A Navigation Architecture for Ackermann Vehicles in Precision Farming

Renzo Fabrizio Carpio*, Ciro Potena*, Jacopo Maiolini, Giovanni Ulivi,
Nicolás Bono Rosselló, Emanuele Garone, Andrea Gasparri†

***Abstract*— In this letter, inspired by the needs of the European H2020 Project PANTHEON[1], we propose a full navigation stack purposely designed for the autonomous navigation of Ackermann steering vehicles in precision farming settings. The proposed stack is composed of a local planner and a pose regulation controller, both implemented in ROS. The local planner generates, in real-time, optimal trajectories described by a sequence of successive poses. The planning problem is formulated as a real-time cost-function minimization problem over a finite time horizon where the Ackermann kinematics and the presence of obstacles are encoded as constraints. The control law ensures the convergence toward each of these poses. To do so, in this paper we propose a novel non-smooth control law designed to ensure the solvability of the pose regulation problem for the Ackermann vehicle. Theoretical characterization of the convergence property of the proposed pose regulation controller is provided. Numerical simulations along with real-world experiments are provided to corroborate the effectiveness of the proposed navigation strategy.**

Fig. 1: A farming robot navigating a hazelnut orchard environment. The field traversing route is represented by the blue solid and the yellow dotted (the part of the route on the other row) lines.

I. Introduction

Precision Agriculture is a farming management concept based on observing, measuring and responding to inter and intra-field variability in crops [1]. Intervariability may result from a significant number of factors such as weather variables (temperature, precipitations, relative humidity, etc.), soil characteristics (texture, depth, nitrogen levels), cropping practices (till/no-till farming), weeds and diseases, etc.

As pointed out in [2], the availability of autonomous system architectures gives the opportunity to develop a new range of flexible agricultural equipment based on small, smart machines that reduce waste, improve economic viability, reduce environmental impact, and increase food sustainability. As a matter of fact, there is considerable potential for robotics technology to provide targeted interventions on different aspects of farming, ranging from selective pesticides spraying to mechanical or laser-based weed removals. However, to effectively carry out almost any (farming) task, a robot must guarantee autonomous and safe navigation of the environment.

The problems of planning and control of ground vehicles have been widely investigated in the literature, e.g. [3]–[11].

Typical application scenarios include self-driving vehicles and autonomous racing. In this setting, the majority of the approaches focus on the trajectory tracking problem. Recent solutions exploit the predictive control paradigm, where the control inputs are retrieved through a real-time trajectory optimization procedure carried out in a receding horizon manner. However, these approaches might not be effective in a farming scenario where the soil terrain is often uneven and, in the case of orchards, the intra-row space may be narrow leaving no space to maneuvering errors. In addition, in this specific scenario, maneuvers are often task-based, meaning that a vehicle is asked to move from a specific pose configuration to another one to perform certain agronomic interventions or to collect data. Therefore, a pose regulation approach seems more adequate in this setting.

In this work, we propose a full navigation stack for a farming robot with Ackermann steering kinematics. This research is motivated by the needs of the European H2020 project PANTHEON where an autonomous farming robot is required to navigate within a hazelnut orchard to collect information and perform agronomic interventions at the resolution of the single tree (see Figure. 1). To achieve this objective, the proposed navigation architecture relies on two major components: i) a local planner and ii) a tailored pose regulation control law. In particular, the local planner generates, in real-time, optimal trajectories by taking into account both the Ackermann kinematics and the obstacles (e.g. trees, rocks, etc.) that might lie, or dynamically appear, along the route. To do so, the planning problem is modeled as a cost-function minimization problem over a finite-time horizon,

*These two authors equally contributed to this work.

Renzo Fabrizio Carpio, Ciro Potena, Jacopo Maiolini, Giovanni Ulivi, and Andrea Gasparri are with Roma Tre University, Rome, Italy, 00146.

Nicolás Bono and Emanuele Garone are with the with the Université Libre de Bruxelles, 1050 Brussels, Belgium.

This work has been supported by the European Commission under the grant agreement number 774571 – Project PANTHEON.

†Corresponding author. E-mail: *gasparri@dia.uniroma3.it*.

[1]`http://www.project-pantheon.eu`

and by re-planning the optimal trajectory in a receding horizon fashion. The outcome of the local planner, i.e., a trajectory described by a sequence of desired poses, is then fed into a non-smooth control law that ensures the convergence toward each of these configurations. This control law, which is inspired by control techniques originally applied to the unicycle kinematics, has been purposely designed *ex novo* to solve the pose regulation problem for vehicles with Ackermann kinematics. A comprehensive theoretical characterization of the convergence properties of the pose regulation controller is also provided by resorting to tools coming from non-smooth theory.

To validate the proposed approach, we report results from an exhaustive set of experiments on both a real and a simulated farming robot with the same kinematical properties. We show that the proposed method guarantees accurate and safe navigation within the target field while keeping the vehicle at a safe distance from the surrounding obstacles. We also provide an open-source C++ implementation of the proposed solution and of the simulated experimental scenario at the following link:

`https://tinyurl.com/s4ofvuz`

A. Related Work

The problem of planning and controlling vehicles with Ackermann steering kinematics has been widely investigated, especially in the context of self-driving cars and outdoor autonomous robots moving in urban environments [12]. Several solutions have been proposed and they can be roughly divided in two categories, non-predictive ([3], [4], [5], [6], [7]) and predictive ([8], [9], [10], [11]).

The formers are not able to plan the motion of the vehicle, therefore it is necessary to compute the trajectory before applying the control law. In Zhang *et al.* [3] the authors propose a sliding mode control law specifically designed for second order differential equation models. The effectiveness of this approach is extensively tested in simulation. A similar non-linear control law for lateral motion of the vehicle is proposed in [4]. In this work, the author presents a lateral control strategy to steer autonomous vehicles using vision. Such a control law takes into account the vehicle velocity to adapt the steering control response, making the control strategy well suited for different speeds.

