III. Section IV describes the integration of the trunk detector into the ARG. Finally, Section V concludes and suggests directions for further research.

II. Materials and Methods

A. Image Acquisition

The images of the used dataset [21] are collected from three vineyards of North Greek wine producers as part of a national research program [5]. Images are acquired under natural daylight counting disturbances such as varying illumination and shadowing, so as to give diversity to the training procedure and robustness to the inference final result. The challenge in vine-trunk detection is that the trunks possess the same brown colour with the terrain, making them difficult to be discriminated. Moreover, in most of the cases, more than one trunk is depicted in one image, causing great occlusions. Image acquisition is in line with ARG's operation plan. The robot will navigate in the vineyard lines, searching for the closest vine trunk on its left-side using machine vision provided by the mounted camera on the robotic arm. Then, it will stop in front of the detected vine trunk and it will perform one of the selected viticulture tasks.

Thus, the capturing height matches the dimensions of the mobile harvesting robot. The capturing distance takes into account the average size and maximum opening angles of the mounted robotic arm. Finally, all images are captured by following the same protocol; from a distance between 30 cm to 100 cm from the crops line, and from a height of 50 cm to150 cm. The original dataset consists of 899 different images; 629 for training, 180 for validation and 90 for testing, keeping the same settings as in [20] for comparative reasons. The original dataset is augmented in order to generate sufficient number of images for training the CNN models. All images are resized to 416×416 before applied to the models.

In Fig. 2. are shown some representative images of the dataset, in order to point out the diversity and difficulties of the vine trunk detection task. The most common of them, are occlusions due to vegetation or leaves on the trunk, blur effects due to the movement of the camera, illumination and shadowing conditions.

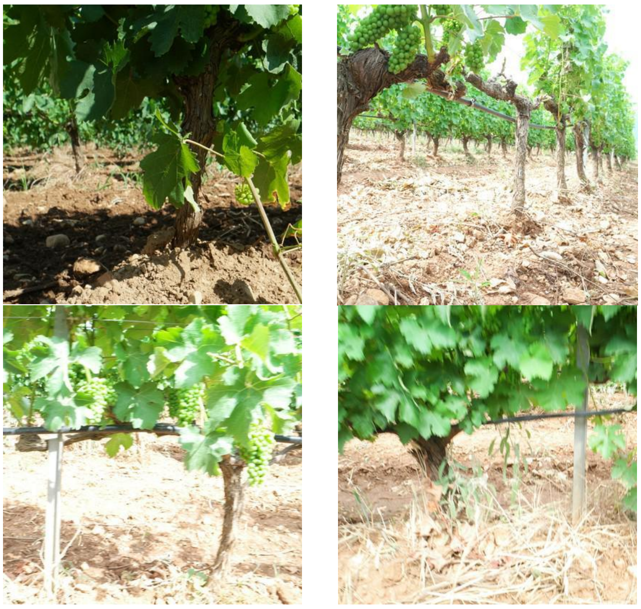


Figure 2. Representative images of the fnal dataset depicting difficult detection cases due to occlusions or varying illumination.

B. Image Annotation

All ground truth images are manually annotated using the Labellmg [22] online graphical image annotation tool. Fig.3. shows an example of a vineyard image with the respective annotations. The output of this process is a set of bounding boxes for each image. Bounding boxes are represented in a .txt file containing the label class considered and the four corners location of each bounding box. All annotated images used in this work are publicly available along with the training images [21].

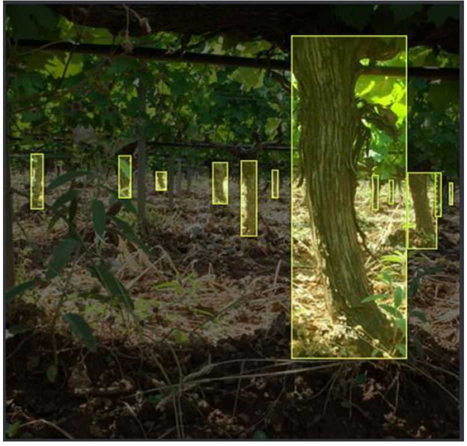


Figure 3. Image of the testing data with annotated trunks.

C. Examined Deep Learning Models

In total six object detection deep learning models are investigated in this work: Faster R-CNN, YOLOv3, YOLOv5, EfficientDet-D0, RetinaNet and MobilNet. The selected architectures use a feed-forward CNN that produces a set of bounding boxes and assigns a score for each one of them. The CNN contains convolutional feature layers to the end of the base network in order to detect objects of different sizes in images.

R-CNN models [23] can achieve high object detection accuracy by combining bottom up region proposals in order to localize and segment objects, and CNNs. Reported drawbacks included the fact that training is a multi-stage pipeline and, thus, it was time and space consuming, and additionally that object detection was slow since it performed a convolutional network forward pass for each object proposal, without sharing computation. Spatial pyramid pooling networks (SPPnets) [24] were proposed to speed up R-CNN by sharing computation. SPPnet accelerates R-CNN by 10 to 100 times at test time and by 3 at training time due to faster proposal feature extraction. Yet, training remained a multi-stage pipeline. A new training algorithm to overcome the disadvantages of R-CNN and SPPnet, was the Fast R-CNN [25]. More specifically, Fast R-CNN architecture provided higher detection quality, training as a single-stage process, could update all network layers and no disk storage was required for the features caching. However, both of the, R-CNN and Fast R-CNN, used selective search, a slow and time-consuming process, to find out the region proposals, affecting the performance of the network.

Faster R-CNN [26] was introduced as the updated version of Fast R-CNN, where a Region Proposal Network (RPN) was introduced, aiming to eliminate the selective search algorithm. An RPN is a fully-convolutional network that simultaneously predicts object bounds and objectness scores at each position. Faster R-CNN enables a unified, deep-learning-based, even faster object detection system. Additionally, the learned RPN improves region proposal quality and overall object detection accuracy.

The above sequel algorithms for object detection use regions to localize the object within the image. This means that the network does not look at the complete image, but parts of the image which have high probabilities of containing the object. YOLO [27] is much different from the region-based algorithms seen above. In YOLO a single convolutional network predicts the bounding boxes and the class probabilities for these boxes. It takes an image and it splits it into an S×S grid that contains m bounding boxes, assigned with a class probability and offset values. The bounding boxes that have the class probability above a threshold value are selected and used to locate the objects in the image. YOLO is demonstrated to be faster than other object detection algorithms. Its main disadvantage is that, it might face difficulties in detecting small objects.

The most popular and stable version of YOLO is YOLOv3 [28], [29]. In YOLOv3 the softmax function is replaced with logistic regression and threshold, and it displays a higher accuracy. The model associates the objectness score 1 to the bounding box anchor which overlays a ground truth object more than others. At the same time, it ignores others anchors that overlap the ground truth object by more than a chosen threshold. Thus, it allocates one bounding box anchor for each ground truth object. The use of prediction across scales using the concept of feature pyramid networks (FPNs), is considered as an additional improvement of the model. YOLOv3 is able to predict boxes at 3 different scales, from which it extracts features. The final outcome of the network is a 3-d tensor that includes bounding box, objectness score and prediction over classes. Moreover, YOLOv3 uses the CNN feature extractor Darknet-53, which is a 53 layered CNN that uses skip connections network encouraged from ResNet [30]. State-of-the-art accuracies have been reported for YOLOv3, with less floating-point operations and enhanced speed [31].

