Robust Visual Servoing for Precision Agriculture Tasks using Learning and De-rain Image Recovery via GANs*

Agelos Kratimenos[1], Athanasios Mastrogeorgiou[2], Konstantinos Machairas[2], Konstantinos Koutsoukis[2], *Student members, IEEE*, and Evangelos Papadopoulos[2] *Fellow, IEEE*

***Abstract*— Recently, robots started making their first steps towards real world applications in agriculture and more specifically, in vineyards. Among other challenges, recognizing clusters of grapes and performing visual servoing towards them is an important task. Although approaches such as deep learning have emerged that seem to simplify the problem, and databases for training data are publicly available, results are affected severely by weather conditions. In this paper, the detection robustness of grape clusters is investigated subject to rain conditions, with the use of two state-of-the-art models for object detection, Mask-RCNN and YOLOv3. It is shown that rain in an image markedly reduces the accuracy of the classifiers, indicating that a de-raining method is vital in classification and training detection methods with rainy images is not enough. Cycle-GANs are exploited to generate de-rained images from rainy samples. The method is validated in a lab experiment using a wheeled robotic platform and a low-cost on board computer. Mask-RCNN proves to be computationally intensive to run on board compared to YOLOv3. In this scope, we demonstrate a complete, robust under rainy weather, low-cost and expandable application for precision agriculture in which a robot identifies a cluster of grapes at a high frequency by running YOLOv3-tiny on board and approaches it at a predefined distance.**

I. INTRODUCTION

According to the Food and Agriculture organization of the United Nations, plant pests and diseases affect food crops, cause significant losses to farmers and threaten food security [1]. Automating the processes of collecting samples (e.g., capture images) from the field can help to monitor the field with precision and prevent diseases from spreading. Usually, such tasks involve the development of:

- an object detection algorithm to identify the points of interest as specified by the agronomist in charge, and
- a visual servoing controller for approaching and capturing images from the targets, e.g., plant parts and soil.

The design and development of object detection and visual servoing algorithms for outdoors applications are notoriously demanding tasks, since their performance and autonomy are affected significantly by weather conditions, e.g., rain, fog, direct sunlight, etc. [2]. To address such challenges, roboticists are adapting emerging technologies such as deep learning; despite their efforts only a few guidelines concerning these tasks are available [3].

Driven by the need for increased quantity and quality of agricultural products, we focus on fundamental research objectives that will allow bringing robots with inspection capabilities into the precision agriculture (PA) domain [4]. Related attempts have shown important results; at least 38 different crop diseases can be identified using machine learning, while especially regarding grapes, black measles, grape leaf blight and grape black rot can be detected [5], [6], [7]. Concerning fruit and yield detection in [8] and [9], per-pixel classification is performed using unsupervised multi-scale RGB and IR feature learning for fruit segmentation. Such methods require high resolution images, impossible to run in real-time. Approaches dealing exclusively with image recovery from rain or fog have been presented in [3], [10], [11], [12]. In these, the focus is on the quality of the output image and not on application constraints. As a result, either they have not been tested in high refresh rates, or require several seconds to de-rain a single frame. Visual servoing for monitoring row-crops was recently implemented in [13] using an on-board laptop combined with a single board computer (SBC) for data processing, while yield estimations in vineyards were performed in [14]. In both cases the weather conditions were ideal and the processing resources plenty. In [15] and [16], a "shroud" was used to block direct sunlight interfering with 3D object detection and weed detection, respectively. Finally, as an important addition to these vision-related advances, centimeter precision RTK-GNSS receivers have been made available to guide robots along accurate pre-programmed paths [17].

Although detection and navigation tasks have been significantly simplified through such works, complete robust solutions for vision driven navigation in the field still remain out of reach. Towards this goal, we consider robust object detection under various weather conditions as the key challenge. Thus, we focus on the problematic scenario in which the robot has successfully navigated in the area of interest, but cannot identify the target to be inspected due to weather conditions, e.g., fog, rain, raindrops on camera lenses etc. We choose the vineyard as the field of interest, and consider robust detection of grapes as the key component of a generic visual-servoing scheme allowing for a basic navigation task, i.e., approach a plant at a predefined distance. A scheme is developed (see Figure 1) that enables a camera-equipped robot to constantly search for grape clusters using an efficient

[1]Agelos Kratimenos is with the School of Electrical and Computer Engineering, National Technical Univerisity of Athens, Zografou, Greece ageloskrat@yahoo.gr

[2]The authors are with the School of Mechanical Engineering, National Technical University of Athens, Zografou, Greece. {amast, kmach, kkoutsou, egpapado}@central.ntua.gr

*This work was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "First Call for H.F.R.I. Research Projects to support Faculty members and Researchers and the procurement of high-cost research equipment grant" (Project Number: 2182).

