Robotica (2024), 1–16
doi:10.1017/S0263574724000857

RESEARCH ARTICLE

Control framework of the ROBILAUT soil sampling robot: system overview and experimental results

Daniele Di Vito[1], Paolo Di Lillo[1], Filippo Arrichiello[1], Cesare Ferone[2], Raffaele Amico[3] and Gianluca Antonelli[1]

[1]Department of Electrical and Information Engineering, University of Cassino and Southern Lazio, Cassino, Italy
[2]Natura srl, Casoria, Italy
[3]OCIMA srl, Caivano, Italy
Corresponding author: Daniele Di Vito; Email: daniele.divito@unicas.it

Received: 13 November 2023; **Revised:** 26 February 2024; **Accepted:** 16 April 2024

Keywords: mobile robotics; field robotics; nonlinear control; model predictive control; environmental soil sampling

Abstract
The paper presents the control architecture of a crawler mobile robot designed and developed to sample potentially contaminated lands. The robot, developed in the framework of an Italian national project named ROBILAUT, carries a driller with a customized sampling mechanism to implement on-site the required quartering, and it is controlled to move the drilling device on specific points acquired in real time before the mission starts. The paper describes the software architecture for the navigation and control, focusing on the control framework of the robotic platform. Specifically, the robot exhibits a differential drive kinematics with actuators' constraints, and two different control strategies have been experimentally tested for comparison both in a structured environment and in the real site in May 2023.

1. Introduction
Soil sampling is required in several scenarios involving modern agriculture, construction, environmental analysis, or, in the marine domain, concerning offshore engineering and seabed mineral analysis. In recent years, several robotic platforms have been developed for sample collection. The work in ref. [1] presents a proof-of-concept demonstrator of a semi-autonomous robotic system for collecting soil samples, where a mechanical soil sampling and a storage system based on augers and turntable storage are used, and where the robot autonomously navigates to desired sampling locations. The paper in ref. [2] presents a planetary-like rover designed by a team of students as a testbed for planetary robotic exploration, like soil and rocks extraction and sampling, and for autonomous navigation in unstructured environment. In ref. [3], an unmanned ground vehicle with a custom-built robotic manipulator for oil sampling and terramechanics investigations is presented. The paper in ref. [4] presents the case study of a robotic soil sample collection system from bottom to top, where the sampling process automation is solved by using the robotic platform and electro-hydraulic mechanism, while the control system is connected with a cloud-based software that enables to create and manage the operation tasks. The work in ref. [5] presents the mechanical design of a soil sampling device and the control software of a mobile robotic platform for precision agriculture. The work in ref. [6] presents a prototype of two ground robots developed, as a team of a robotic swarm, to collect soil samples. The paper in ref. [7] presents a human-portable underwater robot for soil core sampling in the underwater domain; the paper in ref. [8] presents a drilling robot, based on earthworm locomotion, that was developed for seafloor exploration, while the work in ref. [9] presents a seafloor robotic explorer that can excavate and sample seafloor soil, with a discharging mechanism utilizing water jetting to improve the excavation depth.

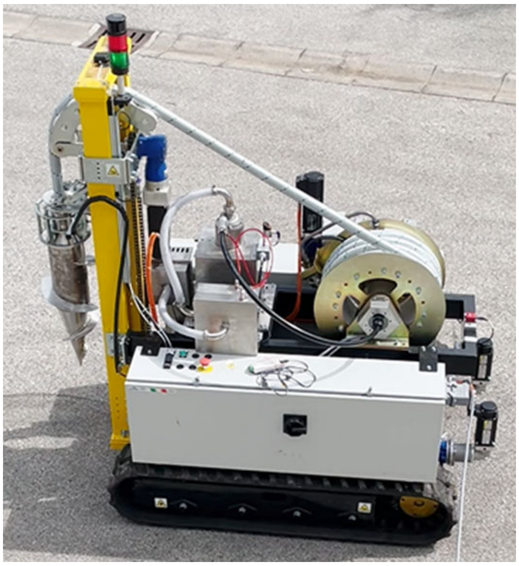

Figure 1. *Side view of the ROBILAUT robot: a crawler vehicle equipped with a driller to perform potentially contaminated soil sampling.*

Environmental sampling is required in several scenarios characterized by the existence of a pile of land made by potentially contaminated soil. Current Italian legislation requires to sample a representative amount of soil within a range of $2 \div 4$ kg, depending on the total volume of soils present in the area under investigation, that has to be sent to chemical laboratories for the successive analysis. More in detail, this operation, named *quartering*, consists in sampling several amounts of soils in specific 3D positions that have to be mixed together. Such activity is currently done manually, which is a tautologically dangerous process given the potential contamination of the site under analysis. The project ROBILAUT, funded by the Italian Ministero dello Sviluppo Economico (Ministry of Economical Development), had the objective to automatize the overall operation with benefits for the operators' health, to improve the representativeness of the sample and to decrease the operation's costs. In a first phase, an operator manually drives a drone in charge of building a 3D model of the soil pile; in the considered case, the pile is up to 3000 m^2 with an height of approximately 3 m. Then, the CAD model is sent to a motion planning engine which, taking into account the legislation constraints and the energy-related optimizing criteria, outputs the points where the samples need to be taken. Once the sampling point list is obtained, then the ROBILAUT robot, shown in Fig. 1, autonomously drives toward them and samples/mixes the specific amount of soil under the supervision of an operator that monitors the overall process using a properly developed graphical user interface (GUI). Indeed, the operator is not in the line of sight with the vehicle due to the pile height (usually 3 m); thus, there is the need to supervise the process and eventually activate an emergency procedure. The GUI is compatible with multiple devices over a local network; thus, the monitoring can eventually be shared or done remotely.

Fig. 2 shows the drilling operation which, however, is beyond the scope of this paper.

2. System overview

The robotic system developed within the ROBILAUT project is represented by a crawler vehicle equipped with a tethered drilling device. The vehicle carries all the computing modules, installed in a proper electrical automation cabinet shown in Fig. 3, and all the sensors needed to perform the autonomous navigation, that is:

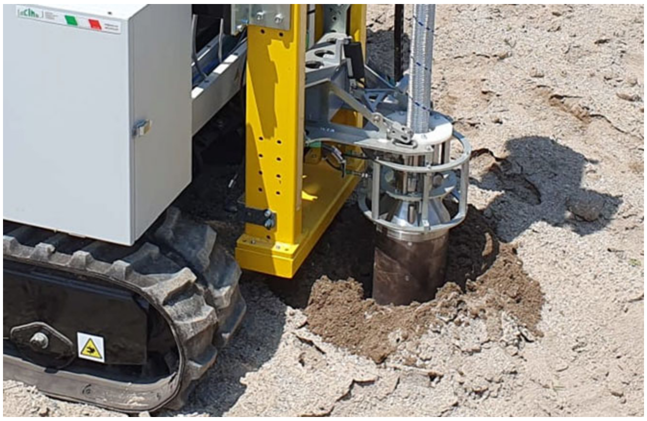

Figure 2. *Snapshot of the ROBILAUT robot during drilling, a mechanical sampling mechanics is embed in the head and allows for the required quartering task.*

