Image-based tree detection for autonomous navigation in orchards

Viviane Cadenat, Adrien Durand-Petiteville, Donatien Billot, Antoine Villemazet

Image-based tree detection for autonomous navigation in orchards

V. Cadenat
Univ. de Toulouse, CNRS, UPS, Toulouse, France
cadenat@laas.fr

A. Durand-Petiteville
Dpt. de Engenharia de Mecânica, UFPE
Recife, PE, Brazil
adrien.durandpetiteville@ufpe.br

D. Billot
Univ. de Toulouse, UPS, Toulouse, France
F-31400 Toulouse, France
donatien.billot@univ-tlse3.fr

A. Villemazet
CNRS, LAAS, 7 avenue du colonel Roche
F-31400 Toulouse, France
avillemaze@laas.fr

***Abstract*—This paper deals with the autonomous navigation problem through orchards. It highlights the different challenges arising in this particular environment and proposes an adequate framework. This latter relies on a reactive control strategy fed by visual data provided by a set of RGB-D cameras adequately positioned to enlarge the field of view. In this work, we show interest in coupling a deep learning solution for image-based detection with an existing point cloud processing algorithm to improve the overall perception. Experimental results validate the approach.**

***Index Terms*—Agricultural robotics, autonomous navigation.**

I. Introduction

Agriculture in the twenty-first century has to face two main challenges. The first one is related to the world population increase [1], which in turn demands to augment production. However, this augmentation must be done while taking into account economic and environmental constraints [2]. The second challenge concerns farm labor shortages which occur more and more often and which were highlighted by the CoViD'19 pandemic. They mainly come from socioeconomic, structural, and political factors, which make agricultural work unattractive. Agricultural robotics technologies can tackle these two challenges. Indeed, on the one hand, they provide the necessary tools for allowing precision farming and phytotechnology (selective plant care). It may then be possible to maximize production while taking care of the environment [3]. On the other hand, agricultural robots increase worker's efficiency and safety, by helping them to fulfill their tasks or even replacing them when it comes to tiring or laborious jobs.

This work is part of a project seeking to use robotics as a tool to develop sustainable agriculture where production increase matches environmental and societal concerns. It aims at developing a framework allowing a mobile robot to autonomously navigate in commercial orchards. Indeed, a safe motion through these environments is a prerequisite to the realization of any treatment or operation in the orchard, whether it relies on navigation skills only (payload or fruit transportation, surveillance, ...) or it includes manipulation skills (harvesting, pruning, weeding, ...). Thus providing a safe and efficient navigation strategy is a keystone towards efficient precision horticulture.

This work is funded by the *Agence Nationale de la Recherche* (ANR-20-CE33-0011-01) and the *Fundação de Amparo a Ciência e Tecnologia do Estado de Pernambuco* (FACEPE APQ-0139-3.04/20)

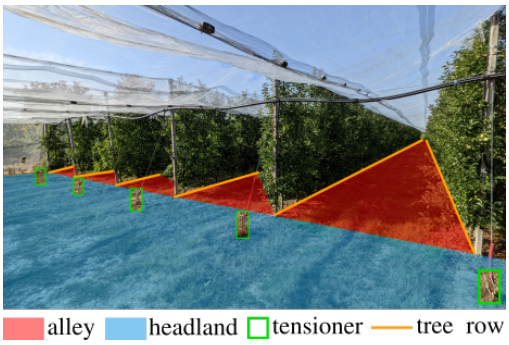


Fig. 1. The CEFEL orchard where we performed the experiment.

If autonomous navigation has been widely studied in robotics [4], its application to orchards remains challenging for two reasons. First of all, because of the canopy or the use of nets to protect the trees, the GNSS signal cannot be used to localize the robot, as it is classically done in other outdoor agricultural environments such as open fields [5], [6]. The control strategy has then to rely on local information for both traversing the tree alleys and maneuvering in the headlands (see Fig. 1). Second, the orchard is a natural environment, which means that it is subject to large visual variations. Indeed, its aspect is different according to not only the seasons and the treatments on the trees (pruning, thinning, harvesting, etc.) but also to the weather and the daytime. These issues thus require a particularly efficient and robust perception process in order to determine consistent and accurate navigation landmarks.

