Pedipulation in Quadruped Robots Using a Heuristic Inverse Kinematics Solver

Marco Tabita, Carmine Tommaso Recchiuto, Enrico Simetti, and Antonio Sgorbissa

Abstract— **Quadruped robots have a wide range of application domains, primarily focused on autonomous exploration and monitoring tasks. Locomotion and manipulation represent the two principal research areas associated with this class of robots. Recently, pedipulation, which consists of executing manipulation tasks using one of the robot's limbs, has emerged as a novel research direction, aiming to provide these robots with a level of dexterity comparable to that of their biological counterparts, while also enhancing their potential for physical and social interaction with humans in everyday environments. Performing pedipulation tasks presents considerable challenges in terms of control and coordination, primarily due to the limited degrees of freedom of the limbs and the complexity involved in fully exploiting their kinematic capabilities. This paper investigates the integration of a heuristic inverse kinematics solver within a model-based, task-priority control framework for executing pedipulation tasks of initial complexity. The proposed approach is validated through both simulation studies and real-world experiments conducted on a quadruped robotic platform.**

I. Introduction

Among mobile robots, quadrupeds have distinguished themselves by offering an effective balance of agility, stability, and compact dimensions. These characteristics make them highly suitable for specialized tasks such as exploration, search and rescue, and monitoring across diverse environments ranging from urban and industrial settings to natural landscapes.

To fully realize their potential and closely emulate their biological counterparts, these robotic systems require advanced locomotion capabilities coupled with sophisticated manipulation skills. Specifically, this involves using legs not merely for environmental interaction related to locomotion, but also for approaching obstacles and executing non-prehensile manipulation tasks aimed at enhancing exploration and operational versatility. Pushing a light obstacle out of its way or opening a door to enter a room—either by pushing it if it is already half-open or by operating the handle—are two key examples. This form of manipulation, known as "pedipulation," represents an emerging research area within quadruped robotics that significantly expands the robot's functional repertoire.

Pedipulation not only enhances robot autonomy in complex scenarios but can also positively influence human-robot interaction in social environments. Actual dogs interact socially with people using their paws. For example, a dog might send a ball back to the person who threw it; place its paws on its owner's hand to establish physical contact, reinforcing an emotional bond or requesting petting, playtime, or food; offer a paw for "shaking hands" as a trained behavior to show trust and connection; or even, in the case of a pointing dog, stop and stand still with one front paw lifted, body tense, and nose directed toward the game. Enabling quadrupedal robots to use their bodies to socially communicate with humans can significantly enhance their social impact and increase their chances of being accepted as companions in everyday environments.

From a control perspective, quadruped robots inherently involve significant complexity due to their dynamic behaviors and numerous degrees of freedom. Such complexity manifests itself in the robot's ability to dynamically coordinate intricate movements in response to environmental interactions. Moreover, the associated dynamics are hybrid, underactuated, and highly nonlinear, further complicating control tasks. As usual, the solutions to these control tasks typically fall into two categories: model-based and model-free approaches. While model-based approaches offer enhanced stability and precision, they are generally computationally demanding and energy-intensive. Conversely, model-free approaches, although less precise, have the advantage of lower computational cost and greater generalization capabilities, making them particularly suitable for long-term operational objectives.

Inverse kinematics (IK) solvers play a critical role in model-based applications, particularly regarding motion planning and manipulation tasks. The precision in controlling and accurately positioning robotic limbs is crucial for applications demanding efficiency, accuracy, and autonomy. Among the various IK methods, Forward And Backward Reaching Inverse Kinematics (FABRIK) [1] has established itself as a particularly efficient technique due to its simplicity, rapid convergence rate, and minimal computational requirements compared to other analytical solutions. However, in its original formulation, this algorithm presents limitations when applied to robotic manipulators characterized by joints with restricted degrees of freedom and strict kinematic constraints.