The latest version of YOLO, YOLOv5, outperforms all previous versions. YOLOv5 has been released recently, on the 9th of June 2020. The model's configuration is available [32] but no official research article is reported yet in the bibliography. YOLOv5, passes training data with every training batch through a data loader, which augments the data online. Three kinds of augmentations take place: scaling, color space adjustments, and mosaic augmentation. YOLOv5 allows for the reduction to half the floating-point precision in training and inference from 32 bits to 16 bits precision. This is able to significantly speed up the inference time of the metwork.

RetinaNet is a single, unified network composed of a backbone network and two task-specific subnetworks. It uses ResNet and FPN as the backbone networks [33]. The backbone is responsible for computing a convolutional feature map over an entire input image and is an off-the-self convolutional network. The first subnet performs convolutional object classification on the backbone's output, while the second subnet performs convolutional bounding box regression. RetinaNet proposes a new loss function that acts more effectively compared to previous approaches for dealing with class imbalance. The loss function is a dynamically scaled cross entropy loss, where the scaling factor decays to zero as the confidence in the correct class increases. Intuitively, this scaling factor can automatically down-weight the contribution of easy examples during training and rapidly focus the model on hard examples. Experiments demonstrated that focal Loss enables training a high-accuracy, one-stage detector that outperforms the alternatives of training with the sampling heuristics or hard example mining, the previous state-of-the-art techniques for training one-stage detectors.

As the name applied, MobileNets are designed to be used in mobile applications [34]. MobileNet uses depthwise separable convolutions. It significantly reduces the number of parameters when compared to a network with regular convolutions with the same depth in the nets. This results in lightweight deep neural networks. More specifically, two simple global hyperparameters that efficiently trade-off between latency and accuracy are introduced; width multiplier α and resolution multiplier ρ. These hyper-parameters allow the model builder to choose the right sized model for their application based on the constraints of the problem uniformly at each layer. More specifically, the first is used to reduce the size of the CNN and the second to reduce the computational cost. When MobileNets were applied to a wide variety of tasks and were compared with different popular models, they demonstrated superior size, speed and accuracy characteristics.

Only recently, two optimizations have been proposed; a weighted bi-directional feature pyramid network (BiFPN) which allows easy and fast multi-scale feature fusion, and a scaling method that uniformly scales the resolution, depth, and width for all backbone, feature network, and box/class prediction networks at the same time. Based on these optimizations and EfficientNet backbones [35], a new family of object detectors, namely EfficientDet, have been introduced, achieving significantly better accuracy and efficiency across a wide spectrum of resource constraints [36]. ImageNet-pretrained EfficientNets form the backbone network. The proposed BiFPN serves as the feature network, which takes level 3-7 features {P3, P4, P5, P6, P7} from the backbone network and repeatedly applies top-down and bottom-up bidirectional feature fusion. These features are fed to a class and box network to produce object class and bounding box predictions, respectively [36]. Here, a new compound scaling method for object detection is proposed. It uses a simple compound coefficient ϕ to jointly scale-up all dimensions of the backbone network, BiFPN network, class/box network, and resolution. Thus, for ϕ={1, 2, .., 7}, there are 7 backbone networks of width/depth scaling coefficients of EfficientNet, B0 to B6. In this work, D0 ($\phi=0$) is selected.

III. Experimental Study

A. Experimental Setup

The 629 images of the original data set are augmented for training Faster R-CNN, YOLOv3, EfficientDet-D0, RetinaNet and MobileNet. Three data augmentation techniques are used: rotation (between -22o and +22o), brightness (between -55% and +55%) and blur (up to 3.5 pixels). Thus, the final training dataset contains 2516 images (629 original and 1887 augmented). YOLOv5 augments the data online using also three kind of augmentation techniques.

All models conclude to the same number of training images. For the validation and testing of the models are used 180 and 90 images, respectively.

All models are implemented in Python 3.7 using TensorFlow. The GPU hardware used for training is provided by Google Colab [37]. For YOLOv3 and YOLOv5, the batch size is 16. Batch size for Faster R-CNN is 12, for EfficientDet-D0 and MobileNet is 16 and for RetinaNet is 8. All the models have been pretrained using the COCO dataset [38] and they are retrained with the designed dataset to detect vine trunks for 8000 steps.

The evaluation of the performance of the models to detect vine trunks, is established by calculating the intersection over union (IU) and the mean average precision (mAP). Since the implementation in this work refers to real-time vine trunk identification for a harvesting robot, the runtime performance of the models is also considered crucial and therefore it is evaluated.

B. *Experimental Results*

The experimental results are summarized in Table I. As it can be observed from Table I, all models perform well and the results obtained range from the same scale. EfficientDet-D0 reports the higher mAP value, reaching 77.9%, with Faster R-CNN competing with very little difference, ranging at 77.2%. The higher IU is 89.3% for YOLOv3, followed by YOLOv5 with 88.6%. Regarding the average inference time per image, the lower time is reported for YOLOv5 with 29.6 ms, followed by MobileNet with 36 ms.

However, YOLOv5, does not report the higher performance in either of the two metrics; mAP is 73.2% and IU is 88.6%. Yet, the aforementioned evaluation metrics for YOLOv5 are high and close to the higher reported performances of the rest of the models.

The IU metric is a method to quantify the percent overlap between the target and the prediction bounding box. However, IU cannot describe adequately the behaviour of the model's precision-recall curve. For this reason, mAP to effectively integrate the area under a precision-recall curve, is considered as a more representative metric of the model's performance. In other words, mAP expresses the detection accuracy, while IU expresses the localization accuracy. In fact, the Microsoft COCO challenge's [23] primary metric for the detection tasks evaluates the average precision score using IU thresholds ranging from 0.5 to 0.95. In our case, IU threshold is set to 0.5. Thus, the mAP is selected as our primary performance criterion.

TABLE I. IU, MAP AND RUNTIME PERFORMANCE FOR THE EXAMINED MODELS

Model	Evaluation Metrics		
	IU (%)	mAP (%)	Average Inference Time per Image (s)
Faster R-CNN	71.0	77.2	1.2519
YOLOv3	89.3	60.2	0.0804
YOLOv5	88.6	73.2	0.0296
EfficientDet-D0	71	77.9	0.038
RetinaNet	72	72.54	0.087
MobileNet	73	71.79	0.036

For this reason, EfficientDet-D0 may be considered as the most efficient model, since it detects the vine trunks more precisely, yet, it lags behind in locating them, compared to YOLOv3.

This works addresses the problem of vine trunks detection for robot localization and mapping. In this context, the requirement is primarily to detect vine trunks and then to locate them on images, in real-time. In this context, and in conjunction with the above requirement, the most appropriate model would be the one of high mAP and low runtime performance.

Thus, as it can be seen from Table I, the optimal mode in our case should be EfficientDet-D0, combining higher mAP among all models (77,9%) in 38 ms, which is a very satisfying performance for real-time applications.

