# An optimization based planner for autonomous navigation in vineyards

**Sara Furioli** [*] **Simone Specchia** [*] **Matteo Corno** [*] **Sergio Savaresi** [*]

[*] *Dipartimento di Elettronica Informazione e Bioingegneria, Politecnico di Milano, 20133 Milan, Italy (e-mail: {name.surname}@polimi.it).*

**Abstract:** Autonomous driving systems found their first applications in the agricultural field, being a way to ease personnel of repetitive jobs and increase precision. Performing operations like harvesting or pruning requires high positioning accuracy, especially in structured environments like vineyards and orchards. In these contexts, the global reference path is dictated by the agricultural procedure to perform. The continuously-changing vegetation and reduced maneuvering space create the need to re-plan the vehicle route with respect to the global reference. Hence, the importance of local planning. This paper proposes a local planning strategy with the objective to follow a park-to-park global path while avoiding obstacles. We formulate the local planning task as a constrained optimization problem. The resulting local plans are not constrained in shape, thus guaranteeing planning freedom, and manage obstacle avoidance in an innovative way. The collision area is precisely determined taking both the vehicle and the obstacles dimension into account, and considering the vehicle approach direction. The proposed system is tested in simulation, where its performance are compared with a benchmark planner. An experimental campaign validates the local planner with satisfactory results.

*Keywords:* Agriculture, Planning, Autonomous Vehicles

## 1. INTRODUCTION

The agricultural field is one of the main forces driving the progress of Advanced Driving Assistance Systems (ADAS) and autonomous driving systems. Performing sensitive procedures like harvesting or pruning requires high positioning accuracy, especially in complex contexts as vineyards and orchards. ADAS for open-field applications are a commercial reality (see Surbrook and Gerrish (1983)), used to carry on repetitive operations such as seeding or plowing. These systems rely on Global Navigation Satellite System (GNSS) technology with Real-Time Kinematics (RTK) correction to obtain vehicle localization precision in the range of a few centimeters (as in Dong et al. (2011), Lenain et al. (2004), Guo et al. (2018)). They function under the hypothesis that the operator is always ready to intervene in case of emergency, as the vehicles are not equipped with exteroceptive sensors to detect obstacles. The complexity rises for vineyard applications, where the vehicle needs to navigate a dynamic environment with numerous obstacles and reduced maneuvering space. Furthermore, GNSS-based systems lack robustness in vineyards and orchards as the RTK correction signal degrades in presence of occlusions and thick vegetation. The most common approach to autonomous driving requires the definition of a global reference path and a local one. In vineyard applications, the global reference path is dictated by the vineyard configuration and the type of procedure the operator must perform. The continuously-changing vegetation and reduced maneuvering space create the need to re-plan the vehicle route with respect to the target path. Hence, the importance of local planning, focus of this paper, which still represents a challenging problem in the literature with space for investigation and possible improvements.

Local planning in structured contexts like vineyards and orchards is often done using only the information provided by vision sensors (*i.e.*, cameras, as in Yun et al. (2018), Subramanian et al. (2006)), or distance sensors like LiDARs (see Barawid Jr et al. (2007), Hiremath et al. (2014)) and ultrasonic sensors (as in our previous work, Corno et al. (2021)). These row-following systems aim at driving the vehicle while it is between crop rows, but struggle outside the rows where it is difficult to locate the vehicle with respect to its surrounding. Expectedly, they cannot autonomously track a park-to-park global reference. To this aim, the exploitation of GNSS sensors is necessary, but not sufficient. The Potential Field Method (PFM) is commonly adopted in mobile robotics to combine local row-following and global path tracking (see Koren et al. (1991), Borenstein and Koren (1990)). PFM cretes an attraction region around the target path, and a repulsion region where obstacles are detected. Astolfi et al. (2018) show an example of its application for vineyard navigation: the authors use the open-source implementation of ROS Navigation Stack[1]. Despite the promising field tests results, they encounter difficulties in tuning the obstacles cost-map: large obstacle inflation is needed to avoid generating paths that intersect the vines, but causes sharp avoidance of small obstacles such as high

[1] ROS Navigation Stack. `http://wiki.ros.org/navigation`.