In this paper, we propose a model-based controller that implements FABRIK, specifically tailored to handle complex joint configurations with kinematic constraints, such as joint angle limitations, in the context of pedipulation tasks. This adaptation ensures valid joint configurations, crucial for seamless integration into a comprehensive robotic control framework. The proposed method has been tested both in simulations and real-world scenarios employing the quadruped robot Go1 by Unitree, thus demonstrating its

All authors are with Department of Informatics, Bioengineering, Robotics and Systems Engineering, University of Genoa, Via All'Opera Pia 13, 16145 Genoa, Italy

Corresponding author: Marco Tabita, marco.tabita@edu.unige.it

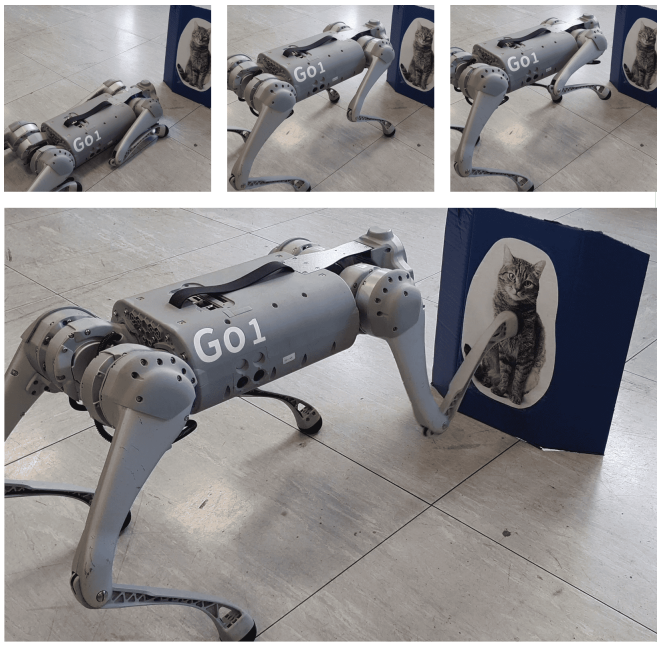

Fig. 1. The robot Go1 interacting with a target standing on three legs.

effectiveness and practical applicability in pedipulation tasks.

The remainder of this paper is structured as follows: Section II provides background information on quadruped robot control and introduces the FABRIK algorithm. Section III details the adaptation of the IK solver and its integration within the proposed control framework, along with a comprehensive description of the framework itself. Section IV presents the conducted experiments, reports the obtained results, and provides an accompanying discussion. Finally, Section V concludes the paper, offering closing remarks and outlining directions for future work.

II. Background

The control problem for quadruped robots has traditionally been approached primarily through model-based control systems [2], [3], [4], [5], [6], [7]. These systems typically employ hierarchical prioritized tasks [8], [3], establishing a structured order of objectives aimed at ensuring safety, stability, mission-specific targets, and optimality criteria. Although this hierarchical framework allows for stable and precise whole-body solutions, it presents the drawback that control solutions are computed at execution time, thus not optimizing their long-term impact.

Model Predictive Control (MPC) methods [4] have been introduced to overcome this limitation. MPC formulates optimal control strategies over a defined future time horizon, replacing hierarchical constraints with non-hierarchical optimization.

A diametrically opposite approach consists of employing model-free controllers, where reinforcement learning techniques are used to train controllers capable of performing even highly complex tasks [9], [10], [11], [12], [13], [14].

When extending quadruped robots to perform manipulation tasks, various strategies have been explored. A common approach involves integrating an additional manipulator limb [4], [15], facilitating complex operations, and leveraging the robot's legs to extend its workspace [16]. However, this strategy significantly increases the control complexity, degrees of freedom, robot dimensions, and energy consumption, thereby reducing overall autonomy. Additionally, adding a manipulator arm to the back of a robotic dog creates a hybrid with no biological counterpart, which some people may perceive as uncanny, potentially hindering its acceptance in social environments.