Faster R-CNN also displays high mAP (77,2%). However, the average inference time per image is 1,2519 second, which means that, comparatively, Faster R-CNN is 32.94 times slower than EfficientDet-D0.

According to the above, the similar mAP performance as Faster R-CNN combined with the large difference in average inference time per image, emerges EfficientDet-D0 as the best detector for the problem under study. EfficientDet-D0 achieves mAP of 77.9% and IU of 71%. In terms of inference runtime performance, the average testing time per image for EfficientDet-D0 is 38 ms, which corresponds to 26.3157 frames per second. The model achieves high mAP in a very little time. This is very important for the selected real-time application, since the in-field detection of vine trunks needs to be fast and accurate.

Fig. 4-9. include vine trunk detection results of the testing set for all models. Additionally, Fig. 10. illustrates the results obtained for all models under the same testing image. The ground truth image is the one of Fig. 3.

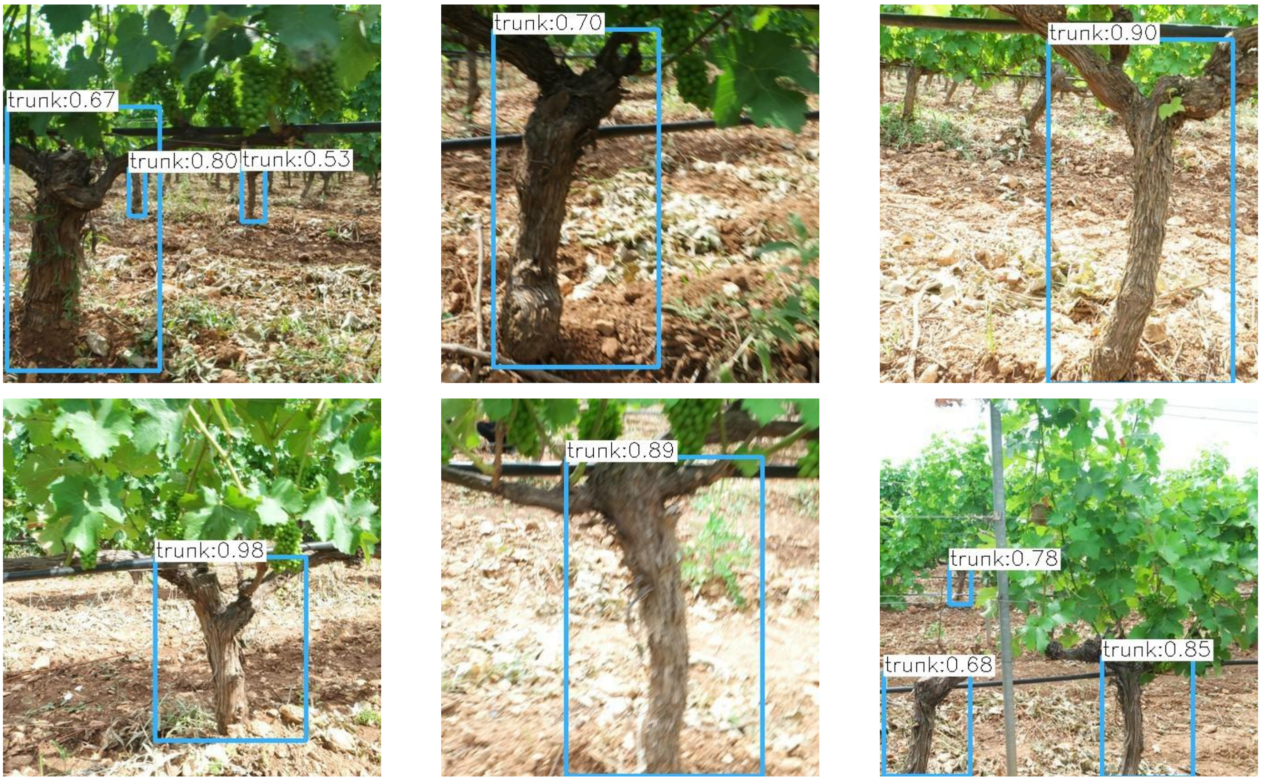


Figure 4. Detection results on the testing set with Faster R-CNN.

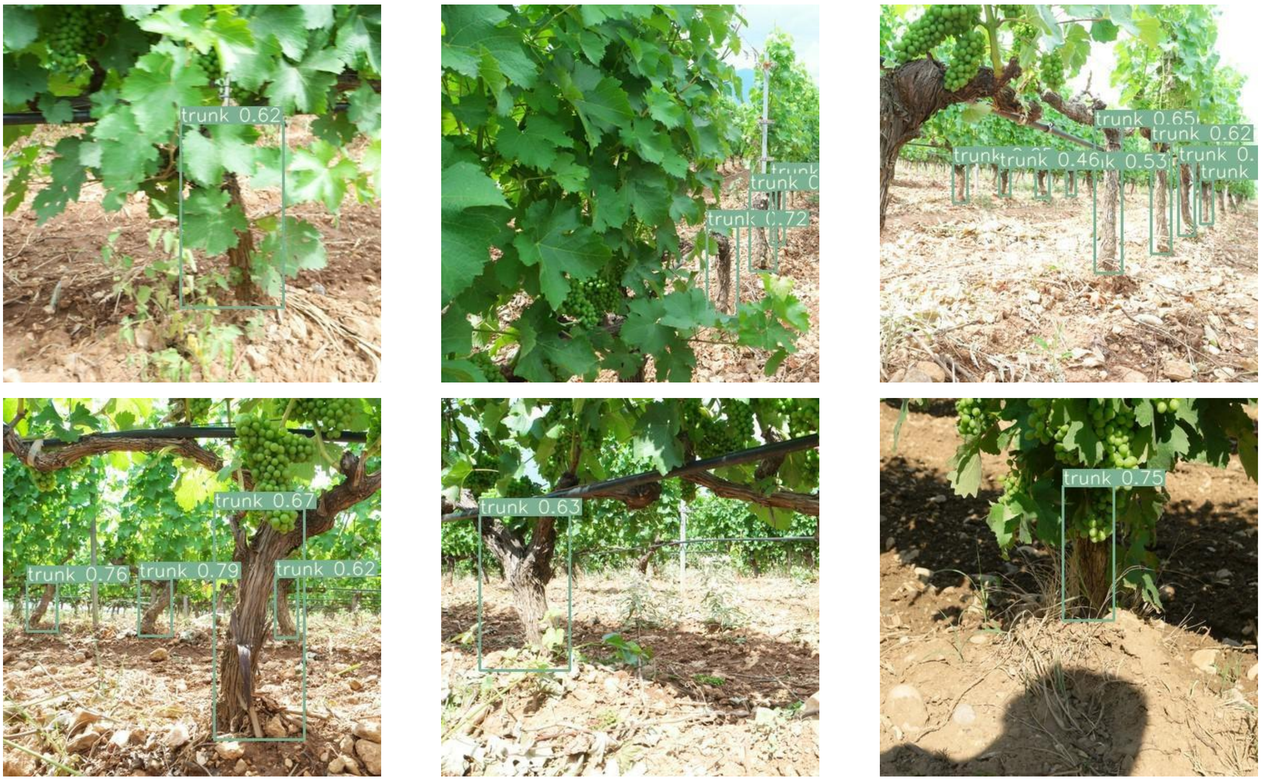


Figure 5. Detection results on the testing set with YOLOv3.