Franch *et al* [5] proposes a trajectory controller and a trajectory generation algorithm for Ackerman vehicles. To handle the non-linearities of the system model, the trajectory generation problem is formulated in terms of flat outputs, making it possible to apply a dynamic linearization, while a closed-loop control law steers the vehicle along the planned waypoints.

Differently from non-predictive methods, predictive approaches are able to plan the optimal trajectory of the vehicle together with the optimal control inputs by re-planning the trajectory in a receding horizon fashion.

In Liniger *et al.* [10], the authors propose two different control laws. The first employs a path planner and a NMPC for tracking. The second combines both tasks in a single non-linear optimization problem. The proposed method is tested by using a 1:43 scale RC race car, showing high performance and feasibility. The former controller formulation has been extended in [8], where the authors present a full stack for an autonomous race car. In particular, they used a novel vehicle model that fits well the real system also when driving slow. They also include actuator dynamics and tire friction constraints. The system has been tested in different Formula Student competitions. To handle the unmodeled vehicle dynamics, in [9] and [11], the authors proposed a learning-based controller: the nominal vehicle model is constantly improved based on sensorial data gathered in real-time and by employing machine learning methods ([13]). This leads to reductions of the lap time up to 10%.

As mentioned in the introduction, these approaches cannot be used effectively in high-precision agricultural environments. Trajectory planning and control in the agricultural context have specific features that make it quite different from the settings for which the above-mentioned laws have been developed. Some of the most relevant differences are that: (i) the intra-row traversing space is narrow leaving little room to maneuvering errors, and (ii) the soil terrain is usually uneven, leading to unmodeled dynamic behaviors (e.g., wheels skidding or steps). In addition, farming robots must guarantee safe operations and, in particular, avoid harming the plants.

In Astolfi *et al* [14], the authors present a full architecture for an autonomous robot in a vineyard environment. In particular, for the planning part, they exploit open-source navigation packages such as Gmapping, Google's Cartographer, and KartoSLAM. In Bascetta *et al* [15], the authors merge the theory developed in [16] and [17] to develop a Lyapunov-based methodology for trajectory tracking of Ackermann steering kinematics. The proposed control law is tested with real experiments on a commercial all-terrain vehicle showing good tracking performance. An efficient navigation planning for farming robots is presented in [18]. In this work, the authors propose a search-based planner for a robot with adjustable wheel positions. The proposed method is evaluated on simulated and real-world data with a Bosch Bonirob robot.

An alternative manner to achieve autonomous navigation in farmlands is through visual feedbacks coming from one or more cameras. In this regards, English *et al.* [19] proposed a vision-based crop-row following system. A highly related work has been proposed by Sharifi *et al.* [20] that addresses the problem of steering an agricultural robot in orchards. The authors introduce a novel vision-based technique that extracts the desired central path to follow from the captured color images. Experiments show that the proposed method correctly estimates a viable path for the mobile robot. While effective, these systems do not take explicitly into account obstacles and might not guarantee safe and reliable navigation in an orchard.

B. Contributions

The contributions of this work are the following: (i) a complete navigation system (with an open-source implementation) that addresses a very specific problem in nowadays high-precision agricultural robotics, (ii) a novel non-smooth control law with a full theoretical characterization of its convergence property for Ackermann steering kinematics; (iii) an experimental validation of the proposed navigation architecture in a real-world (1:1 scale) precision farming setting.

II. Ackermann Steering Kinematics

In this section, we briefly review the basic concept of the Ackermann Steering Geometry. An Ackermann front-wheel-steering vehicle can be conveniently expressed from a mathematical standpoint in terms of a Bicycle model. In this model, the control inputs are the linear velocity v and the steering angle ϕ. Specifically, the following set of equations can be used to describe the kinematics of a Bicycle model:

$$\begin{aligned} \dot{x}(t) &= v(t)\cos(\theta(t)) \\ \dot{y}(t) &= v(t)\sin(\theta(t)) \\ \dot{\theta}(t) &= \frac{v(t)}{\ell}\tan(\phi(t)) \end{aligned} \tag{1}$$

where $q(t) = [x(t),\, y(t),\, \phi(t)]^T$ denotes the state vector at time t with $p(t) = [x(t),\, y(t)]^T$ representing the position of the robot in the Cartesian plane at time t w.r.t. the global reference frame $\Sigma_g = (x_g,\, y_g,\, \theta_g)$, and $\theta(t)$ representing its orientation with respect to the x-axis of global reference frame Σ_g at time t, $u(t) = [v(t),\, \phi(t)]^T$ denotes the input vector at time t with $v(t)$ and $\phi(t)$ representing the linear velocity and the steering angle, respectively, at time (t), and finally ℓ denotes the wheelbase, i.e., the distance between the front and rear axles. In the following we will refer to the Bicycle kinematics in a compact form as

$$\dot{q}(t) = f(q(t), u(t)), \tag{2}$$

with $f(\cdot,\cdot)$ the system dynamics defined according to eq. (1). In the sequel, the time-dependence will be omitted if not strictly required for the sake of readability. In this work we assume that the vehicle reference frame $\Sigma_r = (x,\, y,\, \theta)$ is going to be located at the center of the rear axle (see Fig. 2).