In this paper, we present the framework designed to perform autonomous navigation in orchards. We first recall some works related to this problem and introduce the main principles of the chosen solutions. We then describe the robot and its sensors, before detailing the control strategy. Finally, we focus on the 2-D and 3-D visual data-based perception systems, which is the main paper contribution. To conclude, we show experimental results highlighting the relevancy of the approach.

II. Related works and envisioned solution

We first present some of the works dealing with the specific challenge of navigating in orchards, *i.e.,* outdoor navigation without GNSS relying on embedded exteroceptive sensors. Most of the proposed solutions only address the alley crossing problem. They mainly consist in detecting the tree rows, computing geometric lines in the robot coordinate frame, and using them for guidance. The literature proposes a wide variety of approaches that can be divided according to the space used to compute the lines. The first class of works estimates the vehicle heading with respect to the tree lines in the image space. In [7], the vehicle heading is calculated by extracting the portion of the sky visible by the camera. In [8], images are segmented into classes such as terrain, trees, and sky, before applying a Hough transform to extract the features required to define the desired central path for the robot. In [9], a machine vision is designed to determine the path to follow. The second class of works uses Euclidean space, 2-D or 3-D, to calculate the tree lines. When using a 2-D Lidar, they can be calculated using a Hough transform [10], line regression and filtering [11]–[13], or line-based simultaneous localization and mapping (SLAM) [14]. More recently, in [15], the tree rows are computed by extracting lines, segments, and circles. Regarding 3-D Lidar sensors, [16] and [17] propose to calculate the 3-D coordinates of points belonging to the tree trunks, and to estimate the tree lines parameters using an extended Kalman filter. Finally, in [18], a time of flight camera is used to compute an occupancy grid of the alley.

However, the solutions mentioned above suffer from two main problems. First, they only work in one sole space, either in the image or in the 2-D or 3-D Euclidean space. Yet, images are highly sensitive to illumination variations when the detection method is not AI-based and moreover do not provide Euclidean coordinates. If 3-D approaches appear to be less sensitive to light but appear to be perturbed by uneven trunk branches distribution (tall vegetation, branches partially or totally occluding trunks, ...). In this context, developing solutions able to work both in the image and 3-D Euclidean space would be highly interesting. The second drawback of previous approaches lies in the computed visual feature for navigation. All of them aim to determine a straight line. Such a feature will not be suitable to follow curved alleys (as in circular orchards for example), or even perform a U-turn in the headland. This latter problem is even more difficult because in this zone it is impossible to detect tree lines. This is the reason why this problem is generally dealt with through dead reckoning, which in turn drastically degrades the task robustness [3]. To overcome this problem, it would be relevant to design control laws fed using exteroceptive data, as done in [19] for instance.

This analysis clearly shows the interest in building a reactive navigation strategy relying on exteroceptive sensors only. To do so, we have chosen to endow the robot with several RGB-D cameras to benefit from the two spaces (image and 3-D Euclidean space). These cameras have been positioned to enlarge the field of view so that landmarks are visible whether the robot is in the row or the headland. The path to be followed is computed at each instant using these landmarks and avoids the use of artificial landmarks. An enhanced control strategy based on NMPC (Nonlinear Model Predictive Control) allows following the reference path while taking into account constraints such as actuator saturation, field of view limitations, obstacle collisions, etc. In the sequel, we first present the robotic system before detailing the control strategy and focusing on the designed perception solution.

III. The robotic system

The considered robot is the Hunter 2.0 robot by Agilex (see Fig. 2(a)). It has been chosen for three main reasons. First, it is a car-like robot, which appears to be a very interesting mechanical structure to navigate in the orchard, because it eases the headland maneuvering and avoids damaging soils. Second, it can carry up to a 150kg payload, which makes it versatile in terms of equipment. Third, it is also able to climb small slopes (less than 10 degrees) and small obstacles (less than 5cm), which is an interesting feature in an orchard. As a car-like robot, it has two control inputs, the steering angle, and the linear velocity. Its minimum turning distance is 1.6m, while its maximum velocity is fixed at 6 km/h, which is suitable for our purpose.