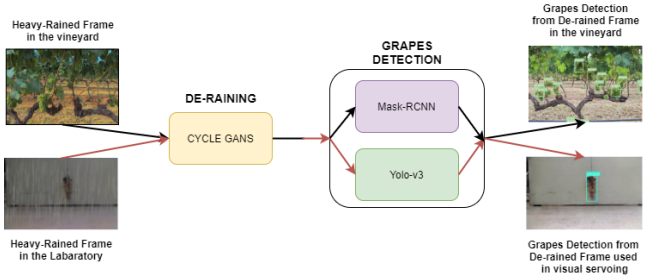


Fig. 1. The method proposed for robust grape cluster detection. The heavy-rained frame is de-rained with the use of Cycle GANs [25]. Detection is performed using Mask-RCNN [20] & YOLOv3 [24]. The red track is chosen for lab experiments since YOLOv3 is faster and the variation of YOLOv3-tiny is running at 10FPS on Jetson nano [27].

learnt strategy, and adjust its position with respect to a selected cluster. The scheme is tested successfully in the lab using a wheeled robot. The approach is expandable and applicable also to other types of crops. Data collection and annotation times are reduced significantly by applying augmentation methods to the training dataset. Systematic evaluation of state-of-the-art (SOTA) learning methods revealed the appropriate algorithm for the visual-servoing PA task. All software runs on the robot's low-cost SBC at a sufficiently high frequency, making the scheme applicable to realistic scenarios. Our contributions include:

- The presentation of a complete visual-servoing method for PA tasks, involving image recovery and fast yet robust object detection running on a low-cost SBC.
- A quantitative comparison of two SOTA learning methods when applied to grape recognition.
- The highlighting of image recovery importance in robust object detection via learning and the development of a solution based on filtering via generative adversarial networks (GANs) [18].

The rest of this paper is organized as follows: In Section II, two methods for recognizing grapes in a vineyard, Mask-RCNN and YOLOv3, are described and evaluated. Section III introduces the de-raining method used, while Section IV describes the lab experiment conducted. In Section V we present the experimental results and, in Section VI we conclude our remarks and propose plausible future directions.

II. GRAPE DETECTION METHODS

This work aims to make the procedure of detecting, identifying and following clusters of grapes, robust and independent of the weather conditions. In this section, the dataset in which the experiments were based on is presented, followed by an analysis of grape recognition methods. The scenario of grape detection was chosen for evaluation purposes since it constitutes a challenging and useful task in the field of robotics, due to changing weather conditions in vineyard and the need to monitor grapes for many months.

A. Dataset

For our experiments we exploit the Embrapa Wine Grape Instance Segmentation Dataset (WGISD) [19] which is publicly available. The dataset consists of 300 RGB images with a total of 4,432 grape clusters, already divided in a train and test set. Since no intervention was performed in the plants before the construction of the dataset, the WGISD depicts a realistic wine grape production scenario. To perform instance segmentation, a set of masks should be provided for the grape clusters. Apparently, only 2,202 out of 4,432 clusters, i.e 137 out of 300 images, contain annotated masks, while bounding boxes are provided for all the clusters.

B. Mask-RCNN Grapes Instance Segmentation

To identify grapes in a vineyard, two different methods are employed. First, the Mask-RCNN method is exploited [20]. It is a SOTA method for generating segmentation masks for each instance of an object in an image. Mask-RCNN combines a fully convolutional network (FCN) with a Faster R-CNN detector resulting in a complete, end-to-end, instance segmentation solution. For the experiments here, 137 images with the annotated masks were used. The network is initialized using the weights from the COCO dataset [21], while only the top layer is left unfrozen for training. The model is trained for a total of 30 epochs, while leaving the images to their initial shape.

To test the accuracy of the trained model, the mean average precision (mAP) metric for object detection was used. Since there is only one class (i.e.: grapes) then $mAP = AP$. Average precision is defined as the area under the precision-recall curve [22], i.e:

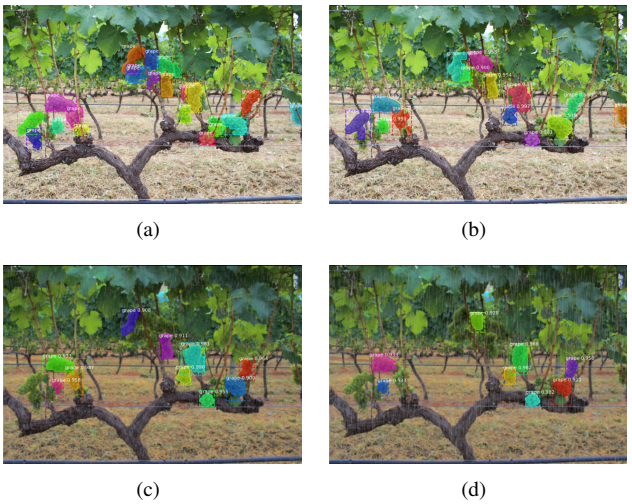
$$AP = \int_0^1 p(r)\,dr \tag{1}$$

Precision and recall are computed according to the Intersection over Union (IoU) metric between the predicted bounding box and the ground truth bounding box. A prediction is assumed to be correct, if IoU ≥ 0.5.