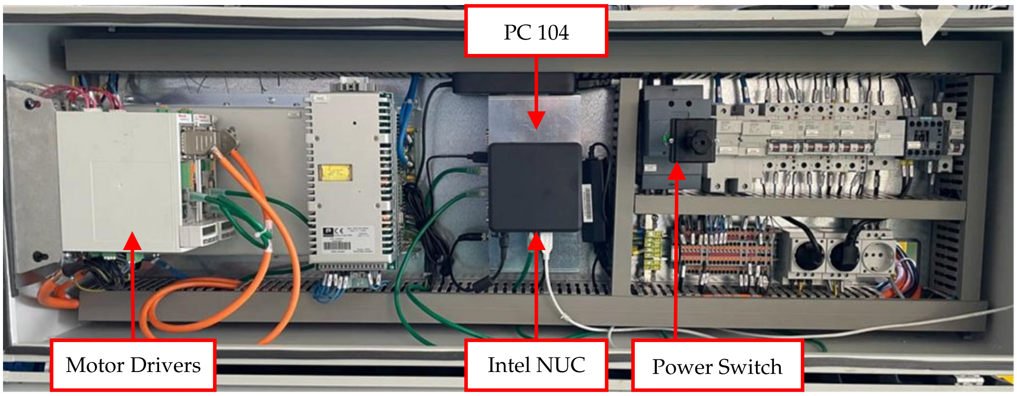

Figure 3. *Electrical automation cabinet containing the computing modules (Intel NUC and PC-104 class device), the motor drivers, and the general Power Switch.*

- an Intel NUC PC used for the acquisition and processing of the data from the inertial sensors and the localization module, and for the execution of the navigation control algorithm;
- an industrial computing module (PC-104 computer class), linked to the crawler motor drivers, dedicated to setting the desired velocities from the control output and reading the current motor status;
- a localization module based on global navigation satellite system (GNSS) technology; specifically, a real-time kinematic positioning (RTK) system which represents the most performing method to correct the errors in current GNSSs, obtaining an accuracy of the order of cm [10]. Such a system uses both information content and wave phase of satellite signals, and it relies on a set of ground reference stations to mitigate the atmospheric effects, for example, the ionospheric delay on the measured distance. The set of ground reference stations can include only one station; however, the latter has to be positioned within a distance of 50 km from the target that needs to be localized. This is the case of the ROBILAUT system where a commercial RTK kit is used, that is, the ArduSimple Starter kit, which includes two GNSS receivers: one installed on top of the vehicle (as shown in Fig. 4) and one to be used for the ground station. The two receivers are equipped with two long-range modules that allow the communication for the localization correction;

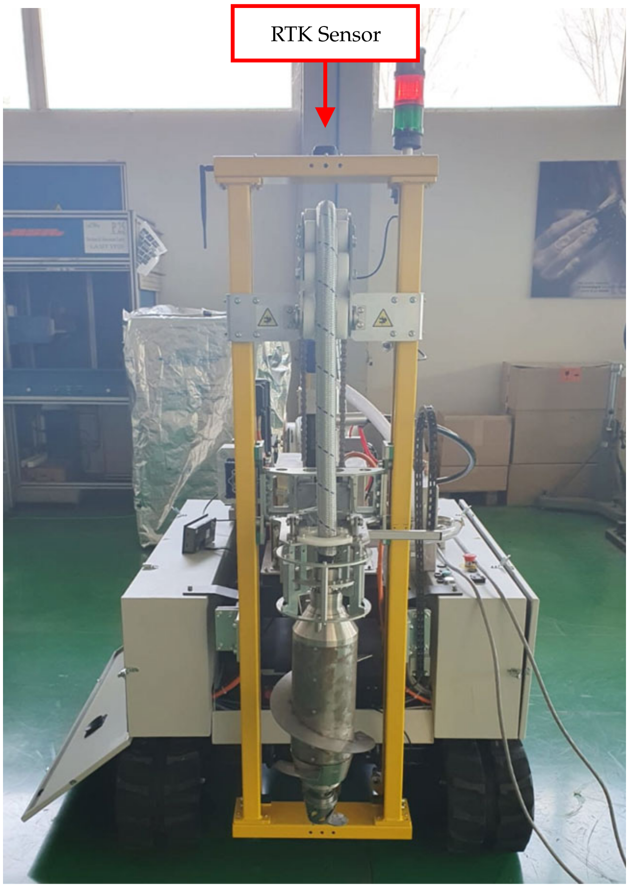

Figure 4. *RTK-GNSS receiver installed on top of the ROBILAUT vehicle.*

- an inertial measurement unit (IMU), installed in the back of the ROBILAUT vehicle, for minimizing interferences (see Fig. 5), represented by the Ellipse-E, a commercial solution provided by SBG Systems. This module includes an accelerometer, a gyroscope, a magnetometer, as well as a temperature sensor. Furthermore, it applies an internal data fusion algorithm, that is, a Kalman filter, to provide a better estimation of navigation data such as the vehicle yaw, which can be directly read from the magnetometer or taken from the data fusion algorithm output.
- a joystick allowing the human operator to manually move the vehicle both for safety and logistics reasons (see Fig. 6); in addition to the direction (arrows) buttons, it is equipped with an emergency stop button to be used in dangerous situations, such as a person moving close to the vehicle during the parking maneuvers.

3. Overall software architecture

The software architecture has been developed using the robot operating system (ROS) middleware. Fig. 7 shows the overall architecture where each box represents the *i*-th ROS node running the corresponding process and where each arrow represents a specific ROS topic used for data exchange. In particular:

- **Web server**: it is the web server that runs the GUI exchanging data with an interface node through a socket. In particular, the GUI is used by the human operator to set the desired

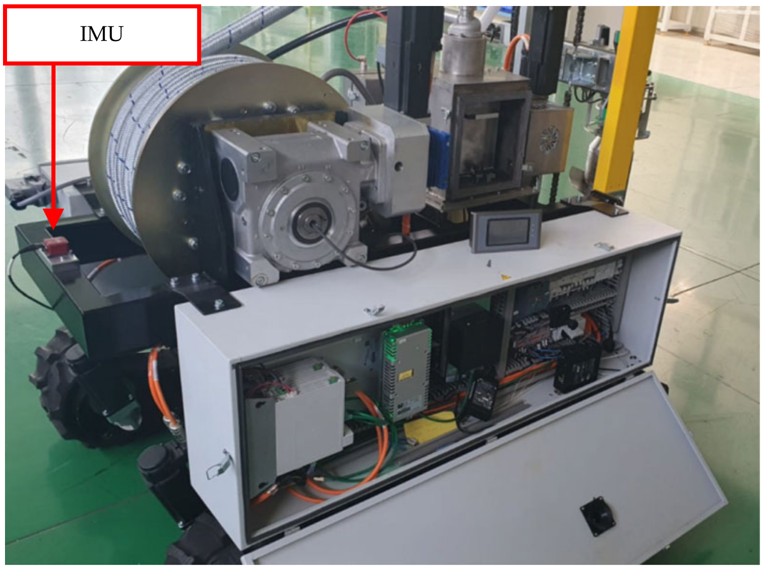

Figure 5. *Ellipse-E IMU by SBG Systems installed in the back of the ROBILAUT vehicle for minimizing the interference.*

