



**HAL**  
open science

# Multi-camera GPS-free Nonlinear Model Predictive Control strategy to traverse orchards

Antoine Villemazet, Adrien Durand-Petiteville, Viviane Cadenat

► **To cite this version:**

Antoine Villemazet, Adrien Durand-Petiteville, Viviane Cadenat. Multi-camera GPS-free Nonlinear Model Predictive Control strategy to traverse orchards. European Conference on Mobile Robots, Sep 2023, Coimbra (PT), Portugal. hal-04634897

**HAL Id: hal-04634897**

**<https://laas.hal.science/hal-04634897v1>**

Submitted on 4 Jul 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Multi-camera GPS-free Nonlinear Model Predictive Control strategy to traverse orchards

A. Villemazet<sup>1,2</sup>, A. Durand-Petiteville<sup>3</sup> and V. Cadenat<sup>1,2</sup>

**Abstract**—This paper deals with autonomous navigation through orchards. It proposes a multi-camera GPS-free strategy relying on a Nonlinear Model Predictive Control (NMPC) scheme to follow a reference path. This latter, based on a Voronoi diagram for the row traversals or a spiral model for the headland maneuvers, is computed as a Non-Uniform Rational Spline (NURBS) curve making it possible to deal with multiple orchard layouts. The method has been implemented on our robot and validated through experimentation conducted in an orchard.

## I. INTRODUCTION

Robotics has been identified as one of the major solutions to promote truly sustainable agriculture where the necessary production increase matches environmental concerns [1]. In this work, we focus on orchard mechanization, and more specifically on the autonomous navigation system, which is mandatory to realize some agricultural tasks such as mowing, spraying, or harvesting. When moving through an orchard, a robot has to autonomously drive from the entrance of an alley to its exit, and then move to the next alley by navigating in the headlands, *i.e.*, the uncultivated area between the edge of the trees and the orchard boundary used for machinery maneuvers. It repeats these two steps to cover the whole area of interest (see Fig. 1(a)).

As the GPS signal is often blocked or perturbed by the dense canopy or nets protecting the trees [2], the existing navigation strategies rely on embedded sensors, either vision systems [3] [4] [5] [6] or LiDAR sensors [7] [8]. These works propose to compute and then follow a straight line passing through the middle of the alleys. The obtained line may be disturbed by the natural environment where branches and foliage are uneven and lighting conditions significantly vary. Moreover, these approaches do not allow coping with modern orchards whose circular layout is specifically designed to control pests thanks to ecological processes [9] (see Fig. 1(b)). Regarding maneuvers in the headland, the few existing works on this topic use dead reckoning because of the lack of sensory information in these zones. In such a case, the execution robustness and repeatability are significantly reduced [10]. We may nonetheless mention the following methods

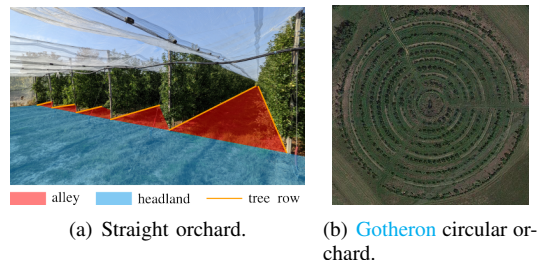


Fig. 1. Example of orchards.

where dead reckoning is coupled with other techniques in an attempt to overcome this drawback: a slip compensation solution [11], automatic detection of the rows extremities using either a laser [12] or dedicated artificial landmarks [13]. It then would be interesting to provide a navigation strategy able to cope with the different types of orchard layouts, while improving the headland maneuver robustness and avoiding any environmental instrumentation. Some of our earlier works have proposed to perform the U-turn using data provided by a 2d laser rangefinder. We have designed a sensor-based nonlinear controller following a spiral centered on the last row tree [14] [15]. This approach was later extended to unify both in-row and headland navigation in a unique spiral-based framework [16] in a straight orchard. Despite promising results regarding the use of a unique sensor-based framework for both parts of the navigation, the solution presented oscillation issues, especially when re-entering the alley. This was due to the idea of modeling the orchard navigation as a point regulation problem where the robot had to reach a sequence of waypoints, *i.e.*, without considering the robot orientation. Moreover, it is necessary to use a more robust perception method. Indeed, although allowing to validate the approach in simulation, a 2d laser rangefinder has a planar field of view, not allowing to detect trees in a robust way in an orchard.

In this paper, we present a novel sensor-based framework allowing the robot to navigate through an entire orchard, *i.e.*, both the alleys and the headlands, without adding any landmark nor considering a particular layout. First, the robot has been equipped with a vision system made of four RGB-D low-cost cameras. On the one hand, it offers a relatively inexpensive solution to acquire 3D data, thus increasing the tree detection capabilities with respect to 2D laser rangefinder-based solutions. On the other hand, it allows benefiting from an overall large field of view to perceive trees both in the row and in the headland, making sensor-based control possible.