Testing the model in the test set yields an AP accuracy of 67.87%. To demonstrate the sensitivity of the method to weather conditions, the clean test images were altered with light and heavy rain, using the Automold library [23]. Consequently, two new test sets are created, i.e. a *light-rained* test set and a *heavy-rained* test set. Testing the same trained model in these datasets, yields an AP accuracy of 43.07% and 31.46% respectively, indicating a 36% absolute drop and over 50% relative drop compared to the clean (no-rain) test set. Qualitative results for the same single RGB image from all three test sets with the use of Mask-RCNN are shown in Figure 2.

C. YOLOv3 Grapes Object Detection

Mask-RCNN indeed produces a detailed and precise mask for each cluster of grapes in an image. Nonetheless, in this work the aim is to create a robust real-time robotic application, and Mask-RCNN falls far behind, achieving around 2 FPS with the use of a GeForce GTX 1660. In the case of the Jetson nano, Mask-RCNN needed several seconds to detect grapes in one frame. Hence, our attention was redirected to YOLOv3 [24], a SOTA, real-time object detection system. With YOLOv3, it is feasible to process images at 20 FPS with a GeForce GTX 1660, which is close

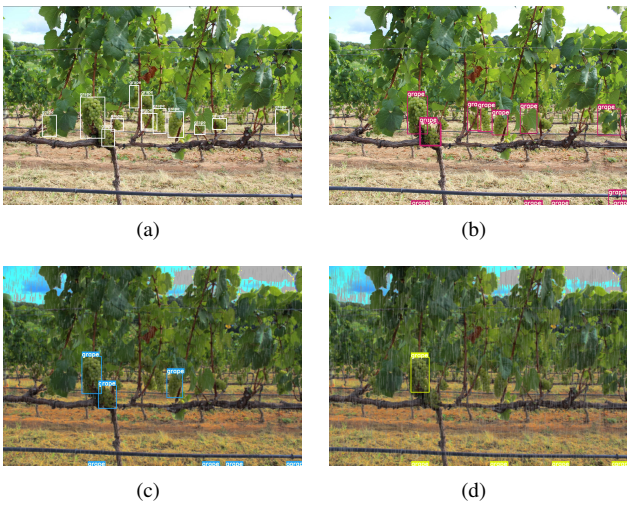


(a) (b) (c) (d)

Fig. 2. (a) Mask Ground Truth for a frame in a vineyard, (b) Prediction of the Mask-RCNN for the clean image, (c) Prediction of the Mask-RCNN model with light rain added, (d) Prediction of the Mask-RCNN with heavy rain added. Adding rain clearly reduces the AP accuracy of Mask-RCNN.

to the max frame rate that the used camera produces. For this experiment, a total of 300 images was used in the WGISD dataset. The model was trained for 300 epochs.

After the YOLOv3 model converged, it was tested in all three test sets. The average precision accuracy was used with IoU$\geq$0.5, the same as previously, although the accuracy was expected to be smaller compared to Mask-RCNN, due to the increase in the detection rate. The model yields an AP accuracy of 49.97%, 47.68% and 44.42% for the clean, light-rained and heavy-rain test sets, respectively, indicating once again a big drop between the clean test set and the rained test set. Detections are achieved at 20 FPS rendering YOLOv3 sufficient for an agriculture robotic application. Qualitative results for the same single RGB image from all three test sets with the use of YOLOv3 are shown in Figure 3.



(a) (b) (c) (d)

Fig. 3. (a) Bounding Box Ground Truth for a frame in a vineyard, (b) Prediction of the YOLOv3 for the clean image, (c) Prediction of the Yolo-v3 model with light rain added, (d) Prediction of the YOLOv3 with heavy rain added. Adding rain clearly reduces the AP accuracy of Yolo-v3.

D. Rain Augmentation

From both recognition methods, it is clear that weather conditions play a critical role in object detection and consequently in PA robotic applications. Hence, to construct a robust and stable method for following and identifying clusters of grapes, the instability due to adverse weather phenomena, e.g. rain, has to be eliminated.

To address this impediment, a first solution is to augment the train set with heavy rain using software and train a new model based on the new augmented train set. In this way, the model would be able to recognize clusters of grapes with ease since it is trained in a rainy environment. As a result, Mask-RCNN and YOLOv3 were both trained with the augmented dataset under the same clean training circumstances for direct comparison. For Mask-RCNN the new model achieves a 65.16% AP accuracy, indicating that it is indeed harder to train an augmented train set. On the other hand, its performance in the light-rained and heavy-rained test set increased by 14% and 20% respectively, reaching 57.19% and 51.46%. Similar results are observed in the YOLOv3 case. The full experimental results of Section II are displayed in Table I.