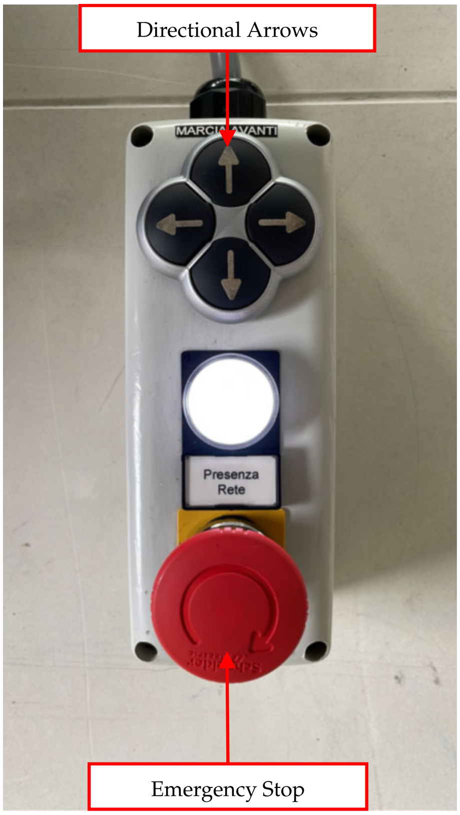

Figure 6. *Joystick for the manual movement of the ROBILAUT vehicle equipped with emergency button for safety reasons.*

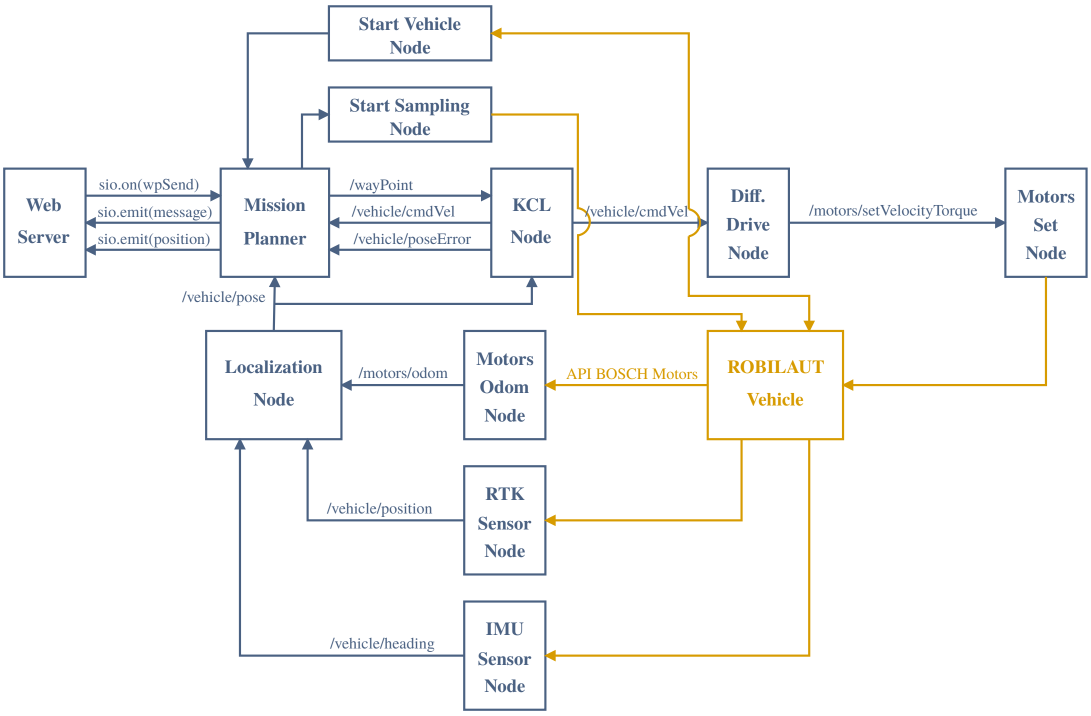

Figure 7. *ROBILAUT system software architecture based on the robot operating system (ROS) middleware.*

waypoints (`sio.on(wpSend)`) and to monitor the mission execution (texttttsio.emit(message), sio.emit(position)). A more detailed description of the GUI is reported in Section 3.1.

- **Mission planner**: it is the node that acts as the bridge between the GUI web server using Python sockets and the ROS middleware. It receives as input the desired waypoints (`sio.on(wpSend)`) for the control node (`/wayPoint`), the trigger to let the vehicle start the movement, the vehicle pose (`/vehicle/pose`), the pose error (`/vehicle/poseError`), and the reference velocity (`/vehicle/cmdVel`). Then, it sends the current vehicle status back to the GUI web server (`sio.emit(message)`, `sio.emit(position)`). It is also in charge of sending the starting trigger for the sampling operations.
- **KCL node**: the kinematic control layer (KCL) node runs the implemented controllers, described in Section 5, necessary to compute the reference velocity (`/vehicle/cmdVel`) for the vehicle to navigate following the desired waypoints (`/wayPoint`). The waypoint list and current vehicle pose (`/vehicle/pose`) represent the node input, while it returns as output the reference velocity (`/vehicle/cmdVel`) and the pose error (`/vehicle/poseError`).
- **Diff. drive node**: since the vehicle presents a differential drive kinematics, as described in Section 4, this node is in charge to take the input reference velocity (`/vehicle/cmdVel`) and to map it into proper desired velocities for the two crawl motors that are sent to motors set node (`/motors/setVelocityTorque`).
- **Motors set node**: it represents a ROS wrapper for the API provided by the motors supplier, that is, Bosch; therefore, it allows to interface the ROS middleware with the motors' low-level controllers.
- **Motors odom node**: it is a ROS wrapper for the Bosch API as well; in this case, the nodes give as output of the motor odometry data, that is, the motor velocity and position.