<sup>1</sup>Univ. de Toulouse, CNRS, UPS, Toulouse, France

<sup>2</sup>CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France  
{avillemaze, cadenat}@laas.fr

<sup>3</sup>Departamento de Engenharia de Mecânica, Universidade Federal de Pernambuco UFPE, Recife, PE, Brazil  
adrien.durandpetiteville@ufpe.br

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)

979-8-3503-0704-7/23/\$31.00 ©2023 IEEE

Second, instead of defining the orchard navigation in terms of point regulation, we now state it as a path-following problem, to reduce the previous oscillations. The reference path is a local path iteratively updated using the position of the trees computed by the vision system. To do so, we use a Voronoi diagram for row traversals and a spiral model for the headland maneuvers. Next, a Non-Uniform Rational Spline (NURBS) curve is computed to unify the different sections of the path and provides a smooth reference to follow. This problem formulation presents three advantages: (i) the in-row and headland navigation are unified during the path computation and are not merged at the controller level as in [16]; (ii) it allows dealing with numerous orchard layouts (and not only rectangle-shaped ones); and (iii) it offers a more consistent reference than the straight-line following approach and takes into account the robot orientation. The path following is performed using a Nonlinear Model Predictive Control (NMPC) scheme coupled with a Frenet-based formulation of the problem. It provides an efficient minimization of the error between the computed and desired paths over the whole prediction horizon while taking into account specific constraints such as actuator saturation. Finally, in order to evaluate the relevancy of the proposed approach, it is first compared with [16] using the Gazebo simulator, and next implemented on the Hunter 2.0 robotic platform to navigate in an orchard.

The rest of the paper is organized as follows. We first present the robotic system before focusing on the proposed navigation framework. The simulated and experimental results are then presented to show the approach's efficiency.

## II. MODELLING

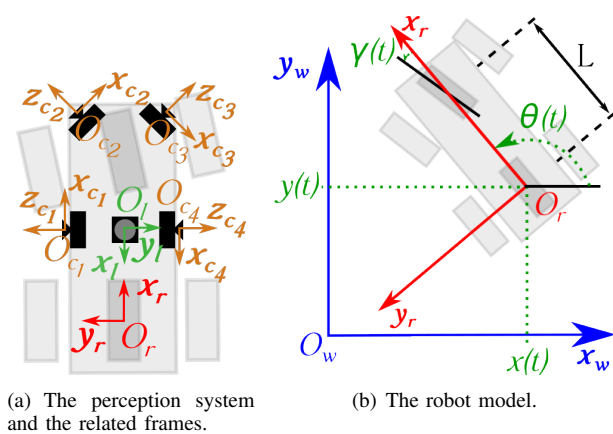


Fig. 2. The robotic system models.

The considered platform is the Agilex Hunter 2.0 car-like robot equipped with a laser rangefinder and four RGB-D cameras (see Fig. 2(a)). To obtain the necessary wide field of view, two cameras are placed at the front of the robot and respectively oriented left forward and right forward, while the two other ones are placed on the sides of the platform (see Fig. 2(a)). To model the system, we define  $F_w = (O_w, \mathbf{x}_w, \mathbf{y}_w, \mathbf{z}_w)$  as the world frame,  $F_r = (O_r, \mathbf{x}_r, \mathbf{y}_r, \mathbf{z}_r)$

as the robot frame,  $F_l = (O_l, \mathbf{x}_l, \mathbf{y}_l, \mathbf{z}_l)$  as the laser frame, and  $F_{c_i} = (O_{c_i}, \mathbf{x}_{c_i}, \mathbf{y}_{c_i}, \mathbf{z}_{c_i})$  as the frame of the  $i^{th}$  camera, with  $i \in [1, 4]$ . We rely on the Ackermann model to represent the robot and therefore its pose is given by  $\chi(t) = [x(t), y(t), \theta(t), \gamma(t)]$ , where  $x(t)$  and  $y(t)$  are the coordinates of  $O_r$  in  $F_w$ ,  $\theta(t)$  represents the angle from  $\mathbf{x}_w$  to  $\mathbf{x}_r$ , and  $\gamma(t)$  is the angular position of the steering angle (see Fig. 2(b)). Moreover, we define the control vector by  $\mathbf{U}(t) = [v(t), \gamma(t)]$  where  $v(t)$  is the linear velocity along  $\mathbf{x}_r$ . For such a system, considering  $L$  the distance between the front and rear wheels, the kinematic model is:

$$\begin{cases} \dot{x}(t) = v(t) \cos(\theta(t)) \\ \dot{y}(t) = v(t) \sin(\theta(t)) \\ \dot{\theta}(t) = \frac{v(t)}{L} \tan(\gamma(t)) \end{cases} \quad (1)$$

## III. ORCHARDS TRAVERSAL STRATEGY

To navigate in the orchard, the robot has to cross an alley, maneuver in the headlands to switch from an alley to the next one, and repeat these two steps until its navigation is completed. In this section, we detail the different processes involved in the proposed navigation framework. We first present the vision system and data processing. Next, we introduce our solution to compute the local path to follow both in the alleys and in the headlands. Finally, our NMPC-based path-following strategy is detailed.