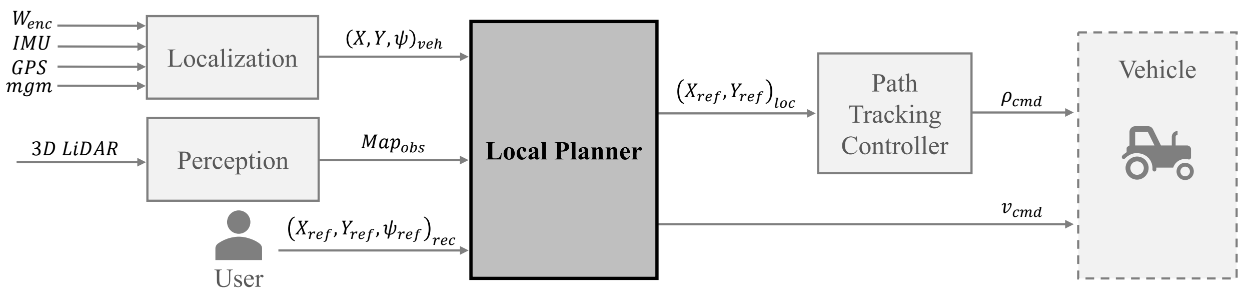

Fig. 1. Block scheme of the developed architecture.

grass. The Navigation Stack employs the Dynamic Window Approach (DWA), which combines path tracking and obstacle avoidance. DWA planner generates a set of trajectories with constant longitudinal and angular speed. Each trajectory is assigned a cost depending on its closeness to the global reference and the presence of obstacles; the one with minimum cost is chosen. Obstacles are inflated to take into account that the trajectory is planned for the vehicle Center Of Gravity (COG). In our experience, deploying ROS Navigation Stack for vineyard navigation poses two problems. The first one is related to the way it generates trajectories, which are necessarily circular, thus limiting the planning possibilities. The second one regards the way it manages collisions through the inflation radius: the vehicle direction while approaching the obstacle is not taken into account and obstacles are enlarged of the same radius along all directions. Finding a trade-off between avoiding obstacles and being able to drive between close vines proves to be challenging.

We propose a local planning strategy which has the objective of navigating a vineyard, following a park-to-park global reference path, and adapt its route in presence of obstacles. The planner receives as input the global reference path, the estimated vehicle position, and a map of the vehicle surrounding, and produces an optimized local plan, which is in turn provided to a path tracking controller. This paper main contributions lie in the optimization problem generating the local plans, particularly in the design of the cost function, and in the decision maker that selects the output plan.

- Differently than DWA, we do not require $(V, \omega)$-constant trajectories: the vehicle reference speed can be set by the user based on the ongoing agricultural procedure, while the steering action is freely optimized to track the global path and avoid obstacles. The resulting local plans are not constrained in shape, except by the vehicle dynamics and actuation limits.
- We propose a new metric to define the collision area as function of both the obstacle and the vehicle size. To do so, the vehicle approach direction is taken into account. As a result, the collision area is precisely determined, hence the planner can be less conservative, which can be crucial in a context where the tractor dimension is non negligible with respect to the environment and the vehicle is forced to brush against the crop rows.
- We design our algorithm to run in real-time. To this aim, the optimization is stopped and forced to produce a local plan after a time interval, resulting in a possibly sub-optimal output local plan. Hence the introduction of a decision maker that compares each new local plan with the old one and selects the one with minimum cost.

The proposed system is able to accurately track the global reference path, and to simultaneously brush against the crop rows when needed but avoid obstacles on the global path outside the vineyard.

The rest of the paper is structured as follows: Section 2 outlines the complete system architecture; Section 3 provides an overview of the auxiliary modules that produce the inputs needed by the local planner and elaborate its outputs. Section 4 details the local planner working principle and the speed planner that monitors the output path and accordingly produces a reference speed. Section 5 presents the planning validation, both in simulation and with experimental data; Section 6 draws some conclusions.