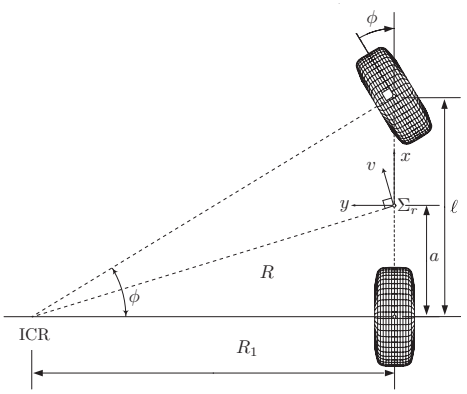

Fig. 2: Equivalent Bicycle model for a front-wheel-steering vehicle.

III. Trajectory Planning with Obstacles

The goal of the trajectory planner is to generate optimal trajectories over a time horizon T from any given initial pose and velocity that fulfill the following requirements:

i) the trajectory steers the robot to a given target pose;
ii) the trajectory satisfies the Ackermann kinematics constraints and is collision-free;
iv) the trajectory is generated in a receding horizon fashion and in real-time.

We reiterate that the latter aspect is essential in the farming context since re-planning in real-time allows the robot to react in time to avoid dynamical obstacles and to handle perception uncertainties as well. The whole problem can be formulated as a cost-function minimization problem over a finite time horizon T. To this end, let $o(t) = [o_{1(t)}^T,\, \ldots,\, o_{\mathfrak{o}(t)^T}]^T$ be the state vector of the surrounding obstacles where $o_{i(t)} = [x_{i(t)},\, y_{i(t)}]^T$ represents the state of the i-th obstacle considered at time t described by its coordinates, and $\mathfrak{o}(t)$ is the number of detected obstacles at time t. Note that, a time-varying index mapping $i(t)$ is used in the mathematical formulation to model the fact that the i-th obstacle considered for planning the trajectory at time t depends on the actual robot position at that time t.

In addition, let $q(t)$ and $u(t)$ be the state and the input vectors of a robot at time t, respectively, with dynamics defined according to eq. (1) and expressed in a compact form according to (2).

The following optimization problem formulation is considered for the navigation of an Ackemann steering vehicle in a high-precision farming environment filled with obstacles

$$\begin{aligned} \underset{q(t)}{\text{minimize}} \quad & h(q(t_f)) + \int_{t_0}^{t_f} c_n(q(t), u(t)) + c_o(q(t), o(t))\, dt \\ \text{subject to} \quad & \dot{q(t)} = f(q(t), u(t)), \\ & r(q(t_0)) = 0, \\ & l(q(t), i(t)) \leq 0 \quad \forall\, i \in \{1,\, \ldots,\, \mathfrak{o}(t)\} \end{aligned} \tag{3}$$

where $h(q(t_f))$ describes the cost in the final state, $c_n(q(t), u(t))$ describes the navigation cost, $c_o(q(t), o(t))$ encodes the obstacle avoidance cost, $r(q(t), o(t))$ and $l(q(t), i(t))$ encode the set of equality and inequality constraints to satisfy along the trajectory, and $T = t_f - t_0$ represents the time horizon in which we want to find the solution. In particular, the cost function and the set of equality and inequality constraints are defined as

$$\begin{aligned} h(q(t_f)) &= (q_d - q(t_f))^T\, Q_f\, (q_d - q(t_f)) \\ c_o(q(t), o(t)) &= \sum_{i=1}^{\mathfrak{o}(t)} \left((p(t) - o_i(t))^T\, W_i\, (p(t) - o_i(t))\right)^{-1} \\ c_n(q(t), u(t)) &= u(t)^T\, R\, u(t) + (q(t) - q_f)^T\, Q\, (q(t) - q_f) \\ l(q(t), i(t)) &= d_{\min}^T\, d_{\min} - (p(t) - o_i(t))^T\, (p(t) - o_i(t)) \\ r(q(t_0)) &= (q_i - q(t_0))^T\, (q_i - q(t_0)) \end{aligned} \tag{4}$$

with q_d representing the goal state, $d_{\min}$ a safety distance from an obstacle, Q_f, W_i, Q, R weighting matrices, q_i and q_f are the initial pose and desired final pose, respectively.

In addition, it should be noticed that in order to comply with the real-time requirements of the application and to increase the flexibility of the approach, obstacles should be dynamically added and removed according to a proper detection and selection approach, e.g., only detected obstacles within a certain radius from the current robot location are considered for the planning. An obstacle detection algorithm purposely designed for the precision farming setting considered within the project PANTHEON, i.e., a hazelnut orchard, is currently being developed and will be the object of future work.

IV. Pose Regulation Problem Formulation

In this section, we focus on the pose regulation problem for the Bicycle kinematic model given in eq. (1), i.e., the problem of designing a control law which is capable of steering the vehicle from any initial configuration $q_i = [x_i,\, y_i,\, \phi_i]^T$ to any desired final configuration $q_d = [x_d,\, y_d,\, \phi_d]^T$. To facilitate the analysis, it is convenient to formulate the regulation problem in polar coordinates $\hat{q} = [\rho,\, \gamma,\, \delta]^T$. For a given pose $q = [x,\, y,\, \phi]^T$ this coordinate transformation is defined as

$$\begin{aligned} \rho &= \sqrt{(x_d - x)^2 + (y_d - y)^2} \\ \gamma &= \operatorname{atan2}((y_d - y), (x_d - x)) - \theta + \pi \\ \delta &= \gamma - (\theta_d - \theta) \end{aligned} \tag{5}$$

where ρ is the distance between the point $(x,\, y)$ of the Ackermann reference frame Σ_r and the origin of the Cartesian plane, γ the angle between the Cartesian error vector $\mathbf{e}_p = p_d - p$ and the sagittal axis of the vehicle, and δ the angle between the same vector and the x-axis.