Alternatively, methods without an extra limb predominantly involve non-prehensile manipulation, with some exceptions that involve attaching an additional gripper directly to one or more of the quadruped's legs [11], employing all four legs while the robot is positioned on its back [17], or finally using the robot's body itself to interact with objects [5]. More recently, the concept of pedipulation has emerged, which refers to manipulation tasks performed using one of the robot's legs. This technique has predominantly been explored using model-free methods. However, there are also cases where model-based controls are employed at the low-level execution stage, integrating reinforcement learning specifically for the manipulation planning task [13].

Regardless of the adopted approach, IK solutions play a fundamental role in enabling robotic manipulation tasks. Within this context, the FABRIK algorithm has gained considerable attention due to its intrinsic simplicity, computational efficiency, and ability to generate smooth trajectories, particularly within computer graphics and animation applications [1]. Nevertheless, the original formulation of the IK-solver presents notable limitations when directly applied to robotic manipulation scenarios, especially for kinematic chains composed exclusively of single-degree-of-freedom joints. As a result, multiple studies have aimed to extend the original algorithm to better accommodate robotic applications, leveraging its inherent simplicity and effectiveness [18], [19], [20], [21].

Among these extensions, the work presented by Santos et al. [18] introduced FABRIK-R, an approach that incorporates joint constraints by projecting joint movements onto predefined planes, taking into consideration the interdependencies and limitations of parent-child joints. Notably, this solution primarily addresses orientation constraints without explicitly integrating joint positional limits. Additionally, Aristidou et al. [21] significantly advanced the solver by including constraints relevant to anthropometric and robotic joint models and proposing optimization techniques for handling unreachable targets effectively.

Finally, expanding further on the applicability of the solver, Santos et al. [20] developed M-FABRIK, a novel inverse kinematics algorithm specifically designed for mobile manipulators. This method not only demonstrates its adaptability but also introduces dynamic repositioning capabilities of the manipulator's base, thus reinforcing the algorithm's robustness and flexibility across different scenarios.

III. METHODOLOGY

A. Adapting FABRIK for pedipulation

Unlike traditional methods that rely on Jacobian matrices, FABRIK avoids directly computing partial derivatives or complex matrix calculations. Instead, it iteratively updates joint positions using simple geometric operations, such as projecting points onto planes and lines, while maintaining accurate link distances. The base algorithm [1] operates through two sequential phases: forward reaching and backward reaching. In the forward phase, joint positions are adjusted from the end-effector inward toward the base, consistently adhering to link length constraints. The backward phase reverses this procedure, starting from the base and moving outward towards the end-effector to enforce the same geometric constraints. By alternately executing these two phases until convergence is reached, the result consists of a suitable inverse kinematics solution, frequently producing visually smooth joint trajectories.

This algorithm was originally developed for computer graphics applications. Consequently, it presents certain limitations when applied to robotics, such as not inherently accounting for the constraints introduced by specific joint types, the relationship between sequential joints, and the angular limits imposed by the robot's physical structure. These limitations have prompted numerous researchers to further develop the algorithm and adapt it specifically for particular robotic applications. The intrinsic simplicity and easily extensible structure of this kind of approach facilitate such modifications and extensions.

To make this solver suitable for our specific application of controlling the leg of a quadruped robot, it is necessary to integrate the constraints defined by the robot's rotational joints into the algorithm. The modifications implemented in our approach were inspired by the work presented in [18], and [19], where this kind of solver was adapted for inverse kinematics problems in spatial manipulation tasks. In that study, the modifications primarily focused on maintaining the manipulator's center of mass fixed in the inertial frame while complying with the kinematic constraints of each joint.