## 2. SYSTEM ARCHITECTURE

The entire system architecture is presented in Fig. 1, where the module object of this paper is highlighted. The localization and perception algorithms exploit proprioceptive and exteroceptive sensors information to produce an estimate of the vehicle position and heading and a map of the surrounding obstacles. The latters feed the local planner, together with the global reference path selected by the user. The planner produces a local path that starts in the current vehicle position and aims at bringing (or keeping) the vehicle on the global reference while avoiding obstacles. The planner also generates the longitudinal speed command. The vehicle will track a global reference speed unless the local path planner cannot find an obstacle free path. In that case, the planner will slow down the vehicle, coming to a stop if needed. The speed planner also halts the tractor at the end of the global reference path. The path tracking controller computes the error from the local plan and the curvature command for the vehicle accordingly.

## 3. AUXILIARY MODULES

This section provides an overview of the auxiliary modules implemented to test and validate the proposed planning strategy.

*Global reference path* The target path represents the route the vehicle has to follow when performing autonomous vineyard operations. The path must be sampled: it consists of a series of triples $(X, Y, \psi)$, where $\psi$ is the direction tangent to the reference at each point. The path feeds the local planner.

*Localization algorithm* An Extended Kalman Filter (EKF), implemented as in Pizzocaro et al. (2021), provides an estimate of the vehicle global position and heading with respect to the geographical North. The latters are inputs of the local planner. The observer exploits measures from wheel encoders (for the vehicle speed and curvature), an Inertial Measurement Unit (IMU), a GNSS antenna with RTK correction, and a magnetometer.

*Perception algorithm* A perception module, out of the scope of this paper, processes a 3D LiDAR point cloud to produce a 2D map of the obstacles surrounding the vehicle. An obstacle is represented by the minimum area convex polygon that contains all the point cloud samples assigned to it. For each obstacle, the local planner receives a matrix inequality describing the area occupied by the polygon and its sampled perimeter.

*Path tracking controller* The controller receives as input the optimal local plan as a series of sampled points. It computes the lateral error from the target path at a look-ahead point, by projecting the vehicle position along a direction defined by the vehicle heading. The curvature command results from an LPV H-infinity controller implemented as in Corno et al. (2020).

## 4. LOCAL PLANNER

This section discusses the local planner, the optimization problem it enforces, and the decision maker that selects the output plan. The planning algorithm receives as inputs the vehicle estimated position and heading, the obstacles map, and the global reference path; it accordingly optimizes a local plan with the objective to track the global reference while avoiding obstacles. The speed planner, discussed in the final subsection, either sets the speed to a global reference or slows down and eventually stops the vehicle, in case the local planner is not able to produce a path that do not cause a collision.

### *4.1 Main rationale*

The local planner algorithm main rationale is summarized in Fig. 2. Check Points (CP) are equispaced points on the global reference path. In correspondence of each CP, perpendicularly to the target path, we define Way Points (WP). The solution of an optimization problem provides the WP distances from the reference. The local plan results from the interpolation of the optimized WP. At each iteration the local planner identifies the CP zero based on the vehicle current position, where the WP zero is placed. The optimization computes the position of the WP corresponding to a number of CP consecutive to the CP zero. Its goals are to minimize the deviance from the target path while avoiding obstacles and producing a smooth local plan. Notice that CP are densely spaced so that, if two consecutive WP do not generate a collision, the points interpolating them do not collide with obstacles as well.

### *4.2 Optimization problem*

The formulation of the optimization problem is discussed hereafter. The optimization variables $x_i$ are the distances of the $i$-th WP from the reference path. The cost function

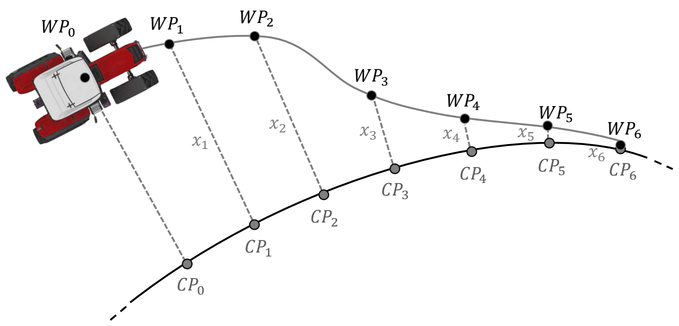

Fig. 2. Local planner main rationale: global reference path (black) and optimal local plan (gray).