As shown in Fig. 2(a), the robot has also been endowed with two exteroceptive sensors, a laser-rangefinder (Slamtech RPLIDAR S1), and four RGB-D Intel Real Sense cameras (two D455 and two D435). These sensors offer a low-cost solution to sense both 3-D point clouds and 2-D images in an outdoor environment. The two D455 cameras have been positioned at the front of the robot, while the D435 have been fixed on its left and right sides, because of the higher range of the former with respect to the latter (0.6 – 6m versus 0.3 – 3m). In this way, the view field is enlarged and it becomes possible to perceive the surrounding trees, whether the robot is following a row or maneuvering in the headland. This will allow feeding the U-turn control law with exteroceptive information, thus avoiding the use of dead reckoning and improving the task execution robustness.

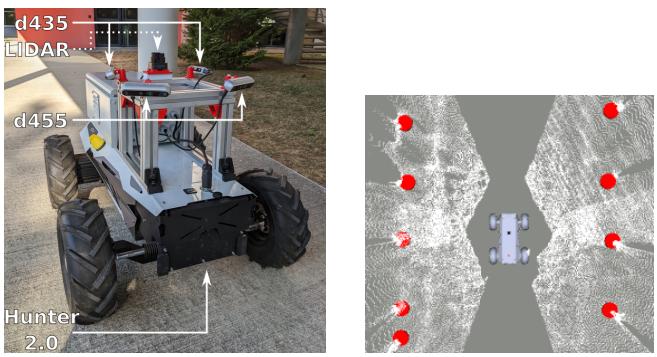


Fig. 2. The robot (left) and the point clouds expressed in the laser frame (right).

The perception system has been calibrated using the technique proposed in [20]. This latter has been proven to be accurate enough for the targeted application. An example of a result of this calibration is shown in Fig. 2(b) where the point clouds provided by the four cameras are expressed in the laser frame to benefit from a common reference.

IV. The control strategy

We have chosen to design a reactive vision-based control strategy. It relies on the 3-D coordinates of the tree trunks which will be computed using our perception algorithm (see Sec. V). In this part, we focus on the definition and the control of the robot's motion.

A. Definition of the reference path

The key idea is to use the 3-D coordinates of the trees to generate at each iteration a path for both the alley crossing and the headland maneuver. To do so, two steps have been followed. The first one consists in defining adequate points, and the second one in computing the path by fitting a suitable curve. These aspects are targeted here below.

1) Determination of the points: From a motion point of view, the orchard navigation task mainly consists of sequencing alley crossing with U-turns in the headland. The two corresponding cases are considered. To allow the alley crossing, the points used to generate the path must be roughly in the center of the row. To compute these points, a Voronoï diagram is built using the tree coordinates and recomputed at each step. The diagram vertices thus define the desired reference points (see Fig. 3, steps ① and ⑤). Concerning the headland navigation, the points used to generate the paths belong to a particular spiral, such as the one presented in [19]. This spiral is centered around the last detected tree[1], and its position is updated for each new data acquisition (*cf.* Fig. 3, step ③). In addition, the shape of the spiral is adapted at the beginning and the end of the U-turn to connect the first (respectively, last) point of the spiral to the last (respectively, first) vertex of the Voronoï diagram (*cf.* Fig. 3, steps ② and ④).

2) Curve fitting: Finally, the reference path is computed by fitting a NURBS (Non-Uniform Rational B-Spline [21]) curve on these points. It thus allows benefiting from the numerous parameters of NURBS to deal with both straight lines and spirals while obtaining a smoother path. Now that the reference path is available, it remains to follow it.

Several advantages follow from this generation method. First, the reference path is generated using the sole tree positions computed from the data. Thus the proposed approach is fully sensor-based, both in the row and in the headland. Indeed, thanks to the cameras fixed on the robot side, the spiral center remains always visible, even during the U-turn. This latter is thus realized using exteroceptive data and not odometry, which improves the robustness of the execution.

[1]In specific cases, the spiral can be centered on an object positioned at the end of the row, *e.g.,* a bin or a tensioner.