In our case, the objective is not to maintain a fixed center of mass, but rather to appropriately manage the kinematic constraints of the joints. Essentially, our modification retains the original forward-reaching phase, thereby obtaining what can be considered the ideal solution, which is subsequently adjusted during the backward-reaching phase by projecting each joint onto the corresponding constraint plane of the previous one. During each joint repositioning, it is verified that the resulting increment does not exceed the joint limits, and if this occurs, the value is saturated accordingly.

The backward-reaching phase, in this way, results in slower convergence of the end-effector towards the target, introducing the risk of falling into a local minimum and thus preventing convergence to the desired target. To mitigate this slowdown, an additional phase called the compensation phase has been introduced. This modification aims to reduce the distance between the end-effector and the target by calculating a projected contribution within the corresponding constraint plane for each joint, traversing the kinematic chain in the same direction as the forward-reaching phase. Throughout this process, joint values are consistently checked and saturated according to their respective joint limits. The pseudo code of this phase is reported in the Algorithm 1. For each joint position, starting from the end-effector towards the base, the projection matrix Φ_i is computed, as indicated in Line 2. This matrix projects a point expressed with respect to the global reference frame $< w >$ onto the constraint plane associated with the *i*-th joint, relative to its local reference frame. This operation enables an easy computation of the relative angle-axis rotation ρ between the projected positions of the end-effector, ${}^j r_e$, and the target, ${}^j r_t$, by exploiting the properties of vector and scalar products. The resulting ρ vector is obtained in Line 5. In the presence of a possible correction, which means that a corresponding rotation magnitude $\Delta\theta$ is greater than a minimum threshold, this contribution is computed by re-projecting ρ onto the corresponding joint's axis of rotation k_j as a safety check Line 7 and saturated at Line 11. Finally, at Line 12, the positions of the subsequent joint frames following the i-th joint are updated accordingly.

Algorithm 1 Compensation Phase

1: **for** $i = n-1$ to 1 **do**
2: $\quad \Phi_i \leftarrow computeProjectionMatrix(i)$
3: $\quad {}^j r_t \leftarrow \Phi_i \cdot {}^w r_t$
4: $\quad {}^j r_e \leftarrow \Phi_i \cdot {}^w r_e$
5: $\quad \rho \leftarrow computeRotation({}^j r_e, {}^j r_t)$
6: $\quad$ **if** $\text{norm}(\rho) > 10^{-4}$ **then**
7: $\quad\quad \Delta\theta \leftarrow \rho \cdot k_j$
8: $\quad$ **else**
9: $\quad\quad \Delta\theta \leftarrow 0.0$
10: $\quad$ **end if**
11: $\quad \theta_i \leftarrow saturate(\Delta\theta, \theta_i, \bar{\theta}_i, \underline{\theta}_i)$
12: $\quad updateJointsPosition(i, \theta_i)$
13: **end for**

B. Task priority architecture

This section outlines the general architecture of the proposed pedipulation framework (Figure 2), highlighting its key modules, which include the Main Loop that manages all components, the Mission Manager, and the IK Solver. The remaining components are: the Unitree IO-Interface, the Unitree State Estimator, the Robot Model, and auxiliary structures that support execution and visualization.

Given the necessity of having a modular framework capable of autonomously executing pedipulation tasks, we developed a solution whose core consists of a Task Priority controller implemented using the TSID library [22]. This choice was driven by the efficiency, versatility, and robustness of this type of low-level control, as well as TSID's compatibility with Pinocchio [23], a library commonly used in other legged systems applications for efficiently computing the dynamics (and derivatives) of a robot model.