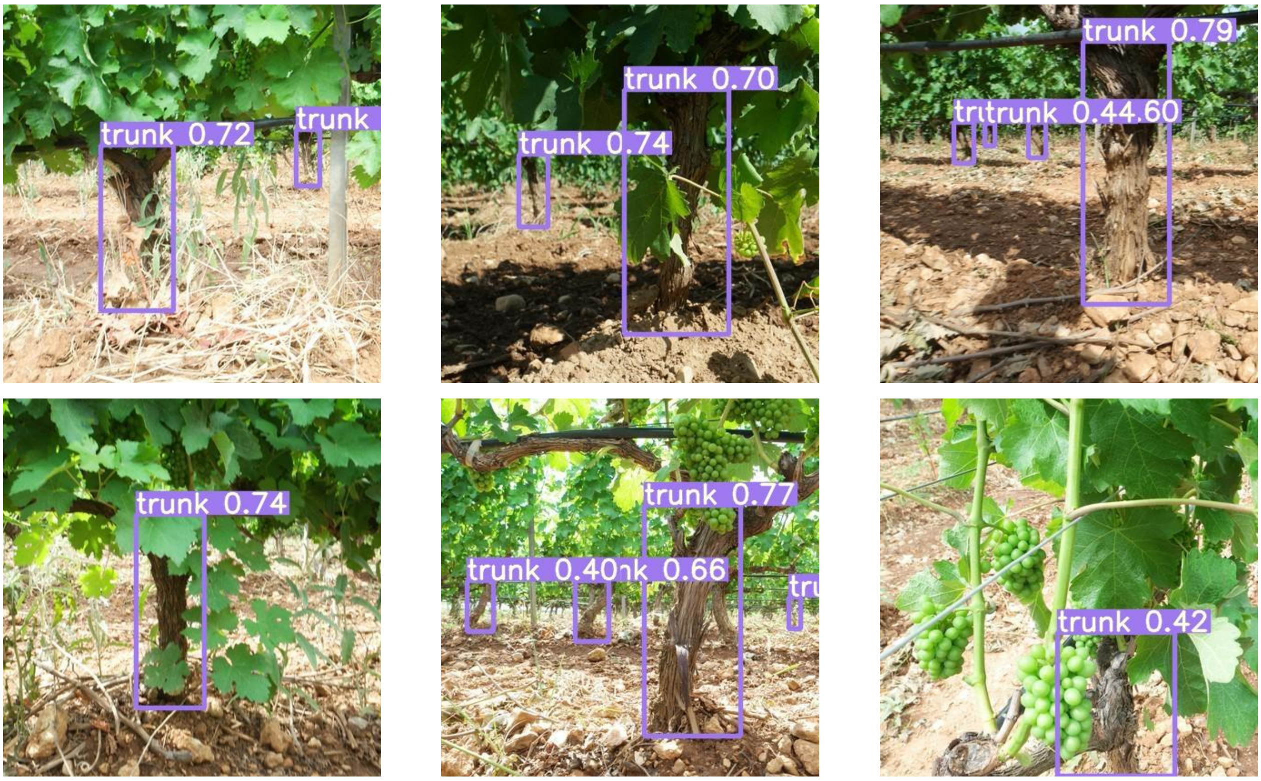


Figure 6. Detection results on the testing set with YOLOv5.

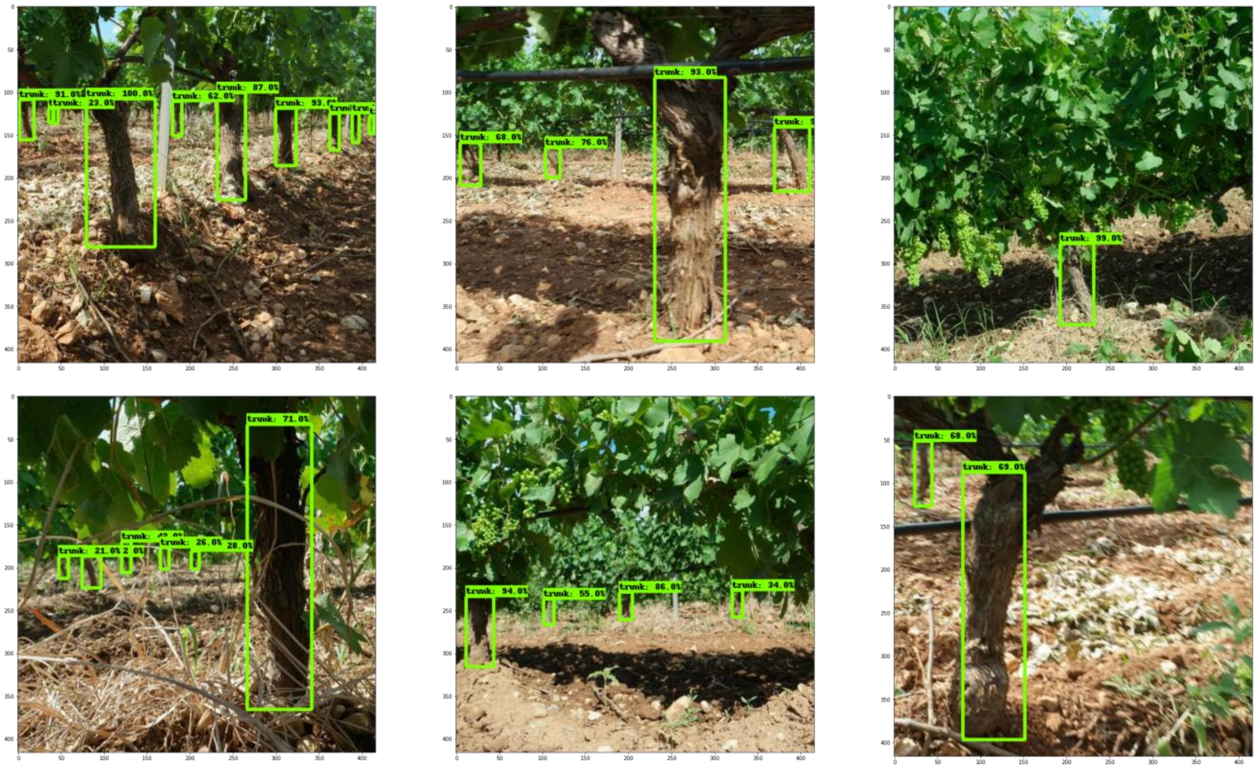


Figure 7. Detection results on the testing set with EfficientDet-D0.