TABLE I

AP ACCURACY FOR THE CLEAN AND AUGMENTED TRAIN SET IN ALL THREE TEST SETS BOTH FOR MASK-RCNN AND YOLOV3 METHODS.

Training / Testing		Clean (%)	Light-Rained (%)	Heavy-Rained (%)
Mask-RCNN	Clean	**67.88**	43.07	31.46
	Augmented	**65.16**	57.19	51.11%
YOLOv3	Clean	**49.97**	47.68	44.42
	Augmented	**48.91**	48.41	47.11

III. DE-RAINING METHOD

While augmenting the training data with rainy images reduces the gap between the clean test set and the rained test sets, the need to create a more robust method remains. As a result, in this section, Cycle GANs [25] are applied for eliminating rain from images. In addition, they are applied for de-fogging purposes as well, to demonstrate the stability and robustness of the method.

Cycle-GANs [25] are a special category of GANs[18], specialized in converting one type of images to another, including zebras to horses, real scenery to Monet's paintings and vice versa. Technically, Cycle-GANs learn to translate images from a source domain X to a target domain Y, through a mapping G: X $\rightarrow$ Y. Apparently, this mapping is highly under-constrained and thus we interconnect it with the inverse mapping F: Y $\rightarrow$ X, introducing a cycle consistency loss to achieve F(G(X)) = X. One part of the cycle gan, namely the discriminator classifier, tries to distinguish real data from fake data created by the second part of the cycle-gan i.e.: the generator. In turn, the generator through the training process learns to produce better fake data by incorporating feedback from the discriminator.

Taking advantage of this technique, a generator was trained to convert rainy images to clear ones. The architecture proposed by [25] and implemented by [26] which

achieves impressive results was utilised. The discriminator network consists of 4 blocks of increasing filters, that contain one 2D convolutional layer, a 2D Instance Normalization layer and one LeakyRELU [30]. The generator consists of 2 similar blocks, followed by 7 residual blocks and a final convolutional layer that map the features to RGB. The identity loss $\mathscr{L}_{id}$ is defined to drive the mapping into preserving the input/output color composition. The cyclic loss $\mathscr{L}_{cyc}$ tries to achieve F(G(x)) = x and F(G(y)) = y. Finally $\mathscr{L}_{GAN} = \mathscr{L}_{GAN}(G, D_Y, X, Y)$ is the adversial loss that encourages G(x) to produce images similar to the domain Y, while D(y) aims to distinguish between real and generated images. The $\mathscr{L}_{GAN}(F, D_X, Y, X)$ takes care of the opposite direction. Conclusively, the following expression has to be minimised:

$$G^*, F^* = \arg\min_{G,F} \max_{D_x, D_y} \mathscr{L}(G, F, D_x, D_y) \tag{2}$$

where: $\mathscr{L}(G, F, D_x, D_y) = \mathscr{L}_{GAN}(G, D_y, X, Y) + \mathscr{L}_{GAN}(F, D_x, Y, X) + \lambda_1 \mathscr{L}_{cyc}(G, F) + \lambda_2 \mathscr{L}G, F)$

The discriminators and generators are trained for 200 epochs for their losses to converge and afterwards they are evaluated in the vineyard. Figure 4 shows qualitative results of a de-rained image produced by the Cycle-GAN generator. Finally, Mask-RCNN and YOLOv3 were used for grape cluster detection on the de-rained dataset and a comparison between no-rain predictions, heavy-rained predictions and de-rained predictions is shown in Table II. It is observed that both methods for grape clusters detection achieve a close-range AP accuracy to the initial test set, where the rain was absent. Moreover, they exceed the AP accuracy of the data augmentation method, indicating that de-raining an image is much more effective and robust than training the data with rain.

TABLE II

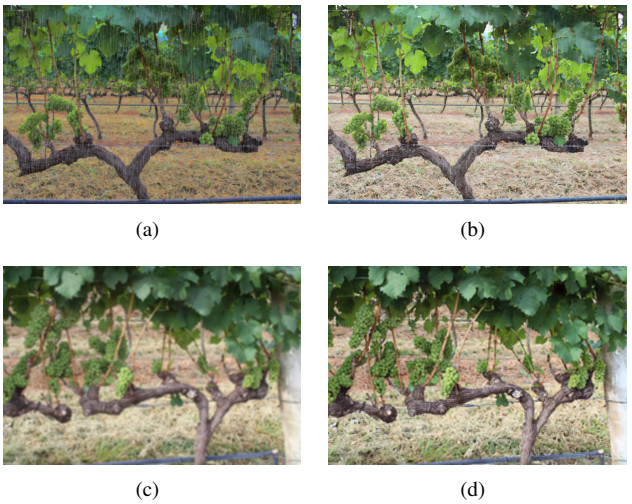
AP ACCURACY FOR MASK-RCNN AND YOLOV3 BEFORE AND AFTER DE-RAINING USING CYCLE-GANS.