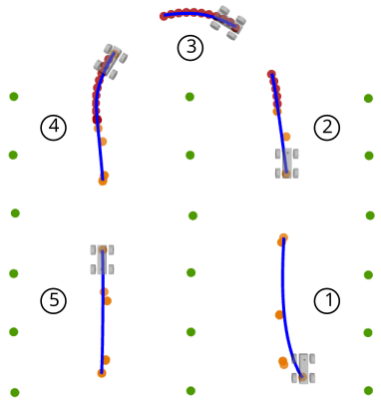


Fig. 3. Examples of path generation. Green circle: tree - Black circle: pivot point - Orange circle: Voronoi vertex - Dark red circle: Spiral point - Blue curve: NURBS - Step 1/5: alley crossing - Step 2: path connecting the alley crossing to the headland maneuver - Step 3: headland maneuver - Step 4: path connecting the headland maneuver to the alley crossing.

Note also that no instrumentation of the orchard is necessary thanks to this approach.

B. Computing the control law

Now, it remains to follow the planned reference path. To do so, we have chosen to impose a geometrical convergence towards the reference path. Therefore, the robot's linear velocity has been fixed to a nonzero constant value. To compute the steering angle, we first express the distance and the orientation errors of the robot with respect to the reference path. Then, to vanish them, we use NMPC. We thus minimize a cost function depending on these errors over a given prediction horizon under the constraints that the inputs are limited. In this way, the obtained solution allows for avoiding actuator saturation.

V. Perception

The perception system is intended to provide the necessary data to feed the control strategy. We first present the point cloud processing that has been used so far to calculate the position of the trunks with respect to the robot. We list the different steps involved in the process and then show the limitations of an approach based solely on point clouds. Finally, we present how recent progress in image processing can be used to improve the trunk pose calculation.

A. Point-cloud processing

To detect the trees present in the robot environment and compute the position of their trunks, we rely on an algorithm that processes in real-time the range component of the point cloud [22]. The method consists in detecting the empty spaces (or shadows), present in the point cloud. Indeed, as it can be seen in Fig. 4, these shadows indicate the presence of trees. Thus, the algorithm first looks for the tree shadows (orange triangles) and then computes their origins (red circles) [22]. This approach offers a high recall rate, *i.e.,* almost all the trees are detected, but it suffers from a fairly low precision rate, *i.e.,*

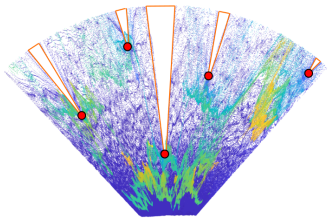


Fig. 4. The orchard point cloud (top view) [22].

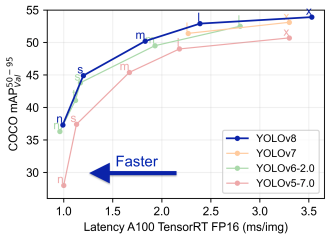


Fig. 5. Latency vs mean Average Precision for different versions of YOLO for the COCO data set [https://github.com/ultralytics/ultralytics].

many elements are confused with trees. To remove the false positive, it was proposed in [22] to implement a filter based on the point cloud density around the estimated tree. Indeed, there are many points in the vicinity of a trunk while there are few ones in the case of a branch. Even if this approach significantly increased the precision rate without degrading the recall rate, it was not sufficient to perform smooth navigation. Indeed, each false positive tree is taken into account during the planning step, leading to inaccurate paths.

B. Image processing

To improve the detection of the trees and the computation of the trunk positions it is proposed to couple the point cloud-based algorithm with an image-based object detection algorithm. In other words, we propose to investigate image-based algorithms to detect the trees to remove the false positive trees detected by the point cloud-based algorithm. Unlike the computation of the position of a given element which is a specific task and requires a handcrafted algorithm, the object detection problem can be tackled with general-purpose algorithms. For this reason, we focus our attention on the recent Deep-Neural-Network-based algorithms which have shown excellent performance. Among the most recognized approaches, we may cite the YOLO (You Only Look Once) [23], U-net [24], or Detectron2 [25] object detector. In order to select the most relevant approach, we have considered the constraints of our navigation framework, *i.e.,* an object detector running at least at 10 fps on an Nvidia Jetson Xavier NX. Thus, the Detectron2 detector was discarded due to its too-small fps rate while the U-net one was not selected because it is less accurate than the last versions of YOLO. We have then chosen to perform the object detection using the YOLO detector, and more specifically to investigate the performances of the YOLOv8 [26] detector for our specific problem[2]. The detector can be used in its nano, small, medium, large, and extra-large version where the width and depth of the convolution modules vary to suit specific applications and hardware requirements. In this work, we focus on the nano and small versions which are lightweight models targeted for low-resource devices (see Fig. 5).