### A. Data processing

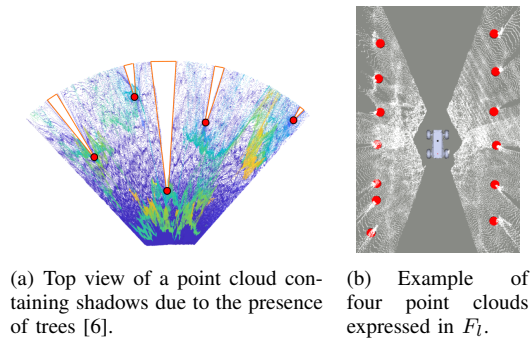


Fig. 3. Data processing examples.

The presented navigation strategy relies on the position of the tree trunks in the current robot frame. The positions are computed using the point clouds provided by the four onboard RGB-D cameras. To do so, we rely on the algorithm [6] which estimates the tree trunk positions by detecting shadows in the point cloud due to the presence of trees (see Fig. 3(a)). The algorithm processes, therefore, the four point clouds separately and provides the position of the detected trees in each camera frame  $F_{c_i}$ .

The tree coordinates must then be expressed in a common frame, which is the laser frame  $F_l$ . Indeed, the laser field of view overlaps the one of the four cameras, allowing computing the extrinsic parameters between  $F_l$  and the four  $F_{c_i}$ . The calibration process between  $F_l$  and  $F_{c_i}$ , *i.e.*, the computation of the homogeneous transformation matrix  $H_{l|c_i}$

is performed using [17]. An example of the result is shown in Fig. 3(b).

### B. Path generation

In this section, we present how the tree coordinates in the current robot frame are used at each iteration to generate a new path to follow. The proposed path generation is a two-step process: first, we calculate a set of waypoints, and next, we compute a path based on these waypoints. The waypoints computation is done differently for the alley traversing and the headland navigation. For the alleys, we generate a Voronoi diagram [18] using the tree coordinates. The vertices of the diagram, which approximately lie in the middle of the row, will then be used to compute the path to follow (see steps 1 and 5 in Fig. 5).

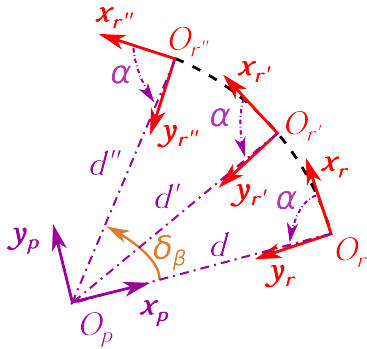


Fig. 4. Several robot frames while describing a spiral.

For the headland maneuver, we propose to compute waypoints lying on a spiral centered on the last tree of the row, called the pivot point and denoted  $O_p$  (see step 3 in Fig. 5). It is used as the origin of the frame  $F_p$ , whose orientation is arbitrary. We rely on the spiral model presented in [19] where  $O_p$ , the pivot point, is considered as the spiral center,  $d(t)$  is the distance between the robot and the pivot point, *i.e.*, between  $O_p$  and  $O_r$ , and  $\alpha(t)$  is the oriented angle from the  $\mathbf{x}_r$  vector to the  $\mathbf{O}_r\mathbf{O}_p$  one (see Fig. 4). Finally  $\beta(t)$  is the angle between  $\mathbf{x}_p$  and  $\mathbf{O}_p\mathbf{O}_r$ . It is shown in [19] that if both  $v(t)$  and  $\alpha(t)$  are constant, then  $O_r$  describes a spiral, and the following equations hold:

$$\dot{d}(t) = -v \cos \alpha \quad (2)$$

$$d(\beta) = d_0 e^{\cot \alpha (\beta_0 - \beta)} \quad (3)$$

Eq. (2) shows that the type of spiral only depends on parameter  $\alpha$ . Indeed, if  $\alpha \in [-\pi; 0]$ , then  $O_r$  turns clockwise with respect to  $O_p$  and counter-clockwise if  $\alpha \in [0; \pi]$ . Moreover, if  $\alpha \in ]-\pi; -\frac{\pi}{2}[ \cup ]\frac{\pi}{2}; \pi[$ , then the spiral is outward and inward if  $\alpha \in ]-\frac{\pi}{2}; 0[ \cup ]0; \frac{\pi}{2}[$ . It becomes a circle if  $\alpha = \pm \frac{\pi}{2}$ , with a radius equal to  $d$ . Thus, the design of the spiral first consists in selecting a value for  $\alpha$  and an initial distance  $d_0$ . Finally, the set of waypoints belonging to the spiral is computed over an angular horizon  $\delta_\beta$  using (3). Note, that the frame  $F_p$  is readjusted at each iteration to align the  $\mathbf{x}_p$  and  $\mathbf{O}_p\mathbf{O}_r$  vectors. This approach allows maneuvering in the headlands on the sole basis of the current

exteroceptive data and does not require any localization process.