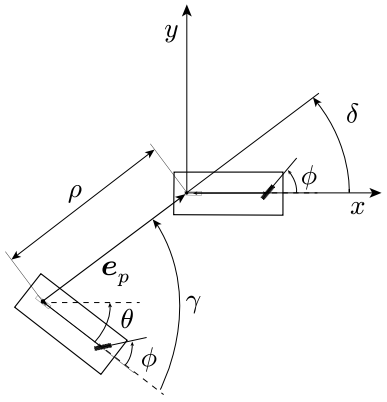

Fig. 3: Definition of polar coordinates for the Bicycle kinematics

In these polar coordinates $\hat{q} = [\rho,\, \gamma,\, \delta]^T$, the kinematic model of the Ackermann steering is

$$\begin{aligned} \dot{\rho} &= -v\cos(\gamma) \\ \dot{\gamma} &= \frac{\sin(\gamma)}{\rho} v - \frac{v}{\ell}\tan(\phi) \\ \dot{\delta} &= \frac{\sin(\gamma)}{\rho} v \end{aligned} \tag{6}$$

note that the input vector field associated with $v = 0$ is singular for $\rho = 0$. In the sequel, with no lack of generality we assume that the desired configuration is the origin, that is $q_d = [0,\, 0,\, 0]^T$. We now provide our main result concerning the pose regulation problem for vehicles with Ackermann steering kinematics.

Theorem 1. *Let a ground vehicle with Ackermann kinematics described by eq.* (1)*. Assume that the initial configuration $q_i = [x_i,\, y_i,\, \theta_i]^T$ of the robot is such that*

$$\rho_i \triangleq \sqrt{x_i^2 + y_i^2} \neq 0 \tag{7}$$

Then, the pose regulation problem can be solved using the following non-smooth control law

$$\begin{aligned} v &= k_1\, \rho\, \mathrm{SIGN}(\cos(\gamma)) \\ \phi &= \operatorname{acos}\left(\frac{1}{\sqrt{\kappa^2 + 1}}\right) \mathrm{SIGN}(\kappa) \end{aligned} \tag{8}$$

with $\kappa = \dfrac{\ell}{v}\kappa'$ where κ' is defined as

$$\kappa' = \left(k_2\gamma + k_1 \frac{\sin(\gamma)\ \mathrm{SIGN}(\cos(\gamma))}{\gamma}(\gamma + k_3\, \delta)\right) \tag{9}$$

with gains $k_3 > 0$, $k_2 > 0$, and $k_1 > 0$, and with the sign operator $\mathrm{SIGN}(\cdot)$ defined as

$$\mathrm{SIGN}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{10}$$

Proof. Note that, being the control law discontinuous, some tools coming from the non-smooth analysis are required, see *e.g* [21]–[23] and references therein for a comprehensive overview of the topic. To prove stability, the following generalized Lyapunov function is considered

$$V = V_1 + V_2 + V_3 = \left(\frac{1}{2}\rho^2 + \frac{1}{2}\gamma^2 + \frac{1}{2}k_3\delta^2\right) \tag{11}$$

the generalized Lyapunov time-derivate along the closed-loop system trajectories is

$$\dot{\tilde{V}} = \dot{V}_1 + \dot{V}_2 + \dot{V}_3 = \rho\,\dot{\rho} + \gamma\,\dot{\gamma} + k_3\delta\,\dot{\delta} \tag{12}$$

Let us now analyze each of the three terms $\dot{V}_1$, $\dot{V}_2$ and $\dot{V}_3$ of the generalized Lyapunov time-derivative $\dot{\tilde{V}}$. For what concerns the term $\dot{V}_1$, we have

$$\begin{aligned} \dot{V}_1 = \rho\,\dot{\rho} &= \rho\ (-K[v]\cos(\gamma)) \\ &= \rho\ (-(K[k_1\,\rho\ \mathrm{SIGN}(\cos(\gamma))])\cos(\gamma)) \\ &= -k_1\rho^2\cos(\gamma)K[\mathrm{SIGN}(\cos(\gamma))] \\ &= -k_1\rho^2\,|\cos(\gamma)| \end{aligned} \tag{13}$$

Similarly, the following holds for the term $\dot{V}_2$

$$\begin{aligned}\dot{V}_2 &= \gamma\,\dot{\gamma} = \gamma\left(\frac{\sin(\gamma)}{\rho}K[v] - \frac{K[v]}{\ell}\tan(K[\phi])\right)\\ &= \gamma\left(\frac{\sin(\gamma)}{\rho}K[v] + \right.\\ &\quad \left. - \left(k_2\gamma + k_1\frac{\sin(\gamma)\,K[\mathrm{SIGN}(\cos(\gamma))]}{\gamma}(\gamma + k_3\,\delta)\right)\right)\\ &= \gamma\frac{\sin(\gamma)}{\rho}K[v] - k_2\gamma^2 - \frac{\sin(\gamma)}{\rho}K[v]\gamma - k_3\,\delta\,\frac{\sin(\gamma)}{\rho}K[v]\\ &= -k_2\gamma^2 - k_3\,\delta\left(\frac{\sin(\gamma)}{\rho}K[v]\right)\end{aligned} \tag{14}$$

where the relationship $\mathrm{atan}(x) = \mathrm{acos}\left(\frac{1}{\sqrt{x^2+1}}\right)\mathrm{SIGN}(x)$ along with properties coming from the calculus for computing the Filipov's differential inclusions [21], [23], were used to perform the following substitution

$$\begin{aligned}\frac{K[v]}{\ell}\tan(K[\phi]) &= \frac{K[v]}{\ell}\tan(K[\mathrm{atan}(\kappa)])\\ &= \frac{K[v]}{\ell}\frac{\ell}{K[v]}K[\kappa'] = K[\kappa']\end{aligned} \tag{15}$$

where it should be noticed that eq. (15) holds true as long as $v \neq 0$, that is $\rho \neq 0$. This further motivates the assumption on the initial condition given in eq. (7). Finally, the following holds for the term $\dot{V}_3$

$$\dot{V}_3 = k_3\delta\,\dot{\delta} = k_3\delta\left(\frac{\sin(\gamma)}{\rho}K[v]\right). \tag{16}$$