- **RTK sensor node**: it is in charge of acquiring the data from the RTK module via HTTP requests, since the latter is interfaced with a micro web server implemented for the case at hand that converts the GNSS coordinates into the UTM (Universal Transverse Mercator) reference system, that is, North-East coordinates in meters. Then, position data are sent inside the ROS framework via the topic (`/vehicle/position`).
- **IMU sensor node**: this is the ROS package provided by SBG which allows to access to every information regarding the sensor status. In the specific case, the vehicle heading is the data of interest (`/vehicle/heading`).
- **Localization node**: this node runs a proper data fusion algorithm, that is, the Kalman filter, taking as input the data coming from motors, RTK and IMU sensor. It returns as output a noise-filtered and continuous signal (`/vehicle/pose`), necessary for the proper operative functioning of the controller, containing the robot pose estimate.
- **Start vehicle node**: it reads the status of the drilling unit by querying the programmable logic controller (PLC) through a standard communication protocol, that is, the OPC UA (Open Platform Communications Unified Architecture). Indeed, before moving the vehicle, it is necessary to control that the system is not performing the sampling operation and that the drilling unit is in its home position. If all the related checks are verified, then the start vehicle node sends a trigger to the mission planner.
- **Start sampling node**: this node waits for a trigger from the mission planner. Indeed, when the robot reaches the i-th desired point, the overall system movement is stopped and a consensus signal is sent to the start sampling node. Then, the latter interacts with the PLC via OPC UA, as mentioned for the start vehicle node. If the consensus is verified, then the PLC stars the drilling operation that is implemented according to a state machine approach.

3.1. Graphical user interface (GUI)

As mentioned above, the human operator is not in line of sight with the vehicle due to the soil pile height. Therefore, a tool for monitoring the overall process status is necessary. For this reason, a proper GUI has been implemented to support the operator. More in detail, the GUI has been developed using web frameworks and libraries resulting in a cross-platform application that is accessible from any mobile and desktop web browser without requiring any installation. As shown in Fig. 8, a 3D scenario of the environment with a CAD model of the robot system is rendered, allowing the operator to immediately understand what is the mission status. Furthermore, it shows several indicators regarding the robot and the controller status, for example, the vehicle reference velocity, the position error, and the battery charge. Moreover, the operator can set the desired waypoints and track the robot's movement in real time, as noticeable in Fig. 9.

3.2. Waypoints generation

The human operator can set any waypoint through the GUI. However, it is worth noticing that the waypoint list is obtained by means of a digitalization operation of the soil pipe. In particular, using an aerial drone equipped with a RGB-D sensor, a scan of the pile is performed. The obtained point cloud is then used to construct the mesh corresponding to the pile and to generate the sampling waypoints list according to the legislation, as observable in Fig 10.

4. Kinematic modeling

The ROBILAUT robot can be modeled as a differential drive kinematic robot, where the two crawlers can be independently controlled, and where the control objective is the positioning of the drilling point.

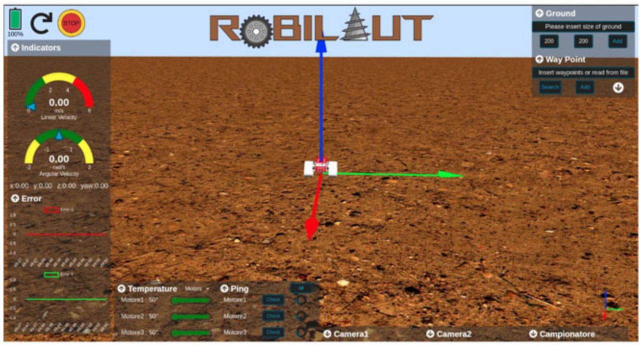

Figure 8. *ROBILAUT graphical user interface (GUI): the human operator can set the desired waypoint and monitor the robot status.*

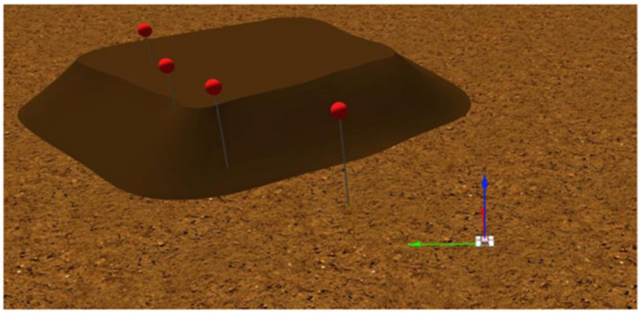

Figure 9. *ROBILAUT graphical user interface (GUI): monitoring of the robot movement through waypoints (red markers).*

Referring to Fig. 11, we denote as Σ_i, $\mathbf{x}_i$, $\mathbf{y}_i$ a fixed inertial reference frame, and as Σ_b, $\mathbf{x}_b$, $\mathbf{y}_b$ a body-fixed frame located in the center of the crawler base $\begin{bmatrix} x' & y' \end{bmatrix}^{\mathrm{T}}$ with $\mathbf{x}_b$ in the advancing direction of the robot. The drilling device positioning, denoted as $\mathbf{p} = \begin{bmatrix} x & y \end{bmatrix}^{\mathrm{T}}$, is a point at distance δ from the center of the crawler base. Furthermore, we denote as θ the vehicle orientation and as v, ω the linear and angular velocities at the body-fixed frame. Denoting as ω_R, ω_L, r, l, respectively, the angular velocity of the driving wheels of the right and left crawlers, the radius, and the inter-axis distance, then

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/l & -r/l \end{bmatrix} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} = \boldsymbol{B} \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \tag{1}$$

The position of the controlled point is described by:

$$\begin{cases} x = x' + \delta \cos\theta \\ y = y' + \delta \sin\theta \end{cases} \tag{2}$$

characterized by dynamics

$$\begin{cases} \dot{x} = v\cos\theta - \delta\omega\sin\theta \\ \dot{y} = v\sin\theta + \delta\omega\cos\theta \\ \dot{\theta} = \omega \end{cases} \tag{3}$$

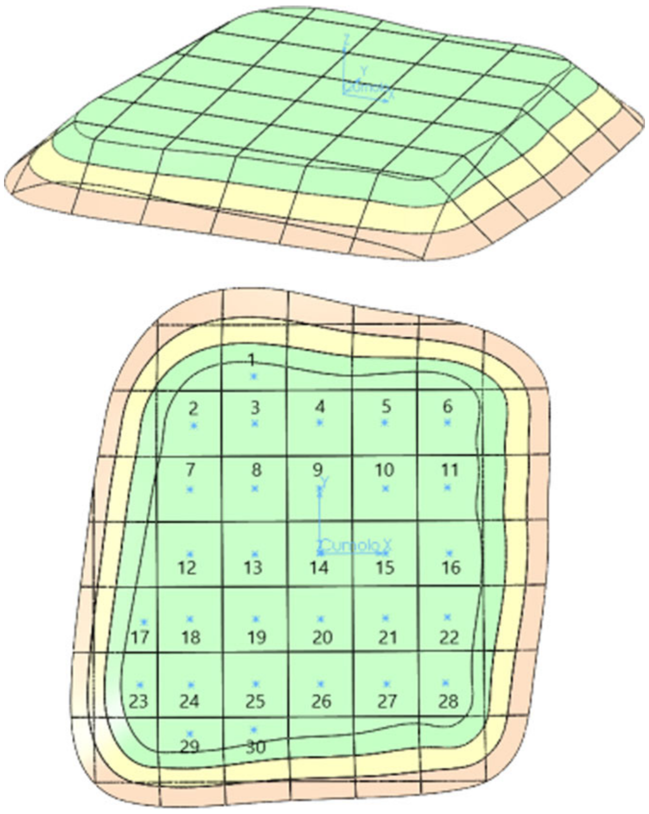

Figure 10. *Mesh corresponding to the digitalized soil pile with the grid to determine the sampling waypoints according to the legislation.*