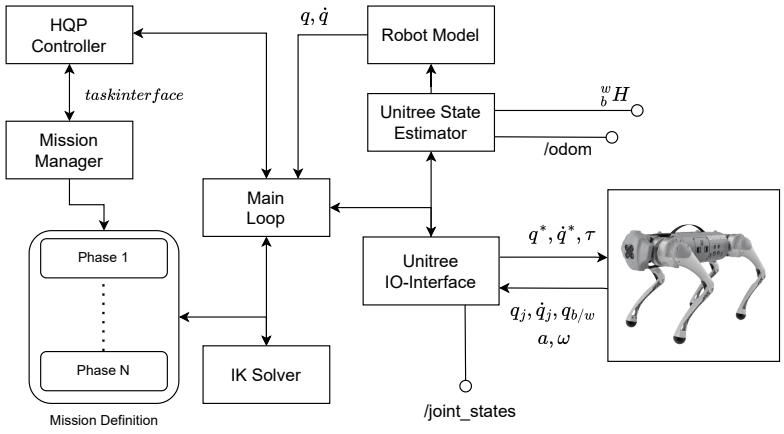

Fig. 2. Control framework architecture: the structure that collects all modules is the Main Loop, the Unitree IO-Interface, and the Unitree State Estimator are provided by the Unitree repository.

We defined in the HQP controller a set of tasks specifically dedicated to pedipulation, temporarily excluding aspects related to locomotion, gait generation, and trajectory optimization. This library allows for the definition of inverse dynamics tasks, representing objectives such as equalities and inequalities to be satisfied according to a hierarchical order. The general structure of such a task is provided in Equation 1:

$$J\ddot{y} = \ddot{x}^* - K_p(x - x^*) - K_d(\dot{x} - \dot{x}^*) - \dot{J}y, \tag{1}$$

where y is the control vector, J is the Jacobian matrix, x is the controlled variable and x^* is the task reference. The following tasks were then implemented and prioritized in the following order:

1) Base dynamics Equality Task
2) Balancing Inequality Task
3) Contact Point Tasks for feet in stance, comprising the no-motion equality constraint and force inequality constraint to prevent slippage
4) Body Position and Attitude Equality Task
5) Joint Posture Equality Task for the pedipulation leg

The first task (1) models the dynamics associated with the unactuated Degrees Of Freedom (DoFs) of the floating base. This task is fundamental for ensuring that the generated command is dynamically feasible.

The balancing inequality task (2) defined in the library constrains the capture point introduced in [24] within a range approximately corresponding to the support polygon defined by ground contacts. To enhance versatility, this task was modified to allow the capture point to be constrained within a defined "safety polygon". This safety polygon is expressed in a dedicated reference frame that can be dynamically adjusted at runtime based on the orientation of the support polygon in the world. This modification ensures a more robust constraint on the capture point's motion, assuring that it remains within the primary support polygon even when there are only three feet simultaneously in contact with the ground. The parameters defining the safety polygon, including its size, position, and the orientation of the dedicated frame, are autonomously set by a module named the Mission Manager, which will be discussed in detail subsequently.

Regarding constraints due to rigid ground contacts (3), these are easily managed by the TSID library, which allows the addition and removal of contacts from the main control problem formulation during execution. This is accomplished by defining key parameters, including the contact position, friction coefficient, and the ground normal vector at the contact point.

The task responsible for controlling the pose of the body frame in the world (4) ensures that the controlled variables match the defined reference up to the second derivative.

Similarly, the posture task for the pedipulation leg (5) operates in the same manner, with the Mission Manager handling the addition and removal of the task for the selected leg whenever necessary within the main formulation.

The complete control formulation is subsequently translated into a Hierarchical Quadratic Programming (HQP) problem, whose solution yields the control signal to be applied. Due to the limitations imposed by TSID, the HQP solver implemented can handle only two priority levels, forcing us to group the defined tasks into two main groups. Without focusing on the trajectory optimization of the body frame, position, velocity, and acceleration references are computed by solving a fifth-order polynomial equation, as shown in Equation 2, assuming zero velocity and acceleration at the initial and final time. This approach is employed for both the body frame and the foot pose trajectories. The computed reference is provided to the controller in Equation 1 and sampled at each control iteration.

$$\begin{cases} x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5, \\ \dot{x}(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4, \\ \ddot{x}(t) = 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3. \end{cases} \tag{2}$$

The Mission Manager is the module responsible for orchestrating the execution of user-defined missions, according to the intended robot operations. Each mission phase is represented as a distinct state, equipped with methods to update task references, monitor the progress of the phase, and assess its successful completion or potential failure. The Mission Manager continuously supervises the state of each phase and manages transitions between them accordingly. The task interface provided by the HQP Controller module allows for setting the desired task references and retrieving the corresponding task errors. Additional methods are then integrated to support visualization and terminal logging functionalities.