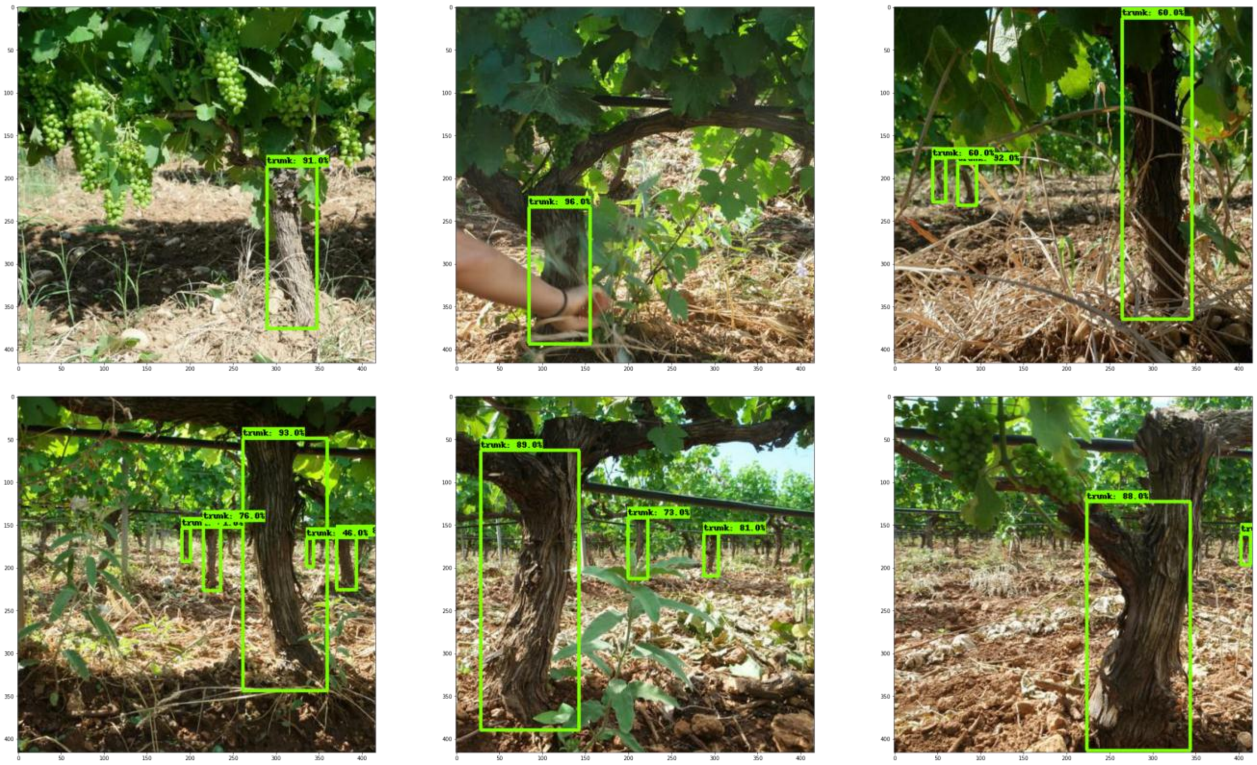


Figure 8. Detection results on the testing set with RetinaNet.

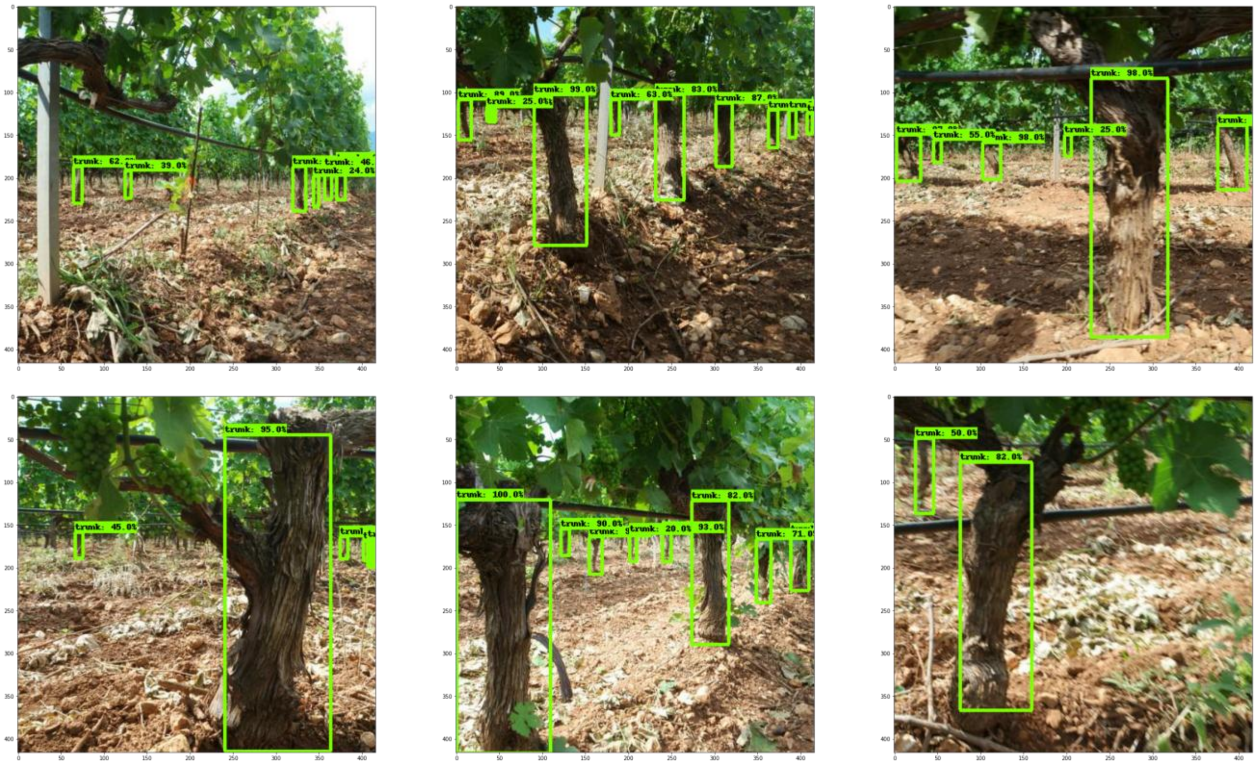
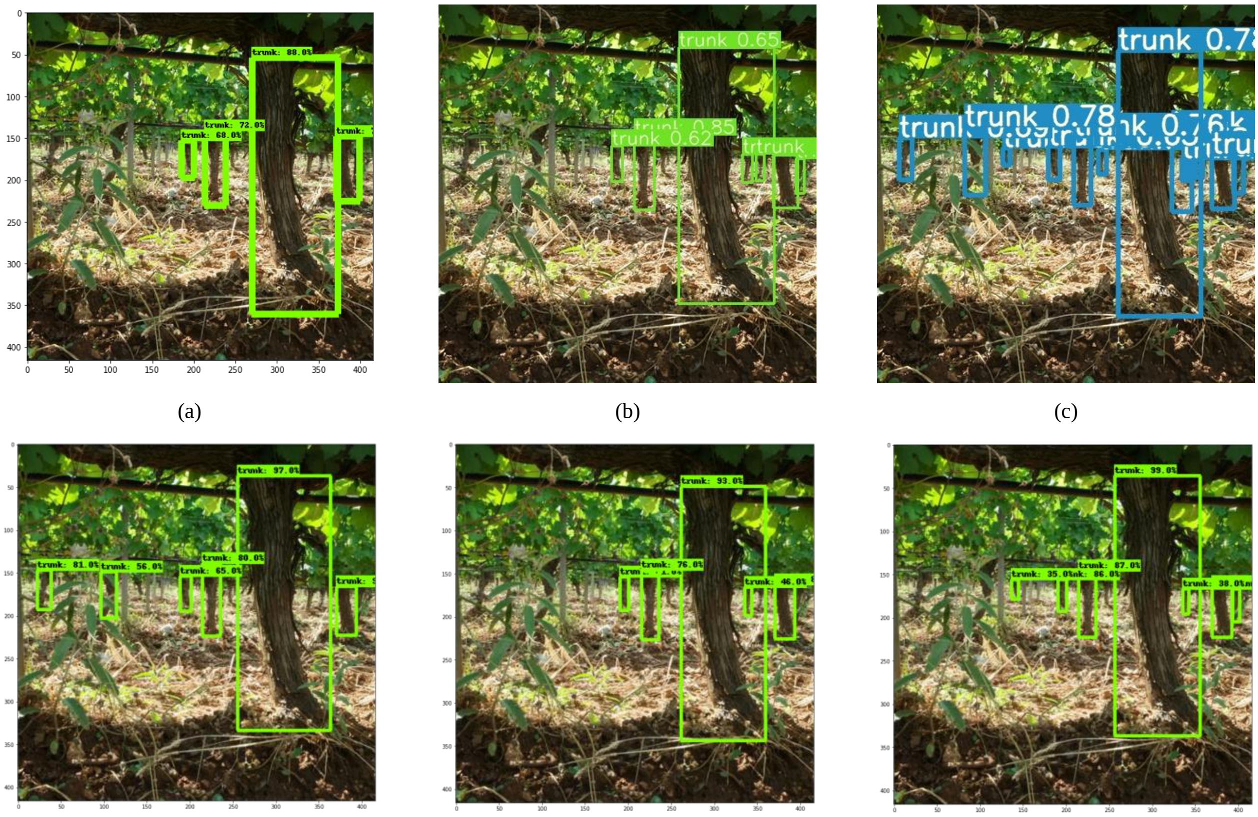