In this work, we aim at detecting the tree trunks in order to navigate in experimental orchards of the "Centre Expérimentation Fruits et Légumes" in Montauban, France.

[2]see [27] for a detailed description of the different version of YOLO.

As can be seen in Fig. 1, these orchards are protected with nets, and tensioners placed at the end of the rows are used to maintain them. Thus, for these specific orchards, it is necessary to detect the tree trunks and the tensioners that will be used as the centers of the spirals when performing the headland maneuvers. To do so, the YOLO networks were trained with images from these orchards to detect two classes: trunks and tensioners. The first set of images was collected on the 30^{th} of November 2022, at the end of the fall season. This first data set is made of 600 images (150 images per camera). The second set of images was collected on the 2^{nd} of June 2023, at the end of the spring season. For the purpose of the project, a fifth camera was installed at the front of the robot. The second data set then contains 900 images (150 images for the first four cameras and 300 for the front one). Thus, for a total of 1500 images, half of the images contain only trunks while the other half contains at least one tensioner (see Fig. 6). In order to train the YOLO network, we have used 80% of the data set, leading to the labeling of 3500 trunks and 700 tensioners. The remaining 20% was used to evaluate the performances of the detection process. Regarding the training, the network was trained using the default parameters, and no specific tuning was performed.

VI. Results

In this section, we present the detection results obtained using the YOLOv8n and YOLOv8s detectors trained and executed on a computer equipped with an Intel i5 9600K CPU, 16 GB of memory, and an Nvidia RTX 2070 GPU.

A. Image-based detection

Figure 7 presents the confusion matrices for both detectors and the precision and recall rates are presented in table I. Both versions have similar performance and allow detecting at least 98% of the trees and 99% of the tensioners. Regarding the precision rate, for both versions, the rate is slightly lower for the tree class than for the tensioner class. Tensioners being human-made objects, they share less similitude with the rest of the environment than the trunks. Moreover, it may happen that non-labeled trees are detected during the evaluation step (for example in the second image of the first row in Fig. 8). This performance rate is perfectly compatible with the needs of the project and it does not seem necessary to further tune

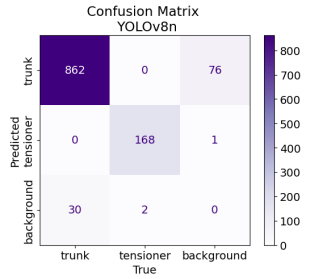
(a) Fall - Front Right.

(b) Fall - Side Left.

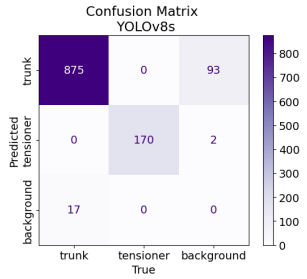
(c) Spring - Front Middle.

(d) Spring - Front Left.

Fig. 6. Images from the data set



(a) YOLOv8n.



(b) YOLOv8s.

Fig. 7. Confusion matrices

the algorithm. Indeed, from a reasonable number of examples, the YOLO detector is able to identify the trunks for different seasons. Finally, regarding the processing time, YOLOv8n version runs at 72 fps while the YOLOv8s one runs at 59 fps. Since the performances are similar, we choose the lightest version, *i.e.,* YOLOv8n, which will free up computing capacity for other processes.