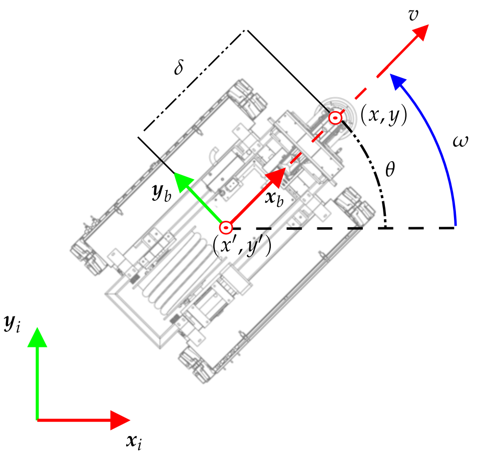

Figure 11. *Kinematic model of the ROBILAUT robot.*

that is,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\delta\sin\theta \\ \sin\theta & -\delta\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{4}$$

5. Kinematic motion control

For the motion control of the robotic platform, two different control strategies taken from the literature have been tested for the scenario at hand, respectively, a feedback linearization technique and a model predictive control (MPC) strategy.

5.1. Feedback linearization

The feedback linearization technique, taken from [11,12], is an efficient design tool leading to a solution simultaneously valid for both trajectory tracking and setpoint regulation problems. Referring to eq. (4), the robot's linear and angular velocity are computed as:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta)/\delta & \cos(\theta)/\delta \end{bmatrix} \begin{bmatrix} \dot{x}_d + k_x(x_d - x) \\ \dot{y}_d + k_y(y_d - y) \end{bmatrix}$$

where the subscript d denotes the desired value and k_x and k_y are properly positive gains to be designed. The body-fixed linear and angular velocities are further converted into wheel velocities resorting to eq. (1):

$$\begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} = \boldsymbol{B}^{-1} \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 1/r & l/2r \\ 1/r & -l/2r \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}. \tag{5}$$

5.2. Model predictive control

The MPC strategy is an optimal control technique based on the minimization of a cost index, that is, a vector-dependent quadratic function of the state and the control inputs. At each time stamp, the MPC output is iteratively computed as the control input that minimizes the cost index over a finite horizon, eventually satisfying specific constraints.

In the considered case, the MPC is computed considering the error state dynamics. Thus, let us first define the error state variable:

$$\boldsymbol{e} = \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{bmatrix}$$

and compute its derivative:

$$\dot{\boldsymbol{e}} = \begin{bmatrix} \dot{e}_x \\ \dot{e}_y \\ \dot{e}_\theta \end{bmatrix} = \begin{bmatrix} \dot{x}_d - \dot{x} \\ \dot{y}_d - \dot{y} \\ \dot{\theta}_d - \dot{\theta} \end{bmatrix}.$$

In a regulation problem, all the derivatives of the reference terms $(\dot{x}_d, \dot{y}_d, \dot{\theta}_d)$ are null; thus,

$$\dot{\boldsymbol{e}} = \begin{bmatrix} -v\cos(\theta_d - e_\theta) + \delta\omega\sin(\theta_d - e_\theta) \\ -v\sin(\theta_d - e_\theta) - \delta\omega\cos(\theta_d - e_\theta) \\ -\omega \end{bmatrix}.$$

The nonlinear model can be linearized around a working trajectory to assume the form (the increment symbol δ will be omitted to increase readability):

$$\dot{\boldsymbol{e}} = \boldsymbol{A}\boldsymbol{e} + \boldsymbol{B}\boldsymbol{u}$$

where $\boldsymbol{u} = [v \quad \omega]^{\mathrm{T}}$,

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & -v\sin(\theta_d - e_\theta) - \delta\omega\cos(\theta_d - e_\theta) \\ 0 & 0 & v\cos(\theta_d - e_\theta) - \delta\omega\sin(\theta_d - e_\theta) \\ 0 & 0 & 0 \end{bmatrix}$$

and

$$\boldsymbol{B} = \begin{bmatrix} -\cos(\theta_d - e_\theta) & \delta\sin(\theta_d - e_\theta) \\ -\sin(\theta_d - e_\theta) & -\delta\cos(\theta_d - e_\theta) \\ 0 & -1 \end{bmatrix}.$$

The model can be discretized, referring to the Euler method, in the form

$$\begin{cases} \boldsymbol{e}(k+1) &= \boldsymbol{A}_k\boldsymbol{e}(k) + \boldsymbol{B}_k\boldsymbol{u}(k) \\ \boldsymbol{y}(k) &= \boldsymbol{C}_k\boldsymbol{e}(k) \end{cases}$$

where

$$\boldsymbol{A}_k = \boldsymbol{I}_3 + T\boldsymbol{A}$$

$$\boldsymbol{B}_k = T\boldsymbol{B},$$

the matrix $\boldsymbol{I}_r$ is the $r \times r$ Identity matrix and T is the sampling time. The matrices $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$ need to be computed at each sampling time with respect to the current working point.

For the MPC strategy (see [13]), the minimization problem can be formulated as:

$$\begin{array}{ll} \min\limits_{\boldsymbol{u}} & \sum_{k=0}^{N-1}\left[\boldsymbol{e}^{\mathrm{T}}(k)\boldsymbol{Q}\boldsymbol{e}(k) + \boldsymbol{u}^{\mathrm{T}}(k)\boldsymbol{R}\boldsymbol{u}(k)\right] + \boldsymbol{e}^{\mathrm{T}}(N)\boldsymbol{P}\boldsymbol{e}(N) \\ s.t. & \boldsymbol{e}(k+1) = \boldsymbol{A}_k\boldsymbol{e}(k) + \boldsymbol{B}_k\boldsymbol{u}(k) \end{array} \tag{6}$$

where $\boldsymbol{Q} = \boldsymbol{Q}^{\mathrm{T}} \geq \boldsymbol{O} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R} = \boldsymbol{R}^{\mathrm{T}} \geq \boldsymbol{O} \in \mathbb{R}^{p\times p}$ are the error state and the input weight matrices, $\boldsymbol{P} = \boldsymbol{P}^{\mathrm{T}} \geq \boldsymbol{O} \in \mathbb{R}^{n\times n}$ is a terminal cost, and N is the finite number of steps of the temporal prediction horizon that is shifted forward at each iteration. This approach is also known as receding horizon, where at instant k an optimal control problem is solved on the finite horizon of N steps, and, at $k+1$, the current measured output is used to iterate the process taking as reference the current time instant $k+1$. The MPC formulation in eq. (6) can also be extended by considering constraints on the input signal $\boldsymbol{u}$.