is designed to be differentiable in order to guarantee the optimization convergence.

$$\begin{aligned} \min_{x_i} \quad & \alpha \sum_i x_i^2 + \beta \sum_i (x_i - x_{i-1})^2 + \gamma \sum_{i,j} f_{coll}^j(x_i) + \\ & + \delta \sum_{i,j} flag_{i,j} \frac{d_p^{i,j}}{d_c^{i,j}} + \varepsilon \sum_{i,j} f_{inf}^j(x_i) \qquad (1) \\ \text{s.t.} \quad & -\rho_{lim} \le \rho(x_i) \le \rho_{lim} \qquad i = 0, ..., n_{WP} \end{aligned}$$

The first term weighs the WP distances from the target path; the second one favors a smooth local plan by weighing the difference of consecutive WP deviance from the global reference. The third term introduces a collision function, defined for each obstacle $j$ in the map through a collision metric. Under the hypothesis of good tracking performance, the local plan describes the vehicle COG path. This introduces the need to take the vehicle footprint into account when determining the collision area. Finding the minimum distance between two polygons (*i.e.*, the obstacle and the vehicle) is a computationally intense operation, hence the introduction of a heuristic metric, characterized as:

$$r_p = \frac{d_r - d_p}{d_r} \qquad (2)$$

where $d_p$ is the distance between the vehicle COG and the closest point on the obstacle perimeter, and $d_r$ is the length of the portion of segment connecting the two points contained in the vehicle perimeter. As shown in Fig. 3, $r_p$ is negative when the vehicle is not in collision, $r_p = 0$ when the obstacle perimeter is tangent to the vehicle, and $0 < r_p < 1$ during collision. The terms $d_r, d_p$ in (2) become meaningless when the vehicle COG is inside the obstacle, thus $r_p$ saturates to 1. The collision function $f_{coll}$ is null when $r_p = 0$ and increases exponentially for $0 < r_p \le 1$. Its limitation lays with the fact that $r_p$ saturates to 1, preventing the cost to keep increasing while the collision *increases*. Hence, the addition of the fourth term in (1): the $flag$ becomes 1 when the vehicle COG is inside the obstacle, and $d_c$ is the distance between the vehicle COG and the obstacle centroid. The fourth term, thus, starts growing from zero to infinity when the obstacle is tangent to the vehicle COG as the collision *increases*. The fifth term represents an inflation function, which can be tuned to keep an additional safety distance from obstacles. Finally, $\alpha, \beta, \gamma, \delta, \varepsilon$ weight the different cost terms. The only hard constraint in (1) guarantees that the resulting local plan is feasible, by imposing that the

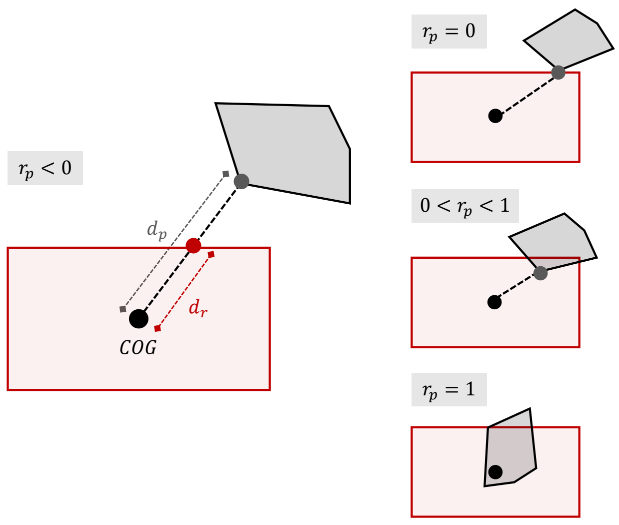

Fig. 3. Collision metric between the vehicle (red) and the obstacle (gray).

its curvature is within the vehicle dynamics and actuator limits. $\rho(x_i)$ is computed as the curvature of the path described by each triple of consecutive points, $\mathrm{WP}_{i-1}$, $\mathrm{WP}_i$, $\mathrm{WP}_{i+1}$.