The waypoints having been computed for both alleys and headlands, it is then necessary to connect them to make the robot navigate in the orchard. In other words, we have to connect the spiral to the last vertex of the Voronoi diagram to make the robot exit the alley and to connect the spiral to the first vertex of the new diagram when the robot enters a new alley. First, when the robot exits the alley,  $d_0$  is defined as the distance between the pivot point  $O_p$  and the last vertex of the diagram in order to connect the two parts of the path. Moreover, we set up  $\alpha = \pm \frac{\pi}{2}$  to make the robot follow a circle of radius  $d_0$  centered on the pivot point (see step 2 in Fig. 5). This approach initially makes it possible to safely turn around the pivot point but does not guarantee that the spiral will connect with the first vertex of the next alley diagram. Thus, once the next alley is visible and it is possible to compute the next Voronoi diagram, the spiral parameters are modified. First,  $\alpha$  is adjusted to make the spiral pass via the vertex (from here the path is no more a circle, but a spiral), and the angular horizon  $\delta_\beta$  is modified to make coincide the end of the spiral with the vertex (see step 4 in Fig. 5). Setting up the spiral parameters as described guarantees the continuity between the different parts of the path.

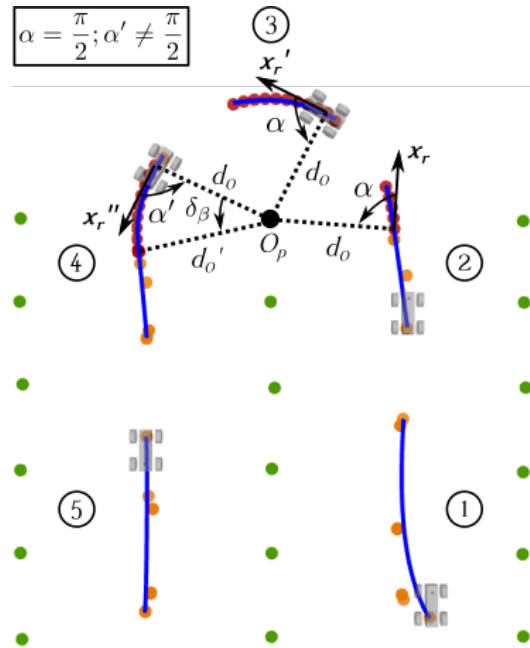


Fig. 5. Examples of path generation. Green circle: tree - Black circle: pivot point - Orange circle: Voronoi vertex - Dark red circle: Spiral point - Blue curve: NURBS - Step 1/5: alley crossing - Step 2: path connecting the alley crossing to the headland maneuver - Step 3: headland maneuver - Step 4: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

Finally, we propose to use a NURBS (Non-Uniform Rational B-Spline) [20] curve to compute a smooth path passing through the waypoints. To summarize, this particular type of curve is defined by a set of weighted control points that locally influence its curvature. Mathematically, its general

form is given by [20]:

$$C(u) = \frac{\sum_{i=1}^n N_{i,p}(u) w_i \mathbf{P}_i}{\sum_{i=1}^n N_{i,p}(u) w_i}, u \in [0, 1] \quad (4)$$

where  $n$  is the number of control points  $\mathbf{P}_i$ ,  $w_i$  are the corresponding weights and  $N_{i,p}$  are the B-Spline basis function of  $p^{\text{th}}$  degree. More details are available in [20]. In our case, the control points are either the endpoints of the Voronoi segments (see orange circles in Fig. 5) or the points belonging to the spiral (see dark red circles in Fig. 5). The NURBS curve was chosen because of its three properties which are useful in our application: the degree of the curve which depends on the number of control points, the knot vector (used by the B-Spline basis function), and the weighted control points. First, the high degree of the curve allows for generating a path for both straight and curved tree rows as well as the circular path for the headland maneuver. Next, the knot vector ensures that the curve passes through the first and the last control points allowing to avoid an abrupt re-alignment of the robot on the reference path. Finally, the weights allow us to adjust the influence of the control points on the curve to make a smooth path and thus obtain a better robot trajectory. We propose to define them as follows:  $[1, w_1, \dots, w_{n-2}, 1]$ . First, the first and last weights are set to 1 with respect to the second property.  $w_1, \dots, w_{n-2}$  must thus be chosen as a compromise to obtain the most stable path over the iterations.

### C. Path following

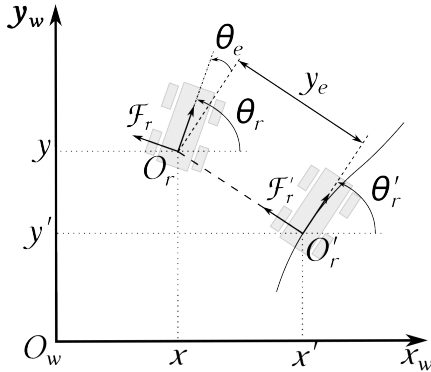


Fig. 6. Principle of the path tracking. [21]