Test Set / Method	Mask-RCNN	YOLOv3
Heavy-Rained Test Set	31.46%	44.42%
Data Augmentation	51.11%	47.11 %
CycleGAN De-rained Test Set	**61.21%**	**48.99** %
Initial Clean Test Set	67.88%	49.97%

To examine the robustness and universality of the de-raining method, we experimented with another common weather condition i.e.: fog and haze due to cold. The initial image set is augmented with fog and a new discriminator and generator were trained to convert a foggy image to a clear one. Figure 4 depicts the results after the convergence of the Cycle-GAN. It is once again clear, that Cycle-GANS can effectively transform an image with heavy weather conditions to a smoother one, independently of the condition, rendering them a perfect solution for a robust outside application.

IV. LAB EXPERIMENT

To combine all the studied elements into one integrated robust method for PA robotic applications, a lab experiment was setup showing how visual servoing is affected by poor object detection and the improvements using our approach. In this section, the whole method is sequentially presented.



(a) (b) (c) (d)

Fig. 4. (a) Vineyard image with the addition of synthetic heavy rain, (b) De-rained image produced by Cycle-GAN generator, (c) Vineyard image with the addition of synthetic fog, (d) De-fogged image produced by Cycle-GAN generator. The Cycle-GAN method removes equally well different types of weather conditions.

A. Rover Description

A wheeled robotic platform was developed to validate the concept (see figure 5). The robot was designed and constructed for research purposes, and thus it comprises custom built in-house parts as well as off-the-shelf parts (e.g. aluminum profiles, bearing units etc.). The motion system features four mecanum wheels [29] to provide to the robot omnidirectional motion capabilities. The wheels are powered by four Maxon DC motors (RE 35) combined with planetary gearboxes (GP 42) and incremental encoders (HEDL 5540), providing 5 Nm of continuous torque at their output. Timing belts and pulleys are used to transmit power to the wheels.

Two RoboteQ SDC2160 motor controllers are used to drive the actuators, since each controller can drive two actuators. The encoders attached to the motors are read by the controllers, which run local PID control schemes that can follow precisely speed commands for all wheels. The two motor controllers are attached to a RoboteQ's proprietary CAN-based meshed networking scheme, called RoboCAN, which enables data exchange between them. The first controller is connected via USB to the system's master computer (Jetson Nano) running Ubuntu 18.04 with JetPack 4.4.1. Through this serial connection, the master computer controls the platform motion system via simple motion control commands (see figure 6). The ROS option was dropped since YOLOv3 is computationally intensive and ROS would create an extra overhead for Jetson nano.

B. The Method

The robot detects the grapes from a distance of up to 1.5 m away. Next, it aligns with the detected grape and

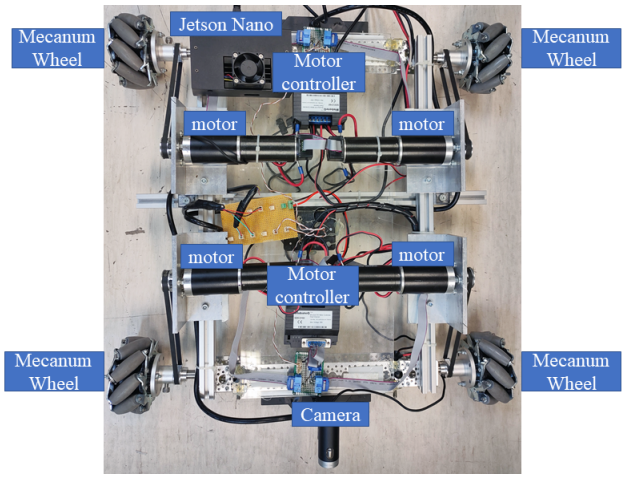


Fig. 5. The robotic platform developed for the visual servoing task.

approaches it until the grape is 20 cm away from the front camera allowing it to capture a clear image of it. The camera used is a Microsoft LifeCam Cinema with 720p resolution and auto-focus. Concerning the visual servoing, the velocity commands sent to the motors are inversely proportional to the area the grapes are covering on every camera frame. As the area of the bounding box is increasing, while the rover is approaching the grapes, it decelerates. The robot stops when the grape covers more than 70% of the total camera frame area which has been tested to be at a distance of 20cm from the grape. The inverse kinematics of the mecanum wheeled rover are described in (3).