5.3. Saturation management strategy

Considering that, at the wheels' angular velocities level, saturation constraints apply (i.e.,$\left|\omega_{R,L}\right| < \omega_{thld}$), the smart saturation strategy described in Algorithm 1 has been applied to properly scale the input generated by the motion control algorithms to affordable wheels' angular velocities that do not change the robot path.

6. Experiments

The kinematic control solutions described in the previous section have been experimentally tested with the developed robotic platform. In particular, the robot has been commanded to reach a sequence of waypoints using the following kinematic controls:

- feedback linearization;
- unconstrained MPC;
- constrained MPC.

Algorithm 1 Smart saturation

```
1:  for $i \leftarrow 1$ to $\#motors$ do
2:      if abs($omega[i]$) $> maxOmega$ then
3:          $saturate \leftarrow True$
4:          $alpha \leftarrow maxOmega/\text{abs}(omega[i])$
5:          if $alpha <= minAlpha$ then
6:              $minAlpha \leftarrow alpha$
7:          end if
8:      end if
9:  end for
10: if $saturate$ then
11:     for $i \leftarrow 1$ to $\#motors$ do
12:         $omega[i] \leftarrow minAlpha * omega[i]$
13:     end for
14: end if
```

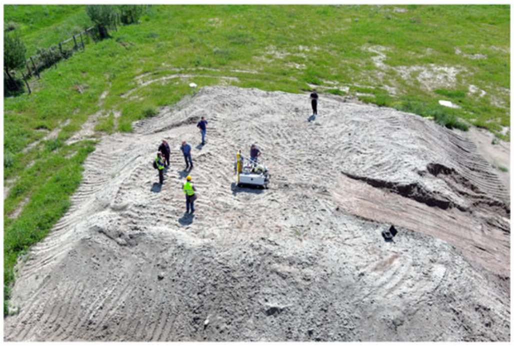

Figure 12. Soil pile of the experimental settings, aerial view.

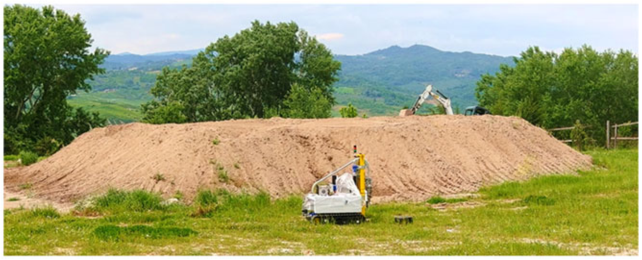

Figure 13. Soil pile of the experimental settings, side view.

Specifically, in the constrained MPC, due to a specific request by one of the project partners, a constrain on the linear velocity has been included to make the vehicle move only in the forward direction ($0 < v < v_{max}$).

All the three approaches are followed by the smart saturation strategy described in Sect. 5 to take into account the maximum velocities of the motors, which is 4000 rpm after a gear reduction of 700.

The sampling time is $T = 100$ ms. For the feedback linearization solution, the gains have been chosen as $k_x = k_y = 0.5$; for the MPC strategy, the weight matrices have been chosen as $\boldsymbol{Q} = I_3$, $\boldsymbol{R} = 2I_2$, and $\boldsymbol{P} = 0.5I_3$ with horizon $N = 5$.

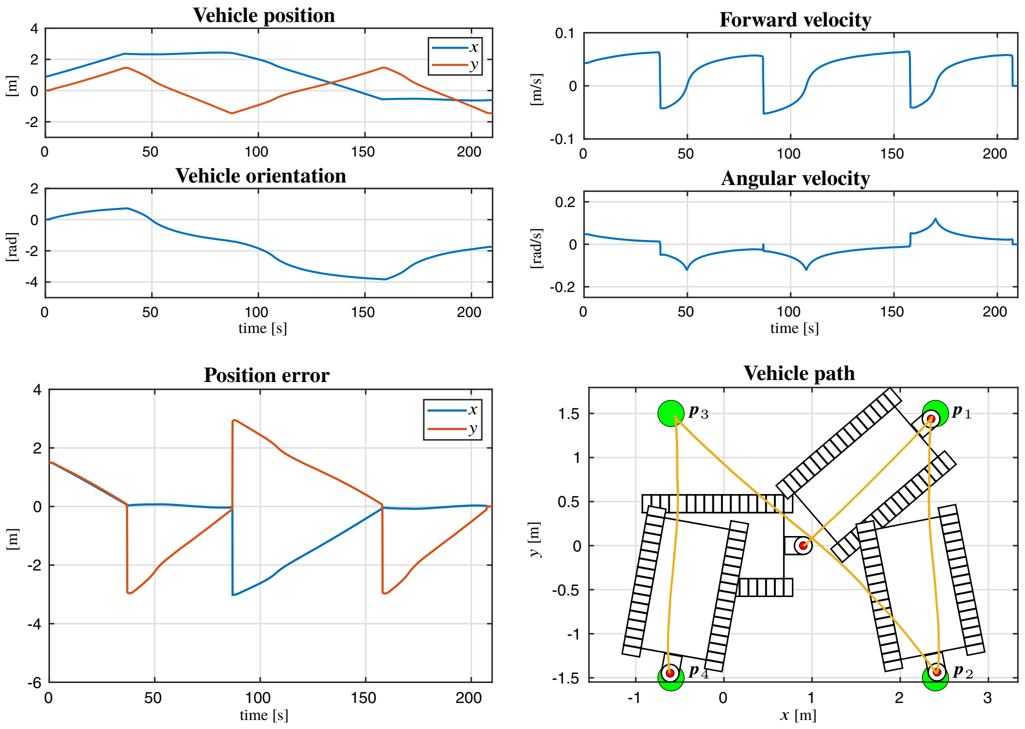

Figure 14. *Vehicle's position and orientation, position error, linear and angular velocity, and path using the feedback linearization kinematic control.*