We now present our approach for following a given path using an NMPC controller. As shown in Fig. 6, it consists in orthogonally projecting the center of the robot  $O_r$  on the reference path to define a Frenet frame  $F'_r$  associated with  $O_r$ . It is then possible to define  $\theta_e$  as the orientation error and  $y_e$  as the lateral error. The path following is then performed by minimizing the error vector  $\mathbf{e}_{pf} = [y_e, \theta_e]$  over a prediction horizon. This approach does not require including the linear velocity in the minimization problem as it is not aiming at reaching a set of points at a given instant sampled from the path, such as in [16]. The linear velocity can be fixed at a constant value or computed accordingly

to a different criterion, such as terrain traversability. Thus, in this work, the linear velocity  $v(t)$  is considered constant so that  $v(t) = v, (v \neq 0)$ . The only control input is thus the steering angle  $\gamma$ . The path following is performed via an NMPC scheme considering the following optimization problem:

$$\bar{\gamma}^*(k) = \min_{\bar{\gamma}(k)} (J_{N_p}(\mathbf{e}_{pf}(k), \bar{\gamma}(k))) \quad (5)$$

with

$$J_{N_p}(\mathbf{e}_{pf}(k), \bar{\gamma}(k)) = \sum_{p=k+1}^{k+N_p} \hat{\mathbf{e}}_{pf}(p)^T \hat{\mathbf{e}}_{pf}(p) + \lambda_\gamma (\gamma(p) - \gamma(p-1))^2 \quad (6)$$

subject to

$$\hat{\mathbf{e}}_{pf}(p+1) = f(\hat{\mathbf{e}}_{pf}(p), \gamma(p)) \quad (7a)$$

$$\hat{\mathbf{e}}_{pf}(k) = \mathbf{e}_{pf}(k) \quad (7b)$$

$$C(\bar{\gamma}^*(\cdot)) \leq 0 \quad (7c)$$

It computes an optimal steering angle sequence  $\bar{\gamma}^*(k)$  of  $\bar{\gamma}(k)$ , with  $\bar{\gamma}(k) = [\gamma(k), \dots, \gamma(k+N_p)]$  which minimizes the cost function  $J_{N_p}$  over a prediction horizon of  $N_p$  steps while taking into account the physical boundaries of the robot actuators as constraints  $C(\bar{\gamma}^*(k))$ . The values of both the prediction and control horizons are considered equal.

**Cost function:**  $J_{N_p}$  is divided in two parts. The first one is defined as the sum of the quadratic predicted configuration  $\hat{\mathbf{e}}_{pf}$ , and is intended to track the reference path. The second one is the sum of the quadratic differences between two consecutive commands, weighted by the parameter  $\lambda_\gamma$ , which allows smoothing of the control inputs and limiting velocities variations between two instants.

**Remark:** To project the predicted positions onto the reference path, we discretize the NURBS curve and search for the closest position belonging to the path for each prediction. The search relies on the k-d tree structure [22] which proposes an efficient nearest neighbor search based on a space-partitioning data structure.

**Prediction model:** Assuming that the steering angle  $\gamma(t_1)$  is constant between the instant  $t_1$  and  $t_2 = t_1 + T_s$ , where  $T_s$  is the sampling time, the robot predicted pose is computed by integrating (1) with a Runge-Kutta method of order 4.

**Input constraints:** The input constraints take into account the physical limits of the mobile base. They are given by:

$$\begin{bmatrix} \gamma(i) - \gamma_u \\ \gamma_l - \gamma(i) \end{bmatrix} \leq 0 \quad (8)$$

where  $i \in [1, N_p]$ ,  $\gamma_l$  and  $\gamma_u$  are respectively the lower and upper boundaries.

## IV. RESULTS

In this section, we present the obtained results, first using a simulator, then using a robotic platform. In both cases, the considered robot is the Hunter 2.0 car-like mobile base. The robot is equipped with a vision system consisting of four Intel Realsense RGB-D cameras, two D455 and two

D435, positioned as shown in Fig. 7(b) to enlarge the field of view as explained earlier. The robot has also been endowed with Slamtech’s RPLIDAR S1 range-finder for the camera calibration step. The physical boundaries of the Hunter 2.0 actuators as well as the optimal ranges of the cameras are shown in Table I.

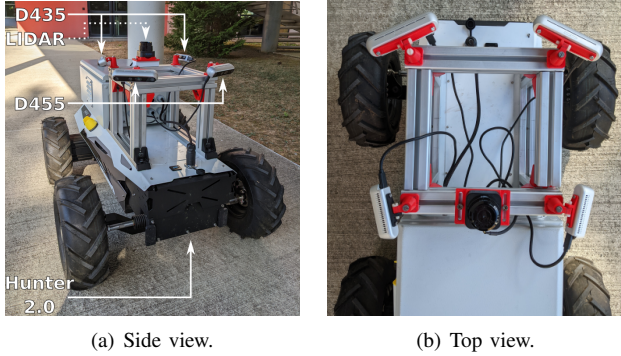


Fig. 7. Robotic platform.