Figure 9. Detection results on the testing set with MobileNet.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 10. Comparative results of all models under the same testing image: (a) Faster R-CNN, (b) YOLOv3, (c)YOLOv5, (d) EfficientDet-D0, (e) RetinaNet and (f) MobileNet. Ground truth image is illustrated in Fig.3.

IV. Trunk Detector Integration

It is worth mentioning that in conjunction with the deep learning models, a depth map estimation algorithm based on stereoscopic vision, is also applied. The latter is to facilitate the robot to locate and navigate towards the closest vine trunk and perform the selected agricultural operation.

The optimal detector, EfficientDet-D0, performs at high frame rate, of approximately 26 frames per second (fps). For the main trunk detection, ARG will use an embedded Orbbec Astra stereo camera, mounted in the front of the wheeled robot, with limiting frame rate to 30 fps. However, when the detector will be integrated into the ARG, will run on a Jetson TX2 with 8GB memory and 59.7GB/s of memory bandwidth. The model architecture then, will be further optimized with TensorRT optimization. Moreover, using an asynchronous parallel execution, i.e. one thread for reading the data from the camera, one for running the inference and one for rendering the results, could speed even more the detection rate. Thus, the final optimized streaming trunk detector, is expected to achieve the maximum detection performance, as defined by the device’s hardware limits.

The detector will then be applied to the stereo camera system. Stereo-vision could benefit from hardware shaders and perform the disparity map calculation in real-time. The proposed detector can be used to extract vine trunks’ depth information using the disparity map. The detected vine trunks on the stereo images will be projected on the disparity map. The depth of each trunk could be calculated by computing the depth of the center of mass of each detected bounding box. This information will be fed to ARG, in order to construct a vineyard map, to localize itself and navigate safely towards the nearest, or any other, detected trunk.

More specifically, the detector integration is a two-step procedure. The first step of the process is illustrated in Fig.11.The front camera of the robot will provide live stream, creating the disparity map of each frame. The trunk detector model will be applied to the captured image frames. The robot is supposed to work only in one side of the vineyard, e.g. on the left side. Then, the closest trunk on the left side of the robot will be defined. The robot will move gently until the closest detected trunk will be located on the left edge of the acquired RGB image, at the closest possible detectable distance. If the robot moves closer to the trunk, following a parallel direction along the vineyard corridors, the trunk will no longer be on site.

The main purpose is for the robot to stand parallel to the vineyard and with the trunk perpendicular to it, centered, so that it can work accurately on it or symmetrically with respect to it. For this reason, the camera mounted on the robotic arm, sited on the left side of the vehicle, will provide lateral vision.

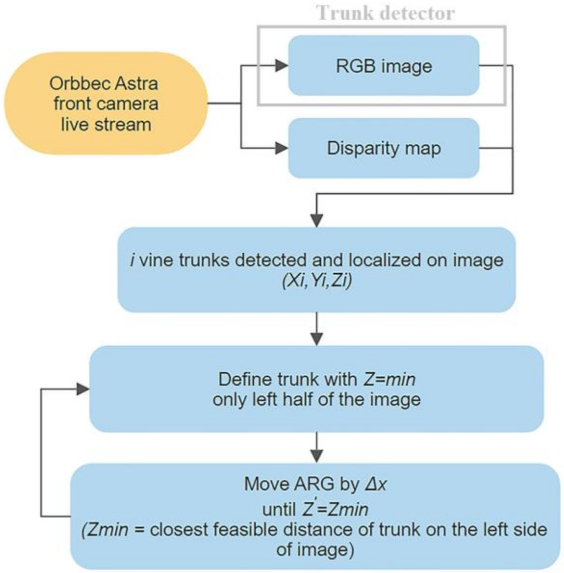

Trunk detector
Orbbec Astra front camera live stream
RGB image
Disparity map
i vine trunks detected and localized on image (*Xi,Yi,Zi*)
Define trunk with *Z=min* only left half of the image
Move ARG by *Δx* until *Z'=Zmin* (*Zmin* = closest feasible distance of trunk on the left side of image)

Figure 11. First step of detecor integration process.

The second step of the process regards the movement of the robot to the exact desired position in relation to the trunk, based on the live stream of a ZED mini side camera mounted on the robotic arm. The process is presented in Fig.12

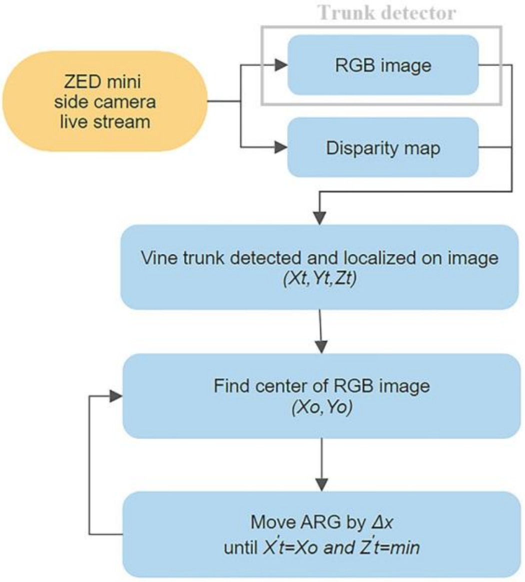

Trunk detector
ZED mini side camera live stream
RGB image
Disparity map
Vine trunk detected and localized on image (*Xt,Yt,Zt*)
Find center of RGB image (*Xo,Yo*)
Move ARG by *Δx* until *X't=Xo and Z't=min*

Figure 12. Second step of detecor integration process.