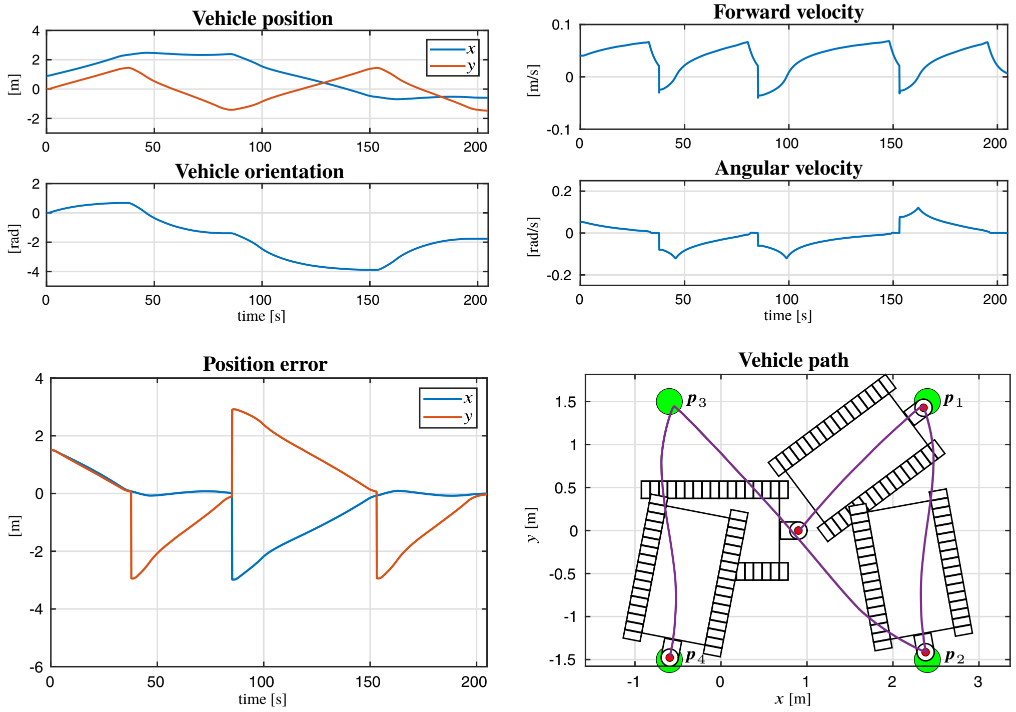

Figure 15. *Vehicle's position and orientation, position error, linear and angular velocity, and path using the unconstrained MPC strategy.*

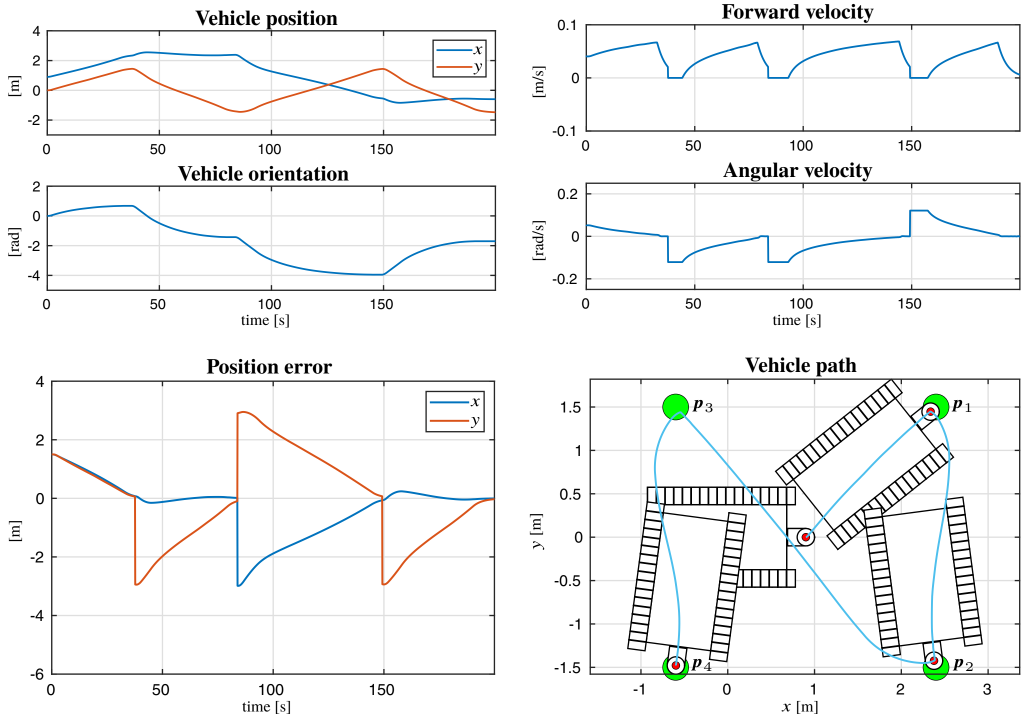

Figure 16. *Vehicle's position and orientation, position error, linear and angular velocity, and path using the constrained MPC strategy.*

The soil to be sampled is usually arranged as shown in Fig. 12 and 13. The initial vehicle position is set at $x' = 0$, $y' = 0$, which implies $\boldsymbol{x}_0 = \boldsymbol{x}(t = 0) = [0.9 \quad 0]^{\mathrm{T}}$ m. A set of four waypoints have been sequentially assigned to make the vehicle move in all directions:

$$\begin{aligned}
\boldsymbol{p}_1 &= [2.4 \quad 1.5]^{\mathrm{T}} \quad \mathrm{m} \\
\boldsymbol{p}_2 &= [2.4 \quad -1.5]^{\mathrm{T}} \quad \mathrm{m} \\
\boldsymbol{p}_3 &= [-0.6 \quad 1.5]^{\mathrm{T}} \quad \mathrm{m} \\
\boldsymbol{p}_4 &= [-0.6 \quad -1.5]^{\mathrm{T}} \quad \mathrm{m} \qquad (7)
\end{aligned}$$

The waypoints have been kept close to each other for safety reasons, since the day of the experiments there was no possibility to check for the proper consolidation of the side of the pile. The effective sampling operation was not the objective of the reported experimental campaign; thus, the vehicle was commanded to move to the successive way point just after one was reached. A threshold of 10 cm has been assigned to the controller to command a switch toward the successive waypoint.

Fig. (14), (15), and (16) show the vehicle's position and orientation, the position error, the linear and angular velocity, and the followed path using the feedback linearization, the unconstrained MPC, and the constrained MPC strategies, respectively. Since all of them have been implemented also considering the smart saturation feature, all the computed velocities resulted physically executable for the low-level controllers.