TABLE I  
SYSTEM SPECIFICATIONS.

	minimum range	maximum range
Linear velocity	-1.5 m/s	1.5 m/s
Steering angle	-0.461 rad	0.461 rad
D455	0.6 m	6 m
D435	0.3 m	3 m

Furthermore, the robot is equipped with an NVIDIA Jetson Xavier NX GPU and an Intel Core i7-1165G7 48 GB RAM CPU. The former is dedicated to data processing while the latter calculates the control inputs. The implementation relies on the C++ 14 language and the ROS middleware. The data processing part uses the OpenCV and PCL libraries and is partially implemented using the CUDA language. The NMPC part is based on several libraries allowing to implement the following features: the [clustering method](#), the [Voronoi diagram](#), the [NURBS curve](#), the [k-d tree structure](#) and the [SQP solver](#).

### A. Simulation

We first compare the proposed approach, the NURBS-based method, with the one described in [16], the spiral-based method. We recall that the NURBS-based method relies on a path following while the spiral-based one consists in reaching a sequence of positions. The simulations are performed with the straight and circular orchards shown in Fig. 8(a) and Fig. 8(b) where the trees’ position and orientation were randomly modified to obtain a more realistic layout. The parameters for both methods are listed in Table II. For the spiral-based method, the set of parameters is similar to the one used in [16] with the exception of the solver tolerance values which are slightly modified to increase performance in the circular orchard. In addition to using a different cost function, path-following vs. positioning, the methods differ

in their use of a terminal constraint. Indeed, the spiral-based method requires a terminal constraint to guarantee the stability of the positioning process while it is not required for the path-following approach. Finally, the lower/upper limits of the input constraints  $\gamma_l$  and  $\gamma_u$  in the NURBS-based method are no longer the physical limits of the steering angle of the Hunter 2.0 actuators, as in the spiral-based method, but the maximum positions reachable in  $T_s$  second ( $\pm 2$  degrees for the Hunter 2.0). This allows only feasible commands to be calculated for the robot, thus reducing solver disturbances between iterations and improving robot behavior.

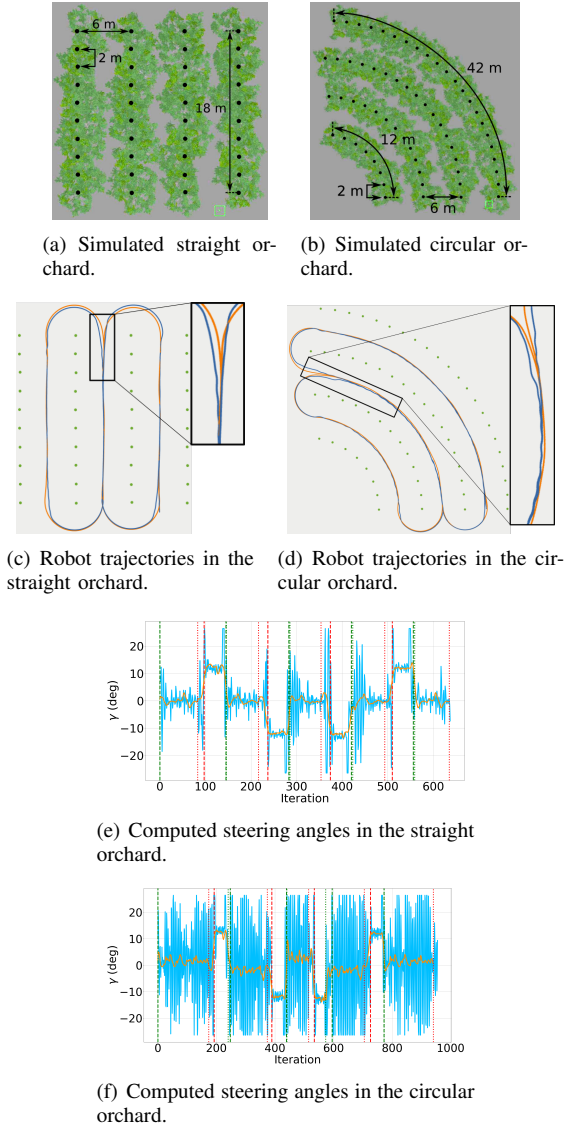


Fig. 8. Navigation results in simulation. (c-d-e-f) Blue plots: spiral-based method results - Orange plots: NURBS-based method results - (e-f) Green vertical lines: Start of the alley crossing for the spiral-based method (dashed lines) and the NURBS-based one (dotted lines) - Red vertical lines: Start of the headland maneuver for the spiral-based method (dashed lines) and the NURBS-based one (dotted lines).

Figure 8 presents the results obtained for both methods and orchard layouts. In Fig. 8(c) and Fig. 8(d), it can be seen that the robot successfully achieves the navigation

TABLE II  
SIMULATION PARAMETERS.

Approach	$v$	$T_s$	$N_p$	Maximum time <sup>a</sup>	Absolute tolerance <sup>a</sup>	$\phi_{ZEC}$ <sup>b</sup>	Number of points NURBS	$w_i$	$\lambda_\gamma$
Spiral	1	0.2	12	0.16	$10^{-3}$	$10^{-4}$	N/A	N/A	N/A
NURBS	1	0.1	20	0.09	$10^{-6}$	N/A	3000	$10^{-2}$	10

<sup>a</sup>Stopping criterion of the SQP solver.