TABLE I
PRECISION-RECALL RATES

	Tree		Tensioner	
	Precision	Recall	Precision	Recall
YOLOv8n	0.91	0.96	0.98	0.99
YOLOv8s	0.91	0.98	0.99	1

B. Trunk detection

We now provide an example of the use of tree detection in the image space in order to improve the navigation process. In Fig. 9, two examples are shown: the first image is from the fall data set and the second one is from the spring data set. The yellow and red dots represent the computed position of the trees detected by the point-cloud-based algorithm. As previously mentioned, it manages to detect the relevant trees but also considers some of the environmental elements as trunks. Thus, using the output of the YOLO detector allows filtering of the false positive results of the point-cloud-based algorithm: the red dots are aligned with a red rectangle and then classified as trees, while the yellow ones do not match with a rectangle and are considered as false positive. This improvement in landmarks detection will significantly impact the path-generation process. Note that image-based detection is used to filter out false positives from the point cloud-based algorithm. Thus, the false positives computed in the image are discarded and have no impact on the final result.

VII. CONCLUSION

This paper has proposed a GNSS-free framework allowing an agricultural robot to autonomously navigate through orchards using embedded exteroceptive sensors. It relies on a perception system made of four RGB-D cameras adequately positioned to detect landmarks both in the headland and the rows. This paper has focused on the evaluation of YOLO, a deep learning solution for image-based detection aiming at improving the existing algorithm based on point clouds only. The obtained results have demonstrated the relevance of YOLO in our context. For future works, several leads are currently considered to enhance the perception: (i) adding a filtering process to track the trees; (ii) improving the detection of the beginning/end of the row; (iii) training YOLO to detect obstacles and design an appropriate avoidance approach. It will then remain to integrate these methods into our robot and to validate them experimentally in the CEFEL orchards.

REFERENCES

[1] J. A. Foley, N. Ramankutty, K. A. Brauman, E. S. Cassidy, J. S. Gerber, M. Johnston, N. D. Mueller, C. O'Connell, D. K. Ray, P. C. West *et al.*, "Solutions for a cultivated planet," *Nature*, vol. 478, no. 7369, p. 337, 2011.

[2] R. Lenain, N. Tricot, and M. Berducat, "La robotique agricole: l'essor de nouveaux outils pour l'agro-écologie," *Sciences Eaux et Territoires*, vol. 29, pp. 64–67, 2019.

[3] S. G. Vougioukas, "Agricultural robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 365–392, 2019.

[4] R. Siegwart, I. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*, ser. A bradford book, Intelligent robotics and autonomous agents series. The MIT Press, second edition, 2011.

[5] M. Li, K. Imou, K. Wakabayashi, and S. Yokoyama, "Review of research on agricultural vehicle autonomous guidance," *International Journal of Agricultural and Biological Engineering*, vol. 2, no. 3, pp. 1–16, 2009.

[6] K. Verbiest, R. and Ruysen, T. Vanwalleghem, E. Demeester, and K. Kellens, "Automation and robotics in the cultivation of pome fruit: Where do we stand today ?" *Journal of Field Robotics*, vol. 38(4), pp. 513–531, 2020.

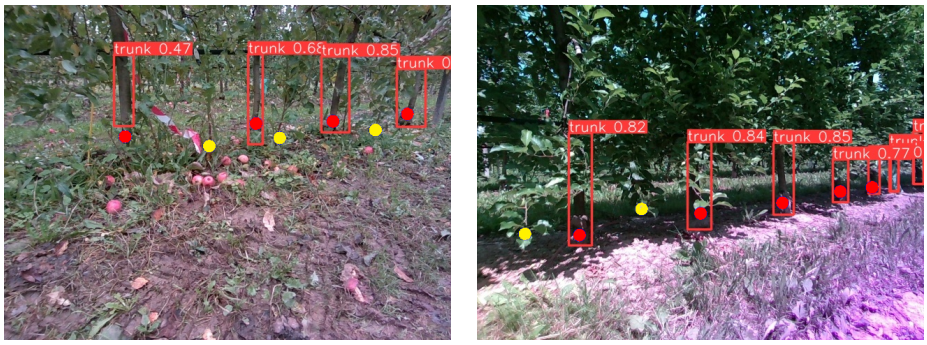
[7] J. Radcliffe, J. Cox, and D. M. Bulanon, "Machine vision for orchard navigation," *Computers in Industry*, vol. 98, pp. 165–171, 2018.