By putting together the expression for the three terms we obtain the following expression for the generalized Lyapunov time-derivative $\dot{\tilde{V}}$

$$\begin{aligned}\dot{\tilde{V}} &= \dot{V}_1 + \dot{V}_2 + \dot{V}_3\\ &= -k_1\rho^2\,|\cos(\gamma)| - k_2\gamma^2 + \\ &\quad - k_3\,\delta\left(\frac{\sin(\gamma)}{\rho}K[v]\right) + k_3\delta\left(\frac{\sin(\gamma)}{\rho}K[v]\right)\\ &= -k_1\rho^2|\cos(\gamma)| - k_2\gamma^2\end{aligned} \tag{17}$$

which proves that the time derivative along the closed-loop system trajectories is negative semi-definite.

At this point, by resorting to the generalized LaSalle invariance theorem [22], let us define the set $\mathcal{S}$, which represents the set of states for which the generalized Lyapunov derivative is zero in polar coordinates as

$$\begin{aligned}\mathcal{S} &= \{\hat{q} \in \mathbb{R}\times\mathbb{S}^2 \;:\; \dot{\tilde{V}} = 0\}\\ &= \{\hat{q} \in \mathbb{R}\times\mathbb{S}^2 \;:\; \rho = \gamma = 0\}\end{aligned} \tag{18}$$

and let us compute the largest invariant set $\mathcal{M}$. To this end, we are going to resort to a special case of viability result in the study of differential inclusions which states that for all points $\hat{q} \in \mathcal{M}$, the following must hold true

$$T_{\mathcal{S}}(\hat{q}) \cap K[f](\hat{q}) \neq \emptyset, \quad \forall\,\hat{q} \in \mathcal{M} \tag{19}$$

where $T_{\mathcal{S}}(\hat{q})$ the contingent cone to $\mathcal{S}$ at $\hat{q}$ and $K[f](\hat{q})$ is the Filippov set with $f(\cdot)$ the stacked vector representation of the Ackermann Steering kinematics in polar coordinates. Let us now compute $T_{\mathcal{S}}$, that is the contingent cone to $\mathcal{S}$ at $\hat{q} \in \mathcal{M}$, as

$$T_{\mathcal{S}}(\hat{q}) = \mathrm{span}\left[0,\ 0,\ 1\right]^T \tag{20}$$

and let us compute the Filippov set $K[f](\hat{q})$ as

$$K[f](\hat{q}) = \begin{bmatrix} 0 \\ k_1\,k_3\,[-1,\ 1]\,\delta \\ 0 \end{bmatrix} \tag{21}$$

At this point, it can be noticed that $T_{\mathcal{S}}(\hat{q}) \cap K[f](\hat{q}) \neq \emptyset$ holds true only at the intersection, implying that the largest set $\mathcal{M}$ must contain only the origin, that is $\hat{q} = [0,\ 0,\ 0]^T$. This proves that the proposed control law solves the regulation problem for the Ackermann Steering kinematics under the condition that the initial configuration q_i satisfies eq. (7). □

A few remarks are now in order:

i) The assumption on the initial configuration q_i is a direct consequence of the control law definition given in eq. (8), which implies that the if $\rho = 0$, then the velocity v is null, thus preventing any vehicle motion to occur. This can be intuitively explained by the fact that if the initial pose is already at the desired position but with a different orientation from the desired one, being the Ackermann Steering vehicle not capable of rotating along its vertical axis, a (complex) maneuver would be required to reorient it, which would necessarily require ρ to increase, thus going against the negative semi-definiteness of the time derivative of the generalized Lyapunov function. We point out that should the problem given in eq. (7) occur, the planner would compute a feasible trajectory through a series of successive way-points.

ii) The proposed control law is discontinuous at the origin of the configuration space. Indeed, it is well-known that any feedback law that can regulate the posture of a vehicle with non-holonomic constraints must be necessarily discontinuous with respect to the state and/or time-varying (Brockett Theorem) [24].

V. Numerical and Experimental Validation

In this section a numerical and experimental validation is provided to demonstrate the effectiveness of the proposed navigation architecture in precision farming settings.

A. Simulated Navigation

In this section, we numerically demonstrate the capabilities of the proposed navigation architecture in a simulated hazelnut farming scenario. The experiments have been carried out through GAZEBO, a robotic simulation toolbox. The non-linear planning problem is set up with the ACADO toolbox and solved by the qpOASES solver [25]. By using the ACADO code generation tool, the problem is exported in a highly efficient c-code that we integrated within a ROS (Robot Operating System) node. We set the discretization step to be $dt = 0.5\ s$ with a time horizon $T = 15s$. The

planned trajectory, which is described by a discrete sequence of poses, is then fed into the control law node as a sequence of waypoints, while the output Ackermann control inputs are mapped to the wheel motors by a simulated low-level controller running on-board.

The experiment consists of planning trajectories from a given initial pose q_i to a sequence of desired poses $\mathfrak{q}_d = (q_{d_1}, \ldots, q_{d_M})$. For each pair of poses, the local planner will compute an obstacle-free trajectory described by a sequence of successive waypoints. To ensure smooth navigation with no stop-and-go behavior, a new waypoint is fed into the control law when the distance from the current waypoint has become smaller than a given threshold (set to 0.2 meters in our simulations). For the numerical validation, the control gains are tuned as $(k_1, k_2, k_3) = (1, 6, 3)$. Indeed, this choice has numerically proven the robot to satisfactory follow the obstacle-free trajectory. It should be noticed that for the numerical validation, a custom-built GAZEBO model of the SHERPA robotic platform has been developed. Briefly, this model is equipped with the same sensor setup as the real platform and moves according to the Ackermann steering kinematics.