<sup>b</sup>Zero terminal equality constraint tolerance.

task in the straight and curved orchards relying on both the spiral-based and the NURBS-based methods. Indeed, it successfully drives through the three alleys and maneuvers in the headlands to switch from one alley to the next one performing a 126 meters long path in the straight orchard and a 187 meters long one in the circular one. Thus, from a task point of view, both approaches are capable of navigating in different orchard layouts unlike other works focusing on straight lines. However, from a control perspective, it can be noticed that the spiral-based method tends to generate oscillations when the path is curved (entrance of a new alley or in the alleys of the circular orchard). This is due to the fact that it is a positioning approach not taking into account the robot's orientation. Thus, as long as the robot is oriented toward the next goal point, it navigates without oscillating, *e.g.*, when crossing the straight alleys. However, when it is not initially oriented toward the point to reach, it has a tendency to oscillate *e.g.*, when entering a new alley or driving through a curved alley. On the contrary, the NURBS-based method presented in this paper does not lead to such oscillations. Indeed, using a path-following formulation of the problem allows for taking into account the robot's orientation. Thus, the quality of the robot path is consistent when navigating in a straight or curved alley or when maneuvering in the headlands. This analysis is supported by the evolution of the steering angles shown in Fig. 8(e) and 8(f). Indeed it can be seen that the value of the steering angle varies more for the spiral-based approach than for the NURBS-based one, for both orchard layouts. Thus, despite the interest in the spiral-based method, the NURBS-based method significantly improves the quality of the navigation system.

### B. Experimentation

To show the efficiency of the NURBS-based method, we conducted an experiment at the agricultural high school<sup>1</sup> in Auzeville-Tolosane, France. The considered orchard has 40 meters long by 4 meters wide tree rows with a space of 1 meter between two consecutive trees. At the time of the experiments, only four tree rows were usable. The following parameters were chosen:  $v = 0.5m/s$ ,  $T_s = 0.1s$ ,  $N_p = 20$ , which corresponds roughly to a prediction window of 1 meter, and  $\lambda_\gamma = 5$ . The other parameters remain identical to the simulation.

The orchard navigation is presented in the [attached video](#). Some additional snapshots, completed with an RVIZ view of the detected trees and the computed path, display the

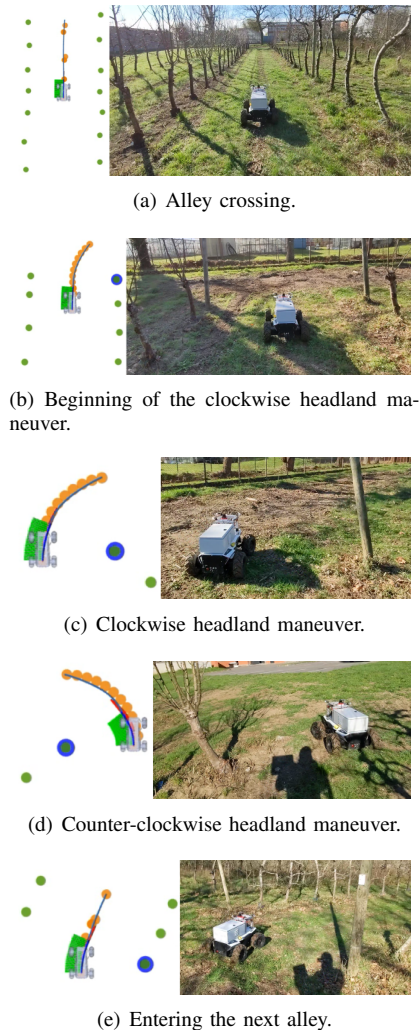


Fig. 9. Navigation snapshots - left: robot centered RVIZ data visualization (green circle: detected trees - blue circle: selected pivot point - orange circle: NURBS control points - blue curve: NURBS) - right: video screenshots.

main key steps of the navigation in Fig. 9. As shown in the video, the navigation task is correctly achieved. The robot successfully moves along the three alleys twice and performs two clockwise and two counter-clockwise U-turn maneuvers in the headlands. It has thus realized a 222 meters long path in 480 seconds. Now, let us go into further details and analyze the main steps of the navigation: the sequence of row followings and U-turn maneuvers. First, the system successfully computes a path based on the tree positions allowing to drive through the alley (see Fig. 9(a)). The path computing/following process is repeated at each iteration

<sup>1</sup>"Lycée Général et Technologique Agricole des Sciences Vertes"

until the row crossing is achieved. Once the robot gets closer to the end of the alley, one of the last trees is selected as the pivot point and the generated path is composed of both a row crossing section and a spiral one (see in Fig. 9(b)). The pivot point is chosen so that the robot makes a loop in the orchard and thus sequences the four U-turns. Next, the robot performs the clockwise/counter-clockwise headland maneuver following a spiral computed on the sole basis of the pivot point as seen in Fig. 9(c) and 9(d). Finally, in Fig. 9(e) the robot is about to reach the next alley. The spiral parameters are adjusted to connect the spiral section of the computed path to the row-crossing section. By doing so, the robot manages to enter the next alley and then restart the crossing step.