### *4.3 Decision maker*

The local planner consists of two main modules: the core optimizer and the decision maker. The optimization runs in loop at the highest possible frequency. When a new local plan is published, the decision maker compares the new plan with the old one. The old plan cost is recomputed under the current conditions (*e.g.*, new obstacles may appear): the old path is prolonged keeping for the added WP the same distance from the global reference path of the last optimized WP. The plan with minimum cost is selected and sent to the underlying path tracking controller. The new plan may be discarded as, to ensure real-time execution, the core optimizer is stopped and forced to produce a local plan after a certain time interval. Hence, the optimization output may be sub-optimal if the timeout is reached. The decision maker thus guarantees that the local plan is only updated when the new plan has smaller cost. Additionally, frequent discontinuities in the control action are thus avoided.

### *4.4 Speed planner*

Unexpected dangerous conditions (*e.g.*, a big and close obstacle suddenly appearing in the vehicle field of view) may inhibit the local planner ability to produce a trajectory that do not cause a collision. Hence, the introduction of a speed planner, which monitors the optimized local plans: it intervenes when the selected local path involves at some point a collision with an obstacle, and reduces the vehicle speed. It eventually stops the vehicle, if the local planner is not able to recompute a collision free path, despite the reduced speed.

## 5. VALIDATION

The following section discusses the results obtained with the designed local planner. The performance assessment derives from the comparison with the state-of-the-art solution for mobile robots navigation, ROS Navigation Stack based on DWA. The comparison with the benchmark planner is carried out in a simulation environment, described hereafter. Finally, an on-field experimental campaign validates the proposed system.

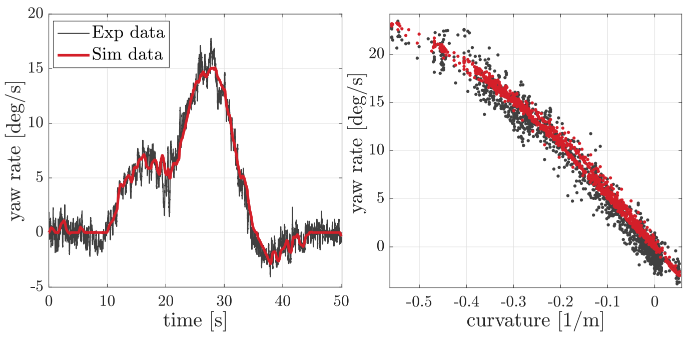

Fig. 4. Model and parameters validation.

### *5.1 Simulation framework*

A simulation environment, developed in Gazebo [2], replicates the tractor behavior and vineyard conditions, with the vines modeled as walls and other obstacles as generic shapes. The model simulating the vehicle dynamics is a linearized single-track model. Table 1 gathers the model parameters: the ones that cannot be measured are identified from experimental data.

Table 1. Single-track model parameters.

| Name | Value | Unit | Description |
|---|---|---|---|
| $M$ | 3000 | kg | Vehicle mass |
| $L_f$ | 1.42 | m | Vehicle front wheelbase |
| $L_r$ | 0.76 | m | Vehicle rear wheelbase |
| $C_f$ | 10000 | N/rad | Front wheel cornering stiffness |
| $C_r$ | 20833 | N/rad | Rear wheel cornering stiffness |
| $J_z$ | 2600 | kg · m | Vehicle yaw moment of inertia |

Fig. 4 validates the model and the identified parameters, by comparing the model output with experimental data collected on field. On the right, the yaw rate measured during a U-turn maneuver performed at 1 m/s is compared with the one simulated in Gazebo when feeding the model with the steering angle and longitudinal speed observed on field. On the left, similarly, the yaw rate - curvature characteristic curve extrapolated from a 20 minutes test drive in a vineyard is compared with the one derived from simulated data.

### *5.2 Experimental set-up*

The test vehicle is a SAME Frutteto CVT, a small dimensions agricultural tractor, characterized by high maneuverability and generally employed for procedures and treatments in vineyards and orchards. The vehicle sensors and actuators set-up follows.

- Wheel encoders measure the vehicle speed and curvature.
- An IMU, placed in the tractor COG, is exploited for the yaw rate measure.
- A single GNSS antenna with RTK correction provides the vehicle global position, with precision of ±2 cm.