$$\begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\\ \omega_4\end{bmatrix} = \frac{1}{R}\begin{bmatrix}1 & 1 & -(l_1+l_2)\\ 1 & -1 & (l_1+l_2)\\ 1 & -1 & -(l_1+l_2)\\ 1 & 1 & (l_1+l_2)\end{bmatrix}\begin{bmatrix}V_x\\ V_y\\ \omega_z\end{bmatrix} \tag{3}$$

where: $[\omega_1..\omega_4]$ are the speeds of the four mecanum wheels, R is the radius of the wheel, l_1, l_2 are the length and the width of the rover, last V_x, V_y, ω_z are the forward velocity, the lateral velocity and the yaw rate of the platform respectively. The rover is aligned with the detected grape by defining:

$$V_y = V_{y_{max}} \cdot (BB_{center} - F_{center}) \tag{4}$$

with BB_{center} the center of the bounding box of the detected grape and F_{center} the center of the camera frame. Since, the distance between the rover and the detected grape less than $1.5m$, more sophisticated control is not required and the current setup is working. Last, the rover approaches the grape by defining:

$$V_x = V_{x_{max}} \cdot (100\% - BB_{area}) \tag{5}$$

with BB_{area} the bounding box area normalised by the max frame size so that is does not change magnitude in case different camera models are used. If $BB_{area} > 70\%$ the rover is moving showily; then we set $V_x = 0$, since the grape is covering 70% of the frame. Using the inverse kinematics presented in (3) the angular velocities of the wheels $\omega_i, (i = 1..4)$ are calculated.

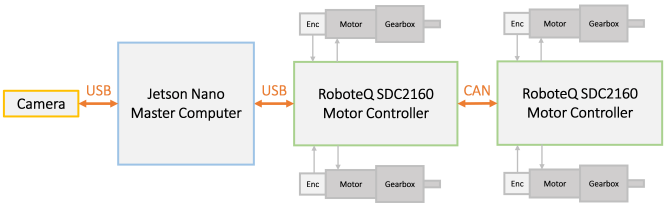


Fig. 6. Block diagram of the electronics, the motors and the motor controllers of the rover.

In a real life scenario, we would most plausibly want to use the robot to photograph the grape cluster and identify a disease, or in an advanced scenario, to grab the cluster. Hence, identifying and following one grape cluster, while smoothly decelerating before getting close enough, efficiently simulates a real-life requirement.

In practice, the camera attached to the rover will receive each frame, while YOLOv3 will perform object detection and return the bounding box corresponding to the unique grape cluster. The algorithm will calculate the area of the bounding box and correspondingly V_x, V_y as in (4) and (5). Our goal was to create a low-cost robust application that could be easily used in PA robotic applications. As a result, instead of using a high-end GPU for the entire processes, we switched to the $100 priced Jetson nano [27] and to YOLOv3-tiny [28], a lighter variation of YOLOv3. Using this setup, grape detection was achieved at 10 FPS on Jetson nano.

Next, the previous setup is repeated, while the camera frames are augmented with synthetic rain and again V_x, V_y are calculated using YOLOv3-tiny. The aim is to show, that neither the recognition of the grape cluster will be continuous and correct, nor the motion of the rover will be smooth, putting in danger, consequently, the robotic equipment and plausibly the cultivation as well.

Finally, synthetic rain is added to the camera frames and the de-rain method developed and discussed in Section IV is applied. The object detection and the motion control, then, follows as known. The third experiment is expected to approach the initial experiment as if there was no rain, not only confirming the theoretical results of the previous sections, but providing us with a robust PA robotic application that performs effectively under various weather conditions such as rain.

V. Results

The experiments are conducted for different values of the grape cluster detection confidence threshold (Figures 7 and 8). A typical application recognizes an object as part of the class when the confidence level is higher than 50%. Thus, experiments were carried out with 20% (bellow average), 50% and 70% (higher than average) confidence levels. In all three cases, in the absence of rain, the rover successfully approaches the grape cluster in a smooth way and ultimately decelerates when the desired distance is covered. The grape detection is clear when no rain is present, even when the confidence level is set to 70%. The robot smoothly approaches the grape, and a couple of frames are excluded due to the use

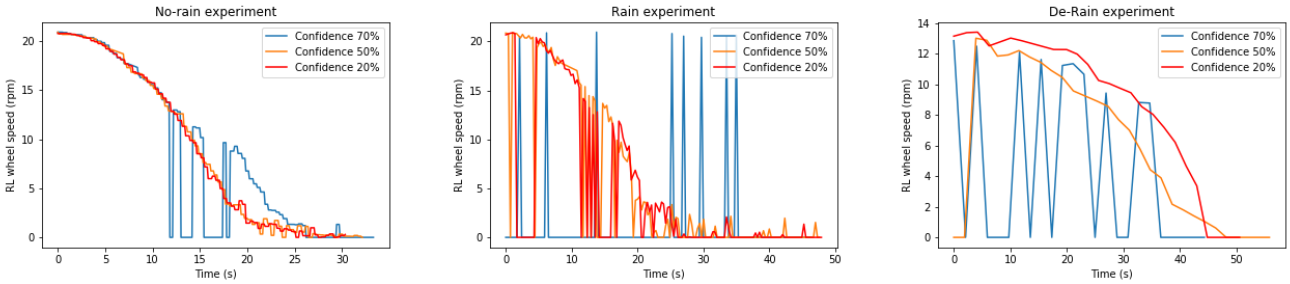


Fig. 7. Rear left (RL) wheel speed for different detection confidence thresholds for the duration of the (a) no-rain experiment, (b) rain experiment and (c) de-rained experiment. Zero velocity means that the algorithms does not recognise any grape in the captured frame.