[8] M. Sharifi and C. XiaoQi, "A novel vision based row guidance approach for navigation of agricultural mobile robots in orchards," in *2015 6th International Conference on Automation, Robotics and Applications (ICARA), Queenstown, New Zealand*, 2015, pp. 251–255.

[9] S. Opiyo, C. Okinda, J. Zhou, E. Mwangi, and N. Makange, "Medial axis-based machine-vision system for orchard robot navigation," *Computers and Electronics in Agriculture*, vol. 185, p. 106153, 2021.

(a) Labels. (b) Predictions.

Fig. 8. Example of validation for YOLOv8n



(a) Fall data set. (b) Spring data set.

Fig. 9. Example of the coupling of point-cloud (yellow and red dots) and image (red rectangles) tree detection.

[10] J. Barawid, C. Oscar, M. Akira, I. Kazunobu, and N. Noguchi, "Development of an autonomous navigation system using a two-dimensional laser scanner in an orchard application," *Biosystems Engineering*, vol. 96, pp. 139–149, 2007.

[11] J. C. Andersen, O. Ravn, and N. A. Andersen, "Autonomous rule-based robot navigation in orchards," in *IFAC Proceedings 2010*, no. 16, 2010, pp. 43–48.

[12] S. Hansen, E. Bayramoglu, J. C. Andersen, O. Ravn, N. Andersen, and N. K. Poulsen, "Orchard navigation using derivative free kalman filtering," in *2011 American Control Conference*, 2011, pp. 4679–4684.

[13] J. Libby and G. Kantor, "Deployment of a point and line feature localization system for an outdoor agriculture vehicle," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1565–1570.

[14] S. Marden and M. Whitty, "Deployment of a point and line feature localization system for an outdoor agriculture vehicle," in *Recent Advances in Agricultural Robotics, International workshop collocated with the 13th International Conference on Intelligent Autonomous Systems*, 2014.

[15] A. Danton, J.-C. Roux, B. Dance, C. Cariou, and R. Lenain, "Development of a spraying robot for precision agriculture: An edge following approach," in *2020 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2020, pp. 267–272.

[16] J. Zhang, G. Kantor, M. Bergerman, and S. Singh, "Monocular visual navigation of an autonomous vehicle in natural scene corridor-like environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. American Society of Agricultural and Biological Engineers, 2012, pp. 3659–3666.

[17] J. Zhang, A. Chambers, S. Maeta, M. Bergerman, and S. Singh, "3d perception for accurate row following: Methodology and results," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 5306–5313.

[18] J. Gai, L. Xiang, and L. Tang, "Using a depth camera for crop row detection and mapping for under-canopy navigation of agricultural robotic vehicle," *Computers and Electronics in Agriculture*, vol. 188, p. 106301, 2021.

[19] A. Durand-Petiteville, E. Le Flecher, V. Cadenat, T. Sentenac, and S. Vougioukas, "Design of a sensor-based controller performing u-turn to navigate in orchards," in *International Conference on Informatics in Control, Automation and Robotics*, vol. 2, 2017, pp. 172–181.

[20] Y. Li, Y. Ruichek, and C. Cappelle, "3d triangulation based extrinsic calibration between a stereo vision system and a lidar," *Conference Record - IEEE Conference on Intelligent Transportation Systems*, pp. 797–802, 10 2011.

[21] L. Piegl and W. Tiller, *The NURBS Book*, 2nd ed. New York, NY, USA: Springer-Verlag, 1996.

[22] A. Durand-Petiteville, E. Le Flecher, V. Cadenat, T. Sentenac, and S. Vougioukas, "Tree detection with low-cost three-dimensional sensors for autonomous navigation in orchards," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3876–3883, 2018.

[23] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[25] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[26] G. Jocher, A. Chaurasia, and J. Qiu, "Yolo by ultralytics." https://github.com/ultralytics/, 2023.

[27] J. Terven and D. Cordova-Esparza, "A comprehensive review of yolo: From yolov1 to yolov8 and beyond," *arXiv:2304.00501*, 2023.