It can be noticed that all the controllers achieved satisfactory results and output smooth control actions. As ex-post validation metrics, the sums of the distances of the path points from the segments connecting two successive via points have been computed and normalized with respect to the largest value (obtained with the constrained MPC); such values are reported in Table I together with the total

Table I. *Sums of the distances of the points of the paths from the segments connecting two successive via points, their total sums, and traveled time. All the values are normalized with respect to the constrained MPC case.*

Algorithm	$s(\boldsymbol{x}_0, \boldsymbol{p}_1)$	$s(\boldsymbol{p}_1, \boldsymbol{p}_2)$	$s(\boldsymbol{p}_2, \boldsymbol{p}_3)$	$s(\boldsymbol{p}_3, \boldsymbol{p}_4)$	s_{total}	**Traveled time**
Feedback linearization	0.04	0.12	0.34	0.10	0.60	1.05
Unconstrained MPC	0.07	0.12	0.20	0.13	0.53	1.025
Constrained MPC	0.07	0.16	0.54	0.23	1.00	1.00

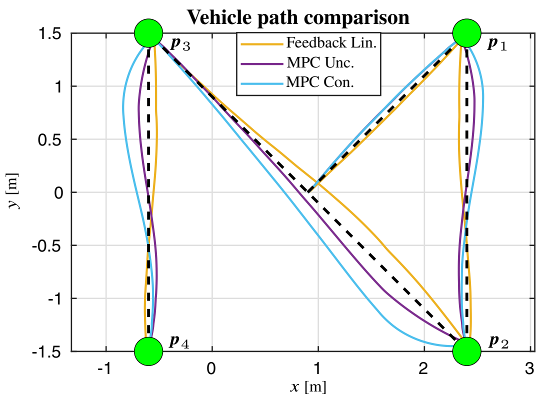

Figure 17. *Path of the experimental implementation of the three controllers.*

traveled time (normalized as well with respect to the constrained MPC case). Fig. 17 reports the superimposition of the paths for the three tested controllers for a graphical comparison. In particular, the two controllers allowed to implement negative velocities exhibit similar performances; considering the rough terrain, indeed, the small difference can be considered as negligible. The constrained MPC, on the other hand, needs larger maneuvering space. Concerning the traveled time, there are no significant differences in the performance of the three controllers.

Another difference in the controllers is related to the ease of gain tuning. It is a common opinion in the control community that, despite the easy theoretical interpretation of the MPC gains, their fine tuning on real hardware turns out to be more time-consuming. Our experience is in line even for a so simple mathematical model.

A video with some extracts of the vehicle movement together with a rendering of the real data is available at https://youtu.be/PKliM3QHA4M.

7. Conclusions

The control architecture and the experimental validation of two motion control approaches for a crawler robot have been discussed in this paper. The crawler robot, developed within the framework of an Italian national project, is designed to properly sample soils for further chemical analysis to detect the eventual presence of contaminants. Three different controllers have been implemented and tested, namely a nonlinear feedback linearization controller and two MPC-based approaches. For the reasons discussed in the experimental section, the nonlinear controller exhibits better performance and has been implemented in the final robot architecture.

Author contributions. Cesare Ferone conceived the Project. Raffaele Amico led the design of the mobile robot. Gianluca Antonelli and Filippo Arrichiello analyzed the kinematics and dynamics and proposed the most suitable control algorithms for the

use-case scenario. Daniele Di Vito and Paolo Di Lillo implemented the entire control framework for the autonomous navigation of the system and conducted the field experiments. Each author equally contributed to the writing of the paper.

Financial support. The research leading to these results has received funding from the Italian Government, Ministero per lo Sviluppo Economico, Fondo per la Crescita Sostenibile – Sportello “Fabbrica intelligente” PON I&C 2014-2020, ROBILAUT prog. n. F/190126/01-02-03/X44, and by Project “Ecosistema dell’innovazione – Rome Technopole” and “Centro Nazionale per la Mobilità Sostenibile – CNMS” financed by EU in NextGenerationEU plan through MUR Decree n. 1051 23.06.2022 – CUP H33C22000420001 and CUP H38H22000300001, respectively.

Competing interests. The authors declare no competing interests exist.

Ethical approval. None.

References

[1] V. Edulji, S. Soman, A. Pradhan and J. Shah, A mobile semi-autonomous robot for soil sampling, (2021).
[2] S. Chiodini, A. Carron, M. Pertile, M. Todescato, E. Bertolutti, A. Bilato, A. Boscardin, G. Corra, A. Correnti, R. Dalla Vecchia, N. Dal Lago, M. Fadone, S. Fogarollo, H. Milani, E. Mion,D. Paganini, F. Quadrelli, G. Soldà, A. Todescan, E. Toffanin and S. Debei, “Morpheus: A Field Robotics Testbed for Soil Sampling and Autonomous Navigation,” **In:** *Proceedings of the Proc. 1st Symposium on Space Educational Activities*, (2015).
[3] N. A. Olmedo, M. Barczyk, H. Zhang, W. Wilson and M. G. Lipsett, “A UGV-based modular robotic manipulator for soil sampling and terramechanics investigations,” *J Unmann Vehi Syst* **8**(4), 364–381 (2020).
[4] E. Vaeljaots, H. Lehiste, M. Kiik and T. Leemet, “Soil sampling automation case-study using unmanned ground vehicle,” *Eng Rural Dev* **17**, 982–987 (2018).
[5] E. Väljaots, H. Lehiste, M. Kiik and T. Leemet, “Soil sampling automation using mobile robotic platform,” *Agro Res* **16**, 917–922 (2018).
[6] C. N. A. David, J. V. Yumol, R. G. Garcia and A. H. Ballado, “Swarm Robotics Application for Gathering Soil Samples,” **In:** *Proceedings of the 2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM). IEEE*, (2021) pp. 1–6.
[7] K. Yokoi, M. Kawabata, S. Sakai, S. Kawamura, N. Sakagami, S. Matsuda, A. Mitsui and K. Sano, “Improvement of a Human-Portable Underwater Robot for Soil Core Sampling,” **In:** *Proceedings of the 2014 Oceans-St. John’s. IEEE*, (2014) pp. 1–6.
[8] K. Isaka, K. Tsumura, T. Watanabe, W. Toyama, M. Sugesawa, Y. Yamada, H. Yoshida and T. Nakamura, “Development of underwater drilling robot based on earthworm locomotion,” *IEEE Access* **7**, 103127–103141 (2019).
[9] K. Isaka, K. Tsumura, T. Watanabe, W. Toyama, M. Okui, H. Yoshida and T. Nakamura, “Soil discharging mechanism utilizing water jetting to improve excavation depth for seabed drilling explorer,” *IEEE Access* **8**, 28560–28570 (2020).
[10] Y. Feng and J. Wang, “GPS RTK performance characteristics and analysis,” *J Glob Position Syst* **7**(1), 1–8 (2008).
[11] B. Siciliano, L. Sciavicco, L. Villani and G. Oriolo, *Robotics: Modelling, Planning and Control* (Springer Verlag, 2009).
[12] G. Oriolo, A. De Luca and M. Vendittelli, “WMR control via dynamic feedback linearization: Design, implementation, and experimental validation,” *IEEE Trans Contr Syst Tech* **10**(6), 835–852 (2002).
[13] F. Borrelli, A. Bemporad and M. Morari, *Predictive Control for Linear and Hybrid Systems* (Cambridge University Press, 2017).

Cite this article: D. Di Vito, P. Di Lillo, F. Arrichiello, C. Ferone, R. Amico and G. Antonelli, “Control framework of the ROBILAUT soil sampling robot: system overview and experimental results”, Robotica. https://doi.org/10.1017/S0263574724000857