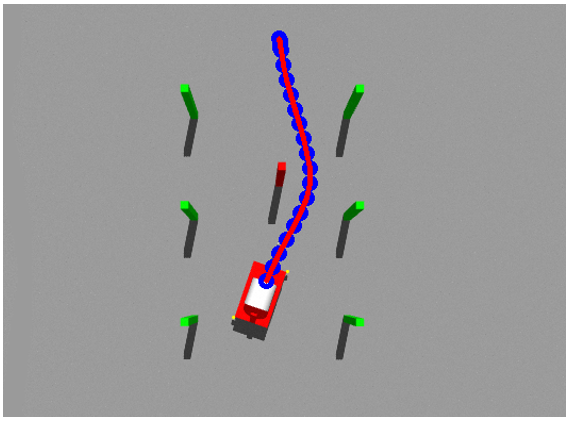

Fig. 4: Examples of a planned trajectory in a simulated scenario filled with static obstacles (green) and a dynamic obstacle (red). The green obstacles are arranged to resemble the planting layout of hazelnut trees as in the real field.

An example of a planned trajectory in the presence of a dynamic obstacle is depicted in Figure 4. Tab. I reports trajectory statistics from exhaustive simulations carried out with different setups. Results are sorted by increasing trajectory planning difficulties, either in terms of path length or presence of dynamic obstacles. The success rate is almost 100%, while it slightly decreases when adding "last-meter" dynamic obstacles along the planned route, which may leave no space to corrective maneuver. Nevertheless, the final position and angle errors remain almost unchanged, proving the effectiveness of the control law.

B. Experimental Setup

Real experiments have been carried out within one of the real-word hazelnut orchards of the PANTHEON project, within the "Azienda Agricola Vignola", a farm located in the municipality of Caprarola, in the province of Viterbo, Italy. In particular, the experiments have been carried out in a young orchard (cultivar Nocchione treated as multi stemmed bushes) with a 4.5 m x 3.0 m layout.

Fig. 5: SHERPA HL robotic platform prototype R-A.

Figure 5 shows the ground vehicle prototype, namely SHERPA HL robotic platform R-A, which has been used for the experimental validation. The SHERPA HL R-A platform is mechanically designed to operate according to the Ackermann Steering kinematics. The platform is equipped with a Trimble MB-Two GNSS Receiver with GPS-RTK capabilities, a SBG Ellipse2-E IMU with an integrated compass, two Sick S300 Safety Laser Scanners, and a Velodyne VLP-16 Puck LITE 3D LIDAR. The platform also mounts an Intel NUC NF697 with an Intel Core i7-8705G Processor where the low-level software and the proposed control law run. The planner runs on an independent laptop with an Intel i7-8750H processor and the optimal trajectories are sent to the robot through a Wi-Fi connection. The non-linear planner and the control law are set up as described in Sec. V-A.

C. Pose Regulation

In this section, we experimentally demonstrate the effectiveness of the proposed control law in a real-world high-precision farming scenario, i.e., the hazelnut orchard described in Section V-B. In all the regulation experiments the control gains and the desired state are tuned as $(k_1, k_2, k_3) = (0.3, 1.5, 3)$ and $q_d = (q_{x,d}, q_{y,d}, q_{z,d})$, respectively. The pose regulation task is performed with the real robotic platform and for 8 different starting poses q_i derived by sampling a set of positions over a circle with a radius of 6 meters. The trajectories performed by the robot are reported in Figure 6, where it can be noticed that the robot successfully reaches the desired pose from all the initial conditions and shows a symmetrical behavior. Note that, to easy the visualization of this experimental validation the origin of the reference frame is relocated at the desired position p_i of the robot.

D. Navigation

In this section, we experimentally demonstrate the capabilities of the proposed navigation architecture in a real-world

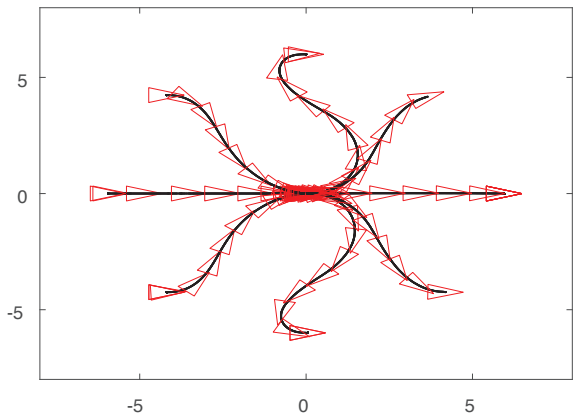

Fig. 6: Trajectories performed by the robot to reach the desired state in the field 16 of the hazelnut orchard. In red, the orientation of the vehicle sampled proportionally to the vehicle velocity.