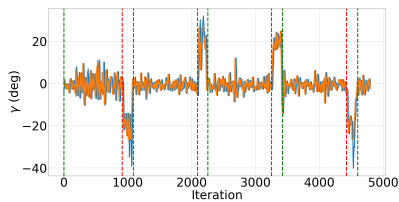


Fig. 10. Computed and applied steering angles. Blue line: computed steering angles - Orange line: Applied steering angles - Green vertical dashed line: beginning of the alley crossing - Red vertical dashed line: beginning of the headland maneuver.

Finally, the computed and applied commands are displayed in Fig. 10. As shown in this figure, the computed steering angle tends towards 0 degrees during the alley crossings and towards  $\pm 18$  degrees during the headland maneuvers, which is consistent with the orchard layout. The variations are mainly due to the variations of the computed tree coordinates. Indeed, these latter are computed on the sole basis of the current data and the results may differ from one iteration to the other. As the command frequency rate is higher than the steering angle capabilities, the robot path is not impacted by these oscillations. This leads to appropriate overall behavior and thus validates the control strategy.

## V. CONCLUSION

This paper presents a novel multi-camera-based NMPC strategy allowing autonomously navigating through various shaped orchards without instrumentation. The proposed method relies on an original fully vision-based computation and update of the reference path and does not require any map. The path following problem is expressed using the NMPC framework, making easier the transition between in-row and headland navigation and the constraints handling. The approach has been implemented and validated through an experimental campaign conducted in an orchard. The obtained results show the relevance and efficiency of the approach. Regarding future works, we plan to increase the perception system robustness by adding a particle filter able to track the trees and coupling the point processing to an image-based tree detection. We also aim at integrating new constraints in NMPC to avoid obstacles and to reduce undesired vibrations due to rough terrains.

## REFERENCES

- [1] J. F. Reid, "The impact of mechanization on agriculture," *Bridge*, vol. 41, no. 3, pp. 22–29, 2011.
- [2] 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.
- [3] J. Radcliffe, J. Cox, and D. M. Bulanon, "Machine vision for orchard navigation," *Computers in Industry*, vol. 98, pp. 165–171, 2018.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] P. M. Blok, K. van Boheemen, F. K. van Evert, J. IJsselmuiden, and G.-H. Kim, "Robot navigation in orchards with localization based on particle filter and kalman filter," *Computers and Electronics in Agriculture*, vol. 157, pp. 261–269, 2019.
- [8] 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.
- [9] A. Favery. The gotheron orchard: when biodiversity becomes circular. [Online]. Available: <https://www.inrae.fr/en/news/gotheron-orchard-when-biodiversity-becomes-circular>
- [10] S. G. Vougioukas, "Agricultural robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 365–392, 2019.
- [11] V. Subramanian and T. F. Burks, "Autonomous vehicle turning in the headlands of citrus groves," in *2007 ASAE Annual Meeting*. American Society of Agricultural and Biological Engineers, 2007, p. 1.
- [12] G. Bayar, M. Bergerman, A. B. Koku, and E. Ilhan Konukseven, "Localization and control of an autonomous orchard vehicle," *Computers and Electronics in Agriculture*, vol. 115, pp. 118–128, 2015.
- [13] J. Zhang, S. Maeta, M. Bergerman, and S. Singh, "Mapping orchards for autonomous navigation," in *2014 Montreal, Quebec Canada July 13–July 16, 2014*. American Society of Agricultural and Biological Engineers, 2014, p. 1.
- [14] 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.
- [15] E. Le Flecher, A. Durand-Petiteville, F. Gouaisbaut, V. Cadenat, T. Sentenac, and S. Vougioukas, "Nonlinear output feedback for autonomous u-turn maneuvers of a robot in orchard headlands," in *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2019.
- [16] A. Villemazet, A. Durand-Petiteville, and V. Cadenat, "Autonomous navigation strategy for orchards relying on sensor-based nonlinear model predictive control," in *2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 05 2022.
- [17] 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.
- [18] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations - Concepts and Applications of Voronoi Diagrams*. John Wiley, 2000.
- [19] K. N. Boyadzhiev, "Spirals and conchospirals in the flight of insects," *The college mathematics Journal*, vol. 30, no. 1, p. 23, 1999.
- [20] L. Piegl and W. Tiller, *The NURBS Book*, 2nd ed. New York, NY, USA: Springer-Verlag, 1996.
- [21] V. Cadenat, P. Souères, and T. Hamel, "A reactive path-following controller to guarantee obstacle avoidance during the transient phase," *International Journal of Robotics and Automation*, vol. 21, no. 4, pp. 256–265, 2006.
- [22] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, p. 509–517, sep 1975. [Online]. Available: <https://doi.org/10.1145/361002.361007>