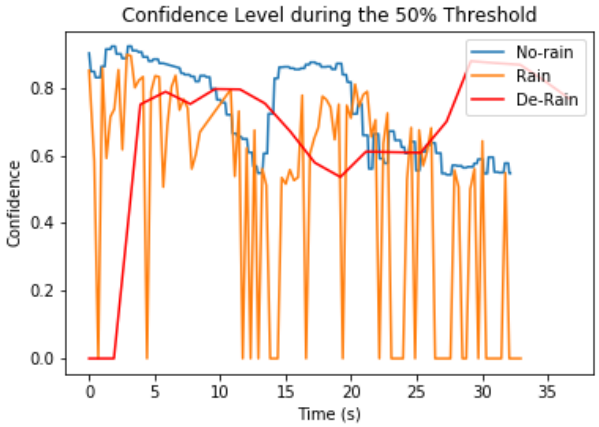


Fig. 8. The prediction confidence of the grape cluster for the duration of all three types of experiments. Using the de-rain method the detection confidence is always greater that 50% leading to smooth visual servoing.

of YOLOv3-tiny onboard, which is weaker than YOLOv3. Undoubtedly, for a typical 50% classification borderline, the motion is ideal for a real application.

On the other hand, when rain is added to the scenery, the task's difficulty heavily increases. When the confidence level is set up to 70%, the algorithm does not identify the grape cluster, except from a sparse number of frames where it vaguely moves, and fails to approach it. In the typical case of 50% confidence level, the rover reaches the grape cluster but in a completely bumpy motion, as it does not recognize the grape in a plethora of frames. When the confidence threshold is lowered to 20%, the rover reaches the grape cluster in a somehow more smooth way, but still with many standstills and discontinuity, rendering the application inefficient for real-life scenarios , as 20% confidence level is unrealistic. It is worth noting that the rover reaches the grapes at least 10 seconds later due to the on-off produced motion.

Finally, the de-rained method is applied. The velocity's smoothness is successfully restored for both 20% and 50% threshold of confidence. At the redundant confidence threshold of 70% the rover platform approaches and finally reaches the grape cluster in a non-smooth way indicating that the algorithm identifies the grape indeed, but not with high confidence, due to the imperfect de-rain recovery. It is worth mentioning, that the rover reaches the target 10 seconds later than the rain experiment, due to the delay caused by the prediction of the Cycle-GAN generator.

All the above experimental results are shown in Figure 7 where the real left (RL) wheel speed is plotted for the duration of each experiment, i.e no-rain, rain and de-rain, for the different levels of detection confidence. Figure 8 depicts the confidence level for each captured frame, for no-rain/rain/de-rain experiments for the case of the 50% threshold. It is clear that the detection confidence level for the no-rain experiment is always over 50% and no frame without detection exists. When the synthetic rain is added, the confidence output oscillates intensely, while in a lot of frames the grape cluster is undetected. Therefore, when de-rain is applied the confidence level is restored leading successfully to a robust method for a PA task.

VI. CONCLUSION AND FUTURE WORK

In this paper, we experimented with recognition of grape clusters in a vineyard by exploiting the state-of-the-art models Mask-RCNN and YOLOv3 using database images. After bringing out the sensitivity of object detection algorithms to weather conditions and specifically rain, we proceeded with applying de-raining methods to alleviate this phenomenon. Cycle GANs proved to be the most suitable and robust for our work, not only due to their ability to efficiently eliminate different weather circumstances but also because they can de-rain an image in real time. Combining the aforementioned experiments, we created a compact and robust robotic method for identifying and following grapes under severe weather circumstances in a real life scenario. At that stage, only YOLOv3-tiny was able to run in real-time on the low-cost SBC computer of the rover.

Future work includes further experiments to achieve an even higher FPS algorithm for de-raining by modifying cycle GANs. Moreover, being able to eliminate more than one weather condition from a single image, i.e fog and rain together, would significantly improve the speed and robustness of the application, while rendering it applicable in more scenarios. Finally, converting the proposed lab experiment into a setup of multiple grape clusters with unique identifiers for each one, so that the robot visits each one, would bring the application one step closer to real life concepts applied in precision agriculture.

REFERENCES

[1] http://www.fao.org/home/en/ (last accessed Oct. 30, 2020).

[2] Sudipta Mukhopadhyay; Abhishek Kumar Tripathi, Combating Bad Weather Part I: Rain Removal from Video , Morgan and Claypool, 2014.

[3] H. Porav, T. Bruls and P. Newman, "I Can See Clearly Now: Image Restoration via De-Raining," *2019 International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada, 2019, pp. 7087-7093.

[4] Mulla, D.. "Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps." *Biosystems Engineering 114* (2013): 358-371.

[5] Mohanty, Sharada P. et al. "Using Deep Learning for Image-Based Plant Disease Detection". *Frontiers in Plant Science 7*. (2016): 1419.