[2] Gazebo Sim. `https://gazebosim.org/home`.

- A magnetometer estimates the vehicle heading.
- A 3D LiDAR, positioned on the tractor hood, returns three-dimensional information about the environment surrounding the vehicle. The sensor point cloud is used to derive the vines and other obstacles position.
- A hydraulic steering actuator implements the requested curvature command by steering the vehicle front wheels.
- A cruise control system enforces the reference speed.

The algorithms described in this paper are implemented in ROS and run on-board on an Ubuntu machine.

### *5.3 Simulation validation*

The proposed planning strategy performance is firstly assessed by comparing it to the benchmark planner and tracking control of ROS Navigation Stack, based on DWA. DWA shows limitations when used to perform path tracking in contexts with high tracking accuracy requirements. DWA main objective is to reach a goal rather than tracking a global reference path. Hence, when approaching a U-turn, it tends to *cut the turn* in order to quickly reach the goal, which in vineyards can result in failing the end-of-row inversion maneuver and not being able to re-enter the vineyard. Additionally, obstacles are inflated without considering the vehicle approach direction. This also may result in failed attempts to re-enter the vineyard, since the inflated vines may preclude its access. Costmap and obstacles inflation parameters tuning enables DWA to correctly plan the end-of-row inversion maneuvers, at expense of its capacity to avoid obstacles, as the obstacles inflation radius must be set almost to zero. Increasing the radius guarantees correct obstacle avoidance, but prevents the planner to identify a viable route between vines. Fig. 5 illustrates what just described, with results from a simulation where a cylindrical obstacle of 1 m radius (in gray) is positioned on the global reference path (in black). The latter drives along a row (the vines are represented as rectangles, in gray) and enters it, after performing a U-turn. The benchmark first tuning is obstacle-avoidance oriented: the inflation radius is conservatively set considering the vehicle semi-length. As shown in blue, DWA succeeds in the avoidance maneuver but stays out of the vineyard, as the selected inflation radius precludes its access. Reducing the obstacles inflation radius so that the vehicle semi-width is considered (tuning 2), DWA is not able to avoid the obstacle positioned on the reference path (as shown in dotted light blue, the planner stops the vehicle immediately before the collision), but it would be able to correctly perform the inversion maneuver (in dotted gray). The proposed local planning strategy can both avoid the obstacle on the global path and accurately track the reference (in red), thus significantly reducing the *cut-the-turn* phenomenon and correctly entering the vineyard. The performance improvement reason lays in the different obstacle avoidance approaches: DWA enlarges the obstacles of the same quantity along all directions. Our solution takes the vehicle dimension and approach direction into account, thus determining the real collision area. The freedom in shaping the local reference path, as opposed to DWA circular trajectories, contributes to further increase the performance. Fig. 6 provides some snapshots of the local plans optimized by the proposed planner (in green) during the obstacle avoidance maneuver: the vehicle avoid the obstacle (*i.e.*, the white point cloud) with the minimum margin that prevents the collision, and returns on the reference path (in red) immediately after.

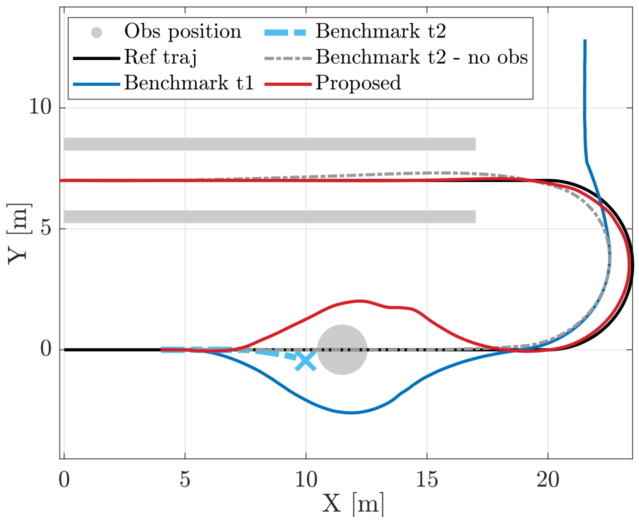

Fig. 5. Simulation validation: comparison with benchmark planning strategy.