Currently implemented mission phases include:

- Body motion towards a target position.
- Body rotation to achieve a desired attitude.
- Transitioning a leg between support and pedipulation modes.
- Target-reaching for the designated foot.

By sequentially combining these distinct phases, it is possible to construct a wide variety of robot actions, ranging in complexity.

The IK Solver is the module responsible for solving the inverse kinematics problem. By accessing the Pinocchio model, the kinematic chain of interest (one of the robot's four limbs) is isolated. Provided a solution exists, the solver returns the corresponding joint configuration of the selected limb that achieves the target pose specified by the current mission phase in the world frame. In case of a non-existent solution, the corresponding mission is considered failed.

IV. Experiments

To validate the modified solver, preliminary experiments were initially conducted using MATLAB, followed by simulations in ROS Gazebo, and finally through real-world testing on the physical robot.

To test the IK Solver, the context was reconstructed in MATLAB to closely mimic the standing configuration of a quadruped robot leg, with reference dimensions derived from the Go1 quadruped platform. The selected limb for reference was the front right leg, though this choice does not affect the correct functioning of the algorithm. Proceeding with the experiment setup, 200 random targets were sampled within a $30 \times 30 \times 30$ cm volume located 20 cm below and 10 cm in front of the shoulder.

The case studies for validating the control framework in simulation and with the Unitree Go1 were chosen by considering scenarios that might frequently be encountered in daily life. Below are three examples considered:

- Pushing objects. In certain situations, some obstacles can be easily managed by moving them out of the robot's way. In these cases, it can be useful, or even necessary, for the robot to interact with environmental objects through pedipulation, such as pushing a light object or opening a door (Figures 1 and 6).
- Removing objects from one's back. During common routines, objects may accidentally fall onto the robot's back or be improperly placed on it by a user with malevolent intent. Removing these objects using pedipulation is feasible, provided joint limits permit such motions. This case is particularly prone to collisions between the calculated trajectory and the robot's body (Figure 8).
- Obstacle sensing. If pressure sensors are installed on the robot's feet, and provided they have sufficient sensitivity and proper calibration, they can be used not only to detect contacts but also to analyze potential obstacles encountered during exploration tasks. Understanding the nature of an obstacle, particularly whether it is mobile or fixed, and soft or rigid, is crucial for subsequent interactions (Figure 5).
- Social interaction. Some gestures can be key to socially interacting with people, rather than physically altering the environment. An example is giving a paw to a person to show trust and connection, which may increase the robot's acceptability in everyday situations (Figure 7).