[6] U. Singh, A. Srivastava, D. Chauhan and A. Singh, "Computer Vision Technique for Detection of Grape Esca (Black Measles) Disease from Grape Leaf Samples," *2020 International Conference on Contemporary Computing and Applications (IC3A)*, Lucknow, India, 2020, pp. 110-115.

[7] S. M and S. V. Uma, "Adaptive machine learning approach for Grape Leaf Segmentation," *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, 2019, pp. 482-487.

[8] C. Hung, J. Nieto, Z. Taylor, J. Underwood and S. Sukkarieh, "Orchard fruit segmentation using multi-spectral feature learning," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, 2013, pp. 5314-5320.

[9] Hung C., Underwood J., Nieto J., Sukkarieh S. (2015) A Feature Learning Based Approach for Automated Fruit Yield Estimation. In: Mejias L., Corke P., Roberts J. (eds) Field and Service Robotics. *Springer Tracts in Advanced Robotics*, vol 105. Springer, Cham.

[10] S. Li, Y. Hou, H. Yue and Z. Guo, "Single Image De-Raining via Generative Adversarial Nets," *IEEE International Conference on Multimedia and Expo (ICME)*, Shanghai, China, 2019.

[11] R. Qian, R. T. Tan, W. Yang, J. Su, and J. Liu, "Attentive generative adversarial network for raindrop removal from a single image," *CoRR*, vol. abs/1711.10098, 2017.

[12] B. Shao, C. Lu and S. Huang, "Lightweight Image De-raining for IoT-enabled Cameras," *2019 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW)*, YILAN, Taiwan, 2019, pp. 1-2.

[13] A. Ahmadi, L. Nardi, N. Chebrolu and C. Stachniss, "Visual Servoing-based Navigation for Monitoring Row-Crop Fields," *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 4920-4926.

[14] G. Riggio, C. Fantuzzi and C. Secchi, "A Low-Cost Navigation Strategy for Yield Estimation in Vineyards," *IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, 2018, pp. 2200-2205, doi: 10.1109/ICRA.2018.8462839.

[15] K. Kusumam, T., S. Pearson, T. Duckett, and G. Cielniak. 3d-vision based detection, localization, and sizing of broccoli heads in the field. *Journal of Field Robotics*, 34(8):1505–1518, 2017.

[16] X. Wu, S. Aravecchia, and C. Pradalier. Design and implementation of computer vision based in-row weeding system. *In Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4218–4224, May 2019.

[17] Blok, P. M., Gookhwan, K., and van Boheemen, K. (2017). Assessing the navigational accuracy of an autonomous orchard robot equipped with 2D laser scanner and particle filter. *In 10th International Conference on Agriculture & Horticulture*. (Agrotechnology, an open access journal; Vol. 6, No. 4 (pp. 107-107)).

[18] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *in Proceedings of the 27th International Conference on Neural Information Processing Systems* - Volume 2, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 2672–2680.

[19] Thiago Santos, de Souza, Leonardo, dos Santos Andreza, Avila, Sandra. (2019). Embrapa Wine Grape Instance Segmentation Dataset – Embrapa WGISD (Version 1.0.0) [Data set]. Zenodo.

[20] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," *IEEE International Conference on Computer Vision (ICCV)*, Venice, 2017, pp. 2980-2988, doi: 10.1109/ICCV.2017.322.

[21] Lin Tsung-Yi, Maire Michael, Belongie Serge, Hays James, Perona Pietro, Ramanan Deva, Dollár Piotr, Zitnick, C. Lawrence, "Microsoft COCO: Common Objects in Context", *Computer Vision – ECCV 2014*, Springer International Publishing, pp. 740–755.

[22] Peter Flach, Meelis Hull, "Precision-Recall-Gain Curves: PR Analysis Done Right", NIPS 2015, Curran Associates, Inc., pp. 838–846

[23] https://github.com/UjjwalSaxena/Automold–Road-Augmentation-Library (last accessed Oct. 30, 2020).

[24] Redmon, Joseph et al. "YOLOv3: An Incremental Improvement". *arXiv*. (2018).

[25] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[26] Varuna Jayasiri, Nipun Wijerathne, "LabML: A library to organize machine learning experiments", 2020, https://lab-ml.com/

[27] https://developer.nvidia.com/embedded/jetson-nano-developer-kit (last accessed Oct. 30, 2020).

[28] P. Adarsh, P. Rathi and M. Kumar, "YOLO v3-Tiny: Object Detection and Recognition using one stage improved model," *6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, 2020, pp. 687-694, doi: 10.1109/ICACCS48705.2020.9074315.

[29] https://www.andymark.com/products/4-in-hd-mecanum-wheel-set-options (last accessed Oct. 30, 2020).

[30] Bing Xu and Naiyan Wang and Tianqi Chen and Mu Li, "Empirical Evaluation of Rectified Activations in Convolutional Network", *arXiv* 2015.