The side camera of the robot will provide live stream, creating the disparity map of each frame. The Trunk detector model will be applied to the frames. The closest detected trunk must then be centered in relation to ARG. The robot will move adequately until the trunk is finaly located on the center of the image and at the closest detected distance. Then ARG will start to perform the selected agricultutal task.

V. Conclusions

In this work, deep learning methods are investigated to accurately determine vine trunks, to facilitate navigation and operation plans of a grape harvesting robot. Six models are tested on an in-house vine trunks dataset owning detection difficulties, such as similarities in colour, occlusions and lighting variations. Moreover, the detector is combined with a depth map estimation algorithm based on stereoscopic vision, to guide the robot to the nearest vine trunk.

Experimental results point out EfficientDet-D0 as the most robust detector for the problem under study, able to detect vine trunks with mAP of 77.9% in 38 ms. Experimental results indicate that the proposed detector can be used to determine vine trunks in real-time in field operations even when the corridors are heavily occluded, to precisely position a harvesting robot and orient its end effector for automated viticultural tasks.

Future work includes increasing the size of the dataset with additional images, also considering thermal images, and consideration of other state-of-the-art CNN models towards investigating and adapting the optimal detector. Finally, future work includes the integration of the most adaptive and robust detector to ARG's sensing system. The objective is to develop a robot able to interact with the real world by means of computation, communication and controls. Integration of computational and physical units would be the final step for the development of a next generation computational CPS able to use intelligent methods associated with the physical world, towards mechanization of viticulture operations.

Author Contributions

GAP conceived and designed the work; EV analyzed the literature and wrote the paper. EB, TK and KT analyzed the data. GAP, TPP and VGK reviewed and corrected the paper; all authors had approved the final version.

Acknowledgment

This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project code: T1EDK-00300).

References

[1] M. Varas, F. Basso, S. Maturana, D. Osorio, and R. Pezoa, "A multi-objective approach for supporting wine grape harvest operations," *Comput. Ind. Eng.*, 2020.

[2] Y. Majeed, M. Karkee, Q. Zhang, L. Fu, and M. D. Whiting, "Determining grapevine cordon shape for automated green shoot thinning using semantic segmentation-based deep learning networks," *Comput. Electron. Agric.*, vol. 171, p. 105308, Apr. 2020.

[3] E. Mavridou, E. Vrochidou, G. A. Papakostas, T. Pachidis, and V. G. Kaburlasos, "Machine vision systems in precision agriculture for crop farming," *J. Imaging*, vol. 5, no. 12, p. 89, Dec. 2019.

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

[5] "Personalized Optimal Grape Harvest by Autonomous Robot (POGHAR)," *HUMAIN Lab*, 2018. [Online]. Available:

http://evtar.eu/. [Accessed: 16-Jul-2020].

[6] Y. Majeed, M. Karkee, and Q. Zhang, “Estimating the trajectories of vine cordons in full foliage canopies for automated green shoot thinning in vineyards,” *Comput. Electron. Agric.*, vol. 176, p. 105671, Sep. 2020.

[7] A. del-Campo-Sanchez, M. Moreno, R. Ballesteros, and D. Hernandez-Lopez, “Geometric characterization of vines from 3D point clouds obtained with laser scanner systems,” *Remote Sens.*, vol. 11, no. 20, p. 2365, Oct. 2019.

[8] M. Kise, F. J. Pierce, D. B. Walsh, and J. Chang, “laser sensor-based trunk detection system for targeted pest control in vineyards,” in *Proc. 2009 Reno, Nevada, June 21 - June 24, 2009*, 2009.

[9] J. Mendes, F. Neves dos Santos, N. Ferraz, P. Couto, and R. Morais, “Vine trunk detector for a reliable robot localization system,” in *Proc. 2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2016, pp. 1–6.

[10] F. Azevedo, P. Shinde, L. Santos, J. Mendes, F. N. Santos, and H. Mendonca, “Parallelization of a vine trunk detection algorithm for a real time robot localization system,” in *Proc. 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2019, pp. 1–6.

[11] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[12] S. Hu, Y. Zuo, L. Wang, and P. Liu, “A review about building hidden layer methods of deep learning,” *J. Adv. Inf. Technol.*, vol. 7, no. 1, pp. 13–22, 2016.

[13] T. Kalampokas *et al.*, “Semantic segmentation of vineyard images using convolutional neural networks,” in *Proc. 21st International Conference on Engineering Applications of Neural Networks (EANN 2020)*, 2020, pp. 292–303.

[14] A. Koirala, K. B. Walsh, Z. Wang, and C. McCarthy, “Deep learning for real-time fruit detection and orchard fruit load estimation: benchmarking of ‘MangoYOLO,’” *Precis. Agric.*, vol. 20, no. 6, pp. 1107–1135, Dec. 2019.

[15] Y. Zhang, C. Song, and D. Zhang, “Deep learning-based object detection improvement for tomato disease,” *IEEE Access*, vol. 8, pp. 56607–56614, 2020.

[16] M. Bah, A. Hafiane, and R. Canals, “Deep learning with unsupervised data labeling for weed detection in line crops in UAV images,” *Remote Sens.*, vol. 10, no. 11, p. 1690, Oct. 2018.

[17] A. G. Smith, J. Petersen, R. Selvan, and C. R. Rasmussen, “Segmentation of roots in soil with U-Net,” *Plant Methods*, vol. 16, no. 1, p. 13, Dec. 2020.

[18] A. S. Aguiar, F. N. Dos Santos, A. J. M. De Sousa, P. M. Oliveira, and L. C. Santos, “Visual trunk detection using transfer learning and a deep learning-based coprocessor,” *IEEE Access*, vol. 8, pp. 77308–77320, 2020.

[19] A. S. Pinto de Aguiar, F. B. Neves dos Santos, L. C. Feliz dos Santos, V. M. de Jesus Filipe, and A. J. Miranda de Sousa, “Vineyard trunk detection using deep learning – An experimental device benchmark,” *Comput. Electron. Agric.*, vol. 175, p. 105535, Aug. 2020.

[20] E. Badeka, T. Kalampokas, E. Vrochidou, K. Tziridis, G. A. Papakostas, T. Pachidis, V. G. Kaburlasos, “Real-time vineyard trunk detection for a grapes harvesting robot via deep learning,” in *Proc. 13th International Conference on Machine Vision (ICMV 2020)*, 2020.

[21] HUMAIN-Lab, “Humain-Lab-vine-trunk-dataset,” 2020. [Online]. Available: https://github.com/humain-lab/vine-trunk. [Accessed: 07-Nov-2020].