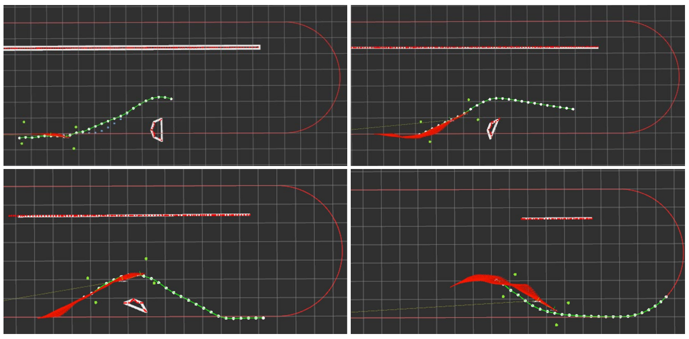
Fig. 6. Local plans generated during the obstacle avoidance maneuver. Vehicle extremities are the green dots.

Tuning the parameters and the shape of the functions of the minimization problem presented in Section 4 tailors the planner performance: tracking accuracy, margins left when avoiding obstacles, and aggressiveness of local plans leading back to the reference. Anyway, the cost function minimum is tuning-invariant, thus guaranteeing that the planner always maintains the desired behavior.

### *5.4 On-field validation*

The proposed planning and control system is finally validated on-field. Fig. 7 shows the results of an experiment with a global reference path that starts with a straight and ends with a U-turn maneuver, and with two obstacles placed on the target path in the inside of the U-turn and at its exit. The local planner keeps the vehicle on the reference path when the track is clear, and re-plans the route when obstacles are on or close to the target path. The speed planner requires to slow down during the U-turn maneuver due to the closeness to the obstacle, and stops the vehicle at the end of the global reference path.

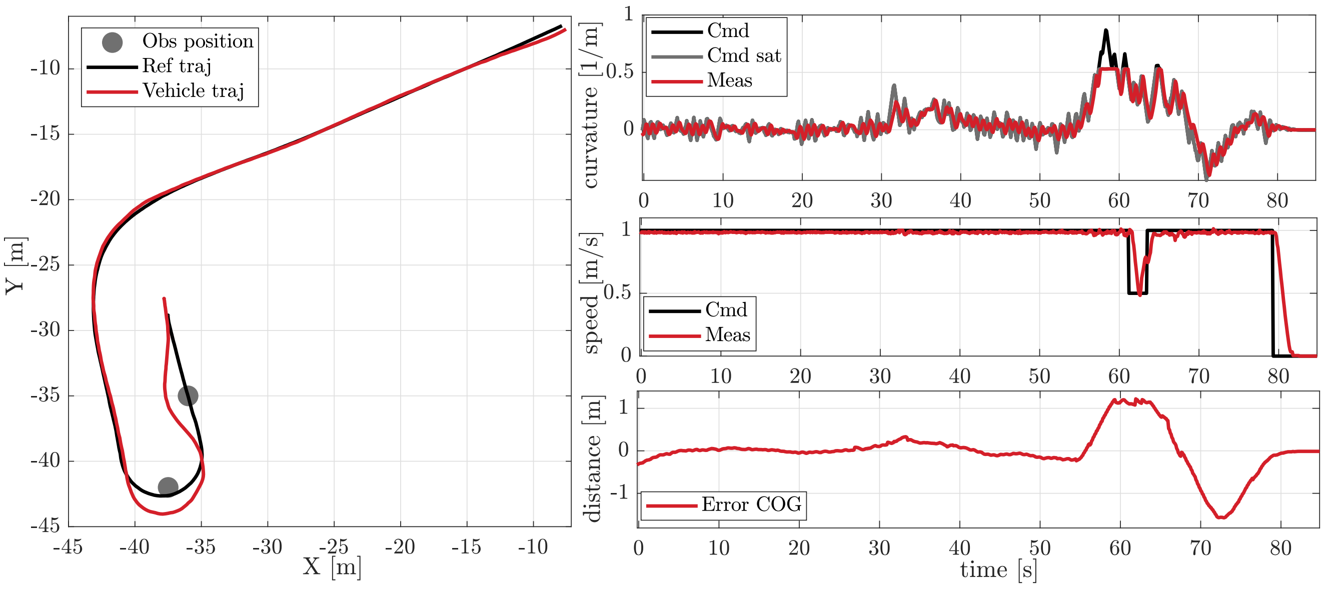

Fig. 7. On-field system validation.