Dynamic Obstacle	Trajectory lenght $[m]$	Failure rate [%]	v $[\frac{m}{s}]$	ϕ $[rad]$	Min. Obst. Distance	Position err. $[m]$	Angle err. $[rad]$
	5	.02	.34	.75	1.34	.11	.08
	7	.03	.33	.81	1.41	.09	.05
	9	.03	.36	.78	1.36	.07	.07
✓	9	4.06	.4	.98	1.19	.13	.12
✓	12	4.75	.43	1.04	1.25	.11	.09
✓	13	3.97	.41	1.05	1.23	.12	.11

TABLE I: Simulations: trajectory statistical analysis.

high-precision farming scenario, i.e., the hazelnut orchard described in Section V-B. In particular, we demand the robot to move from an initial pose q_i to the desired pose q_d on a different orchard row. Thus, forcing the robot to plan a safe trajectory among trees. Three examples of trajectories performed by the real robotic platform are illustrated in Figure 7, where in the first case the robot moves forward while in the second and third cases the robot moves backward. It can be noticed that in all three cases, the robot successfully reaches the desired goal by passing in the intermediate point between two adjacent trees when switching from one row to another. In the third case, we spawn a virtual obstacle on the planned route. The robot is then forced to re-plan a different trajectory (depicted in red) to reach the desired goal. As reported in Table II, the robot always manages to reach the goal with a small error. The only exception is the third trajectory where, having delayed the switching maneuver between the rows, the robot does not find enough room to align its yaw with the desired one. Note that, to easy the visualization of these experimental results, in this case, the origin of the reference frame is relocated at the initial position p_i of the robot. In addition it should be further noticed that the second and the third trajectories depicted in Figure 7 are also reported in the attached multimedia material together with a real footage of the robotic platform.

E. Computational Time

In this section, we numerically validate the real-time planning capability of the proposed navigation architecture. To this end, it should be noticed that the computational cost is not constant. Indeed, it may vary according to the specific

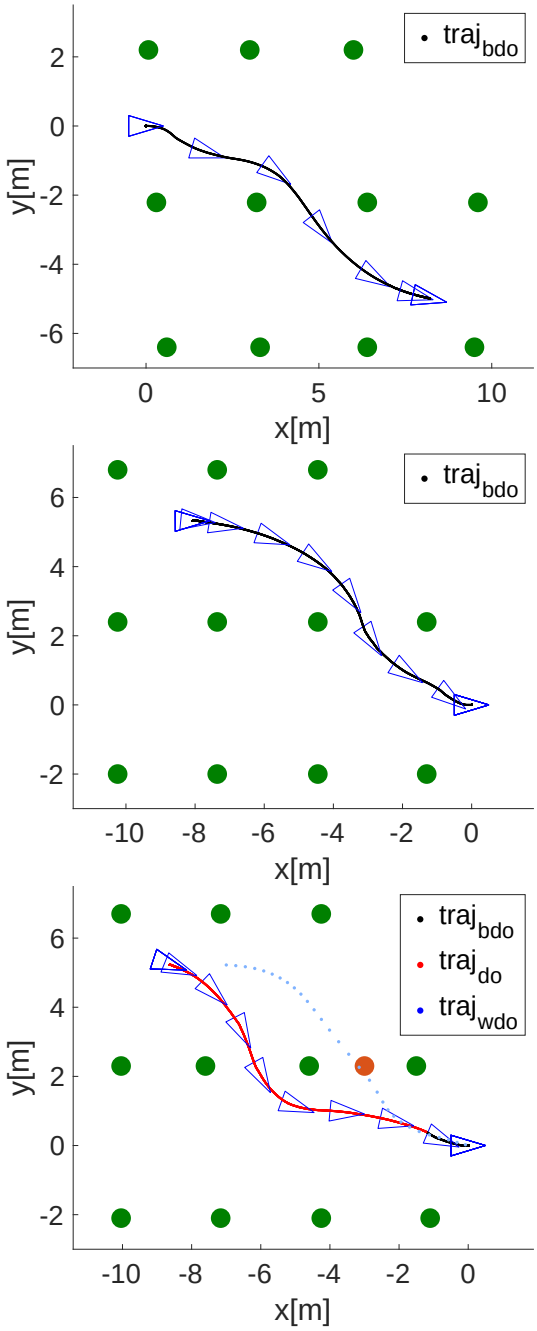

Fig. 7: Trajectories performed by the robotic platform in a real hazelnut orchard (plants are represented by the green circles). From top to bottom, a front-wise maneuver, a rear-wise maneuver and a maneuver with a dynamic obstacle (red dot). The second and third trajectories are also reported in the attached multimedia material together with a real footage of the robotic platform.

Dynamic Obstacle	Trajectory lenght $[m]$	v $[\frac{m}{s}]$	ϕ $[rad]$	Min. Obst. Distance	Position err. $[m]$	Angle err. $[rad]$
	9.5	.28	.30	1.01	.09	.12
	9.5	.31	.31	1.05	.08	.09
✓	9.5	.29	.41	1.14	.11	.19

TABLE II: Experiments: trajectory statistical analysis.

operating mode of the planner. For this reason, we distinguish among three different planning modes:

1) Initial mode (blue): which occurs when the vehicle start planning a new trajectory;
2) Steady mode (green): which occurs when the vehicle is already moving toward the target pose;
3) Emergency mode (red): which occurs when a dynamic obstacle suddenly appears along the optimal trajectory.

Figure 8 reports the average computational costs for all the planning phases. The average computational cost is constantly lower than $0.033s$. As expected, the steady-planning phase turns out to be the cheapest one since the robot moves slowly and the trajectory to re-plan is close to the previous one. Conversely, the emergency re-planning phase is the most expensive and variable, as the trajectory to re-plan is usually far from the previous one since the dynamic obstacle might drastically change the optimal path.

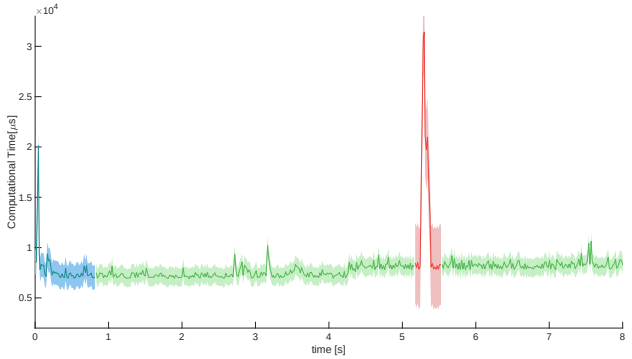

Fig. 8: Average computational time plot across the planning phases over a time horizon of 8 seconds: the (i) planning phase in blue, (ii) the steady-planning phase in green, and (iii) the emergency re-planning phase in red. The shaded areas represent the variance of the average computational time.