[22] Tzutalin, “LabelImg,” *Git code*, 2015. [Online]. Available: https://github.com/tzutalin/labelImg. [Accessed: 26-May-2020].

[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[25] R. Girshick, “Fast R-CNN,” in *Proc. 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[26] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[28] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” Apr. 2018.

[29] E. Badeka, E. Vrochidou, G. A. Papakostas, T. Pachidis, and V. G. Kaburlasos, “Harvest crate detection for grapes harvesting robot based on YOLOv3 model,” in *Proc. Fourth International Conference on Intelligent Computing in Data Sciences (IEEE ICDS 2020)*, 2020.

[30] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[31] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, “Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3,” in *Proc. 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, 2019, pp. 1–6.

[32] G. Jocher, “YOLOv5,” *Ultralytics*, 2020. [Online]. Available: https://github.com/ultralytics/yolov5. [Accessed: 06-Aug-2020].

[33] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[34] A. G. Howard *et al.*, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” *arXiv Prepr. arXiv1704.04861*, Apr. 2017.

[35] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *Proc. 36th Int. Conf. Mach. Learn. ICML 2019*, May 2019.

[36] M. Tan, R. Pang, and Q. V. Le, “EfficientDet: Scalable and efficient object detection,” in *Proc. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10778–10787.

[37] Google, “Google Colab.” [Online]. Available: https://colab.research.google.com/. [Accessed: 26-May-2020].

[38] T. Y. Lin *et al.*, “Microsoft COCO: Common Objects in Context,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, pp. 740–755.

Eftichia Badeka was born in Serres, Greece on January 4, 1986. In 2015 she graduated from the Technological Educational Institute of Central Macedonia in Serres and received the Diploma of Computer Engineering. In 2020 she received the Degree of MPhil Postgraduate Studies Program – “Advanced Technologies in Informatics and Computers” hosted by the Computer Science Department of the International Hellenic University. Since 2018, during her postgraduate studies, she is a member of the HUman-MAchines INteraction Laboratory (HUMAIN-Lab). Her main interests are in Image Processing, Computer Vision, Artificial Intelligence and Machine and Deep Learning techniques

Theofanis Kalampokas was born on September 5, 1994. In 2018 he graduated from the International Hellenic University, department of Computer Science in Kavala, receiving the qualification of software engineer. From 2018 until now, he is a postgraduate student in the Department of Computer Science at IHU, Kavala. As a researcher he is a member of the HUman-MAchines INteraction Laboratory (HUMAIN-Lab) where he participates in research projects.

Eleni Vrochidou received the Diploma [Embedded Systems], the M.Sc [Automatic Control Systems] and Ph.D. [Signal Processing] Degrees from the Department of Electrical & Computer Engineering, Democritus University of Thrace (DUTH), Greece, in 2004, 2007 and 2016, respectively. She is currently a part-time lecturer in the Department of Computer Science (IHU) at the International Hellenic University (former Eastern Macedonia and Thrace Institute of Technology (EMaTTech)) Greece, where she is teaching undergraduate courses (2005-2020) and Master courses (2018-2020). Her research interests are mainly in intelligent systems, signal processing, pattern recognition and embedded systems. In these areas, she has several publications in international scientific conferences, international scientific journal and book chapters. As a researcher she is a member of the HUman-MAchines INteraction Laboratory (HUMAIN-Lab) where she participates in research projects.

Konstantinos Tziridis was born on October 21, 1995. In 2018 he graduated from the International Hellenic University, department of Computer Science in Kavala, receiving the qualification of software engineer. From 2018 until now, he is a postgraduate student in the Department of Computer Science at IHU, Kavala. As a researcher he is a member of the HUman-MAchines INteraction Laboratory (HUMAIN-Lab) where he participates in research projects.

George A. Papakostas has received a diploma in Electrical and Computer Engineering in 1999 and the M.Sc. and Ph.D. degrees in Electrical and Computer Engineering in 2002 and 2007, respectively, from the Democritus University of Thrace (DUTH), Greece. Dr. Papakostas serves as a Tenured Full Professor in the Department of Computer Science, International Hellenic University, Greece. Dr. Papakostas has 10 years of experience in large-scale systems design, as a senior software engineer and technical manager and currently, he is the Head of the “Visual Computing” division of HUman-MAchines INteraction Laboratory (HUMAIN-Lab). He has (co)authored more than 140 publications in indexed journals, international conferences and book chapters, 1 book (in greek), 2 edited books and 5 journal special issues. His publications have more than 2000 citations with h-index 27 (GoogleScholar). His research interests include machine learning, computer/machine vision, pattern recognition, computational intelligence. Dr. Papakostas served as a reviewer in numerous journals and conferences and he is a member of the IAENG, MIR Labs, EUCogIII and the Technical Chamber of Greece (TEE).

Theodore P. Pachidis has a B.S. degree in physics, M.S. degree in electronics (A.U.TH), and Ph.D. degree in robotics and machine vision systems (ECE Dept, D.U.TH). Since 2017, he is an Associate Professor and Head of the Computer and Informatics Engineering Department, EMaTTech and then of the Department of Computer Science, International Hellenic University, Greece. He is also Head of the Robotics Division, HUman-MAchines INteraction Laboratory (HUMAIN-Lab). He has participated as a coordinator or as a member in a number of projects. He has published 46 totally refereed papers. He is a reviewer in many international journals and conferences. He has also written many other teaching notes and documents. During his 31 years career, he has taught numerous laboratory and theoretical courses. His research interests include software engineering, computer programming, robotic systems, robot behaviors and sensory-based control, cooperating robots, machine vision, image processing, control systems, analog and digital electronics, sensors and measurements. He is member of the Union of Greek Physicists. He is Senior Member of IEEE (Computer, Robotics and Automation Society) and Member of many Scientific Committees.

Vassilis G. Kaburlasos has received the Diploma degree from the National Technical University of Athens, Greece, in 1986, and the M.Sc. and Ph.D. degrees from the University of Nevada, Reno, NV, USA, in 1989 and 1992, respectively, all in electrical engineering. He has been participant or (principal) investigator in 30 research projects, funded either publicly or privately, in the USA and in the European Union. He has been a member of the technical/advisory committee or an invited speaker in numerous international conferences and a reviewer of 35 indexed (WoS) journals. He has (co)authored more than 185 scientific research articles in indexed journals, refereed conferences, edited volumes and books. His research interests include cyber-physical system modeling applications. Currently, there are more than 2700 citations to his published work corresponding to an h-index of 28 (Google Scholar). Dr. Kaburlasos serves as a Tenured Full Professor in the Department of Computer Science at the International Hellenic University (IHU), Greece. Since 2019 he also serves as an elected member of IHU’s Research Committee. He is founder and director since 2016 of the HUman-MAchines INteraction Laboratory (HUMAIN-Lab) (http://humain-lab.cs.ihu.gr/?lang=en) bringing in projects with total budget over 5.0 million EUR. Dr. Kaburlasos is a member of several professional, scientific, and honor societies around the world including the Sigma Xi, Phi Kappa Phi, Tau Beta Pi, Eta Kappa Nu, and the Technical Chamber of Greece.