## 6. CONCLUSIONS

We propose a local planning strategy for autonomous vineyard navigation which has the objective to track a park-to-park global reference path, while avoiding obstacles. The proposed solution overcomes limitations shown by the state-of-the-art planner and path tracker of ROS Navigation Stack. The main differences from the benchmark are related to the shape of the optimized local plans, which is not constrained to be circular, and the obstacle avoidance approach. Our planner determines the collision area between the vehicle and the obstacles, taking the vehicle footprint and approach direction into account. Consequently, the system tracks the global reference path, which sometimes forces the vehicle to brush against the crop rows, and simultaneously avoids obstacles on or close to the target path. The discussed planning strategy is tested in simulation, where it shows improved performance with respect to the benchmark, and experimentally validated with satisfactory results.

## REFERENCES

Astolfi, P., Gabrielli, A., Bascetta, L., and Matteucci, M. (2018). Vineyard autonomous navigation in the echord++ grape experiment. *IFAC-PapersOnLine*, 51(11), 704–709.

Barawid Jr, O.C., Mizushima, A., Ishii, K., and Noguchi, N. (2007). Development of an autonomous navigation system using a two-dimensional laser scanner in an orchard application. *Biosystems Engineering*, 96(2), 139–149.

Borenstein, J. and Koren, Y. (1990). Real-time obstacle avoidance for fast mobile robots in cluttered environments. In *Proceedings., IEEE International Conference on Robotics and Automation*, 572–577. IEEE.

Corno, M., Furioli, S., Cesana, P., and Savaresi, S.M. (2021). Adaptive ultrasound-based tractor localization for semi-autonomous vineyard operations. *Agronomy*, 11(2), 287.

Corno, M., Panzani, G., Roselli, F., Giorelli, M., Azzolini, D., and Savaresi, S.M. (2020). An lpv approach to autonomous vehicle path tracking in the presence of steering actuation nonlinearities. *IEEE Transactions on Control Systems Technology*, 29(4), 1766–1774.

Dong, F., Heinemann, W., and Kasper, R. (2011). Development of a row guidance system for an autonomous robot for white asparagus harvesting. *Computers and Electronics in Agriculture*, 79(2), 216–225.

Guo, J., Li, X., Li, Z., Hu, L., Yang, G., Zhao, C., Fairbairn, D., Watson, D., and Ge, M. (2018). Multi-gnss precise point positioning for precision agriculture. *Precision agriculture*, 19(5), 895–911.

Hiremath, S.A., Van Der Heijden, G.W., Van Evert, F.K., Stein, A., and Ter Braak, C.J. (2014). Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter. *Computers and Electronics in Agriculture*, 100, 41–50.

Koren, Y., Borenstein, J., et al. (1991). Potential field methods and their inherent limitations for mobile robot navigation. In *ICRA*, volume 2, 1398–1404.

Lenain, R., Thuilot, B., Cariou, C., and Martiner, P. (2004). A new nonlinear control for vehicle in sliding conditions: Application to automatic guidance of farm vehicles using rtk gps. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 5, 4381–4386. IEEE.

Pizzocaro, S., Corno, M., Pantano, M., and Savaresi, S. (2021). Magnetometer aided gps-free localization of an autonomous vineyard drone. In *2021 European Control Conference (ECC)*, 1132–1137.

Subramanian, V., Burks, T.F., and Arroyo, A. (2006). Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. *Computers and electronics in agriculture*, 53(2), 130–143.

Surbrook, J. and Gerrish, J. (1983). Mobile robots in agriculture. In *Amer Society of Agricultural*, volume 84, 30.

Yun, C., Kim, H.J., Jeon, C.W., and Kim, J.H. (2018). Stereovision-based guidance line detection method for auto-guidance system on furrow irrigated fields. *IFAC-PapersOnLine*, 51(17), 157–161.