VI. Conclusions

In this letter, we present an effective navigation architecture for Ackermann vehicles in orchard environments which leverages a synergy between an optimal planner and a novel pose regulation controller. The former generates, in real-time, a sequence of successive poses that steers the robot towards its desired location while avoiding all the surrounding obstacles. The pose regulation ensures the convergence toward each of these poses. A numerical and experimental validation has been carried out to demonstrate the effectiveness of the proposed navigation architecture, showing reliability even in the case of dynamic obstacles suddenly appearing along the planned route. Future work will be mostly focused to enrich this navigation architecture, e.g., by developing the obstacle detection module responsible for the "on-the-fly" detection of the obstacles to be fed into the local planner.

References

[1] W. Lee, V. Alchanatis, C. Yang, M. Hirafuji *et al.*, "Sensing technologies for precision specialty crop production," *Computers and Electronics in Agriculture*, vol. 74, no. 1, pp. 2 – 33, 2010.

[2] T. Duckett, S. Pearson, S. Blackmore, B. Grieve, and M. Smith, "White paper - agricultural robotics: The future of robotic agriculture," 2018.

[3] J. Zhang, S. Xu, and A. Rachid, "Sliding mode lateral motion control for automatic steering of vehicles," 09 2001, pp. 3636–3641.

[4] M. A. Sotelo, "Lateral control strategy for autonomous steering of ackerman-like vehicles," *Robotics and Autonomous Systems*, vol. 45, no. 3, pp. 223 – 233, 2003.

[5] J. Franch and J. M. Rodriguez-Fortun, "Control and trajectory generation of an ackerman vehicle by dynamic linearization," in *2009 European Control Conference (ECC)*, Aug 2009, pp. 4937–4942.

[6] K. Kritayakirana and J. Christian Gerdes, "Autonomous vehicle control at the limits of handling," *Int. J. of Vehicle Autonomous Systems*, vol. 10, pp. 271 – 296, 01 2012.

[7] M. Klomp, K. Olsson, and C. Sandberg, "Non-linear steering control for limit handling conditions using preview path curvature," *Int. J. of Vehicle Autonomous Systems*, vol. 12, pp. 266 – 283, 01 2014.

[8] J. Kabzan, M. de la Iglesia Valls, V. Reijgwart, H. F. C. Hendrikx *et al.*, "AMZ driverless: The full autonomous racing system," *CoRR*, vol. abs/1905.05150, 2019.

[9] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3363–3370, Oct 2019.

[10] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale rc cars," *Optimal Control Applications and Methods*, vol. 36, no. 5, pp. 628–647, 2015.

[11] U. Rosolia, A. Carvalho, and F. Borrelli, "Autonomous racing using learning model predictive control," in *2017 American Control Conference (ACC)*, May 2017, pp. 5115–5120.

[12] B. Paden, M. p, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, March 2016.

[13] A. Latif, A. Rasheed, U. Sajid, J. Ahmed *et al.*, "Content-based image retrieval and feature extraction: A comprehensive review," *Mathematical Problems in Engineering*, vol. 2019, 08 2019.

[14] P. Astolfi, A. Gabrielli, L. Bascetta, and M. Matteucci, "Vineyard autonomous navigation in the echord++ grape experiment," vol. 51, pp. 704–709, 01 2018.

[15] L. Bascetta, D. A. Cucci, and M. Matteucci, "Kinematic trajectory tracking controller for an all-terrain ackermann steering vehicle," *IFAC-PapersOnLine*, vol. 49, no. 15, pp. 13 – 18, 2016, 9th IFAC Symposium on Intelligent Autonomous Vehicles IAV 2016.

[16] G. Indiveri, "Kinematic time-invariant control of a 2d nonholonomic vehicle," in *Proceedings of the 38th IEEE Conference on Decision and Control*, vol. 3, Dec 1999, pp. 2112–2117 vol.3.

[17] M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino, "Closed loop steering of unicycle like vehicles via lyapunov techniques," *IEEE Robotics Automation Magazine*, vol. 2, no. 1, pp. 27–35, March 1995.

[18] F. Fleckenstein, C. Dornhege, and W. Burgard, "Efficient path planning for mobile robots with adjustable wheel positions," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[19] A. English, P. Ross, D. Ball, and P. Corke, "Vision based guidance for robot navigation in agriculture," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1693–1698.

[20] M. Sharifi and XiaoQi Chen, "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)*, Feb 2015, pp. 251–255.

[21] B. E. Paden and S. S. Sastry, "A calculus for computing filippov's differential inclusion with application to the variable structure control of robot manipulators," in *1986 25th IEEE Conference on Decision and Control*, Dec 1986, pp. 578–582.

[22] D. Shevitz and B. Paden, "Lyapunov stability theory of nonsmooth systems," *IEEE Transactions on Automatic Control*, vol. 39, no. 9, pp. 1910–1914, Sep. 1994.

[23] R. K. Williams, A. Gasparri, G. Ulivi, and G. S. Sukhatme, "Generalized topology control for nonholonomic teams with discontinuous interactions," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 994–1001, Aug 2017.

[24] R. W. Brockett, "Asymptotic stability and feedback stabilization," in *Differential Geometric Control Theory*, R. S. M. R. W. Brockett and H. J. Sussmann, Eds. Boston: Birkhauser, 1983, pp. 181–191.

[25] J. Ferreau, "An online active set strategy for fast solution of parametric quadratic programs with applications to predictive engine control," Ph.D. dissertation, 01 2006.