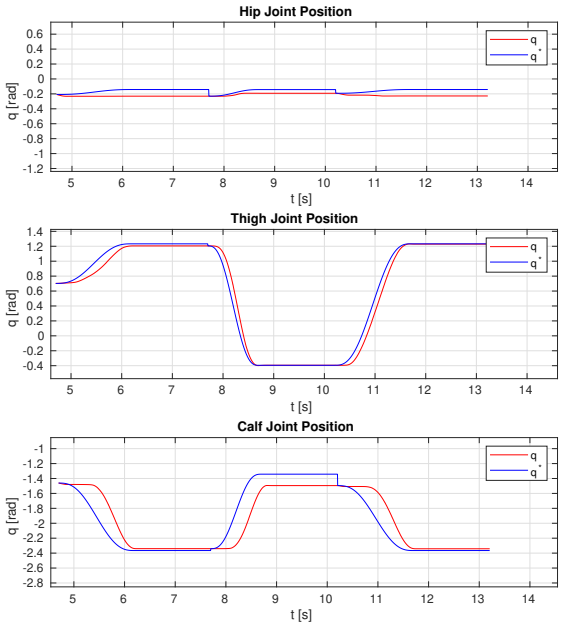

Fig. 3. Corresponding joints positions during the phases of pedipulation in the mission of hitting a target (Figure 1). The positions in radians measured from the robot's joint encoders are indicated in red, while the desired joint positions computed are in blue.

As reported in Section III-B, experiments defined as autonomous missions consist of a series of main phases that must be completed, otherwise, the mission fails. The defined missions share a similar structure, beginning with the robot lying on the ground. Subsequently, the robot stands up, adjusts its posture to ensure adequate stability to the resulting support polygon, and then lifts the designated leg. The leg starting working position is located 15 cm below its shoulder. From this point, all target poses are sequentially reached to perform the required actions. The mission concludes with the robot placing the leg back on the ground.

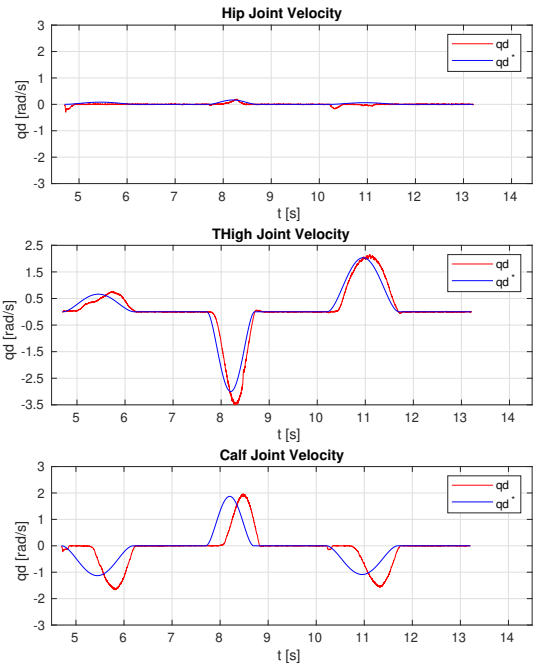

Fig. 4. Joints velocities during the phases of pedipulation in the mission of hitting a target (Figure 1). The velocities in radians per second measured from the joint encoders of the robot are reported in red. The desired angle rates computed are in blue.

A. Results

The results obtained from testing the IK Solver in MATLAB show that 180 out of 200 targets were successfully reached, with an average positioning error of 4 mm and an average convergence rate of 1.76 iterations. Upon further analysis of the 20 targets that were not reached, it was found that 19 were physically unreachable due to exceeding the limb's positional limits, given the defined threshold of 1 cm.

Pedipulation case studies were implemented both in simulation and with the Unitree Go1 robot. In the following, for brevity, we will present only results with the actual robot. All defined missions were completed, with the robot effectively using each of its four limbs to perform pedipulation tasks as needed. Specific tasks successfully performed in real scenario included opening an office door, showing a foot positioning error of 0.059 m; interacting with a cardboard cat placed around the robot (in front, behind, and on its back) with positioning errors of 0.032 m, 0.031 m, and 0.075 m, respectively; interacting with a human 0.062 and reaching various targets to inspect soft and fixed obstacles, with a mean error of 0.041 m[1] Graphs in Figures 3 and 4 illustrate joint position and velocity trends for the pedipulation leg during relevant phases of interaction with the cardboard cat in Figure 1, excluding preliminary body movements. The plots demonstrate satisfactory convergence, although static errors are visible in certain instances, likely attributable to limitations in the model and the QP solver used by TSID.

[1]Videos documenting all conducted experiments are available at the following link: `https://youtu.be/rXkSEL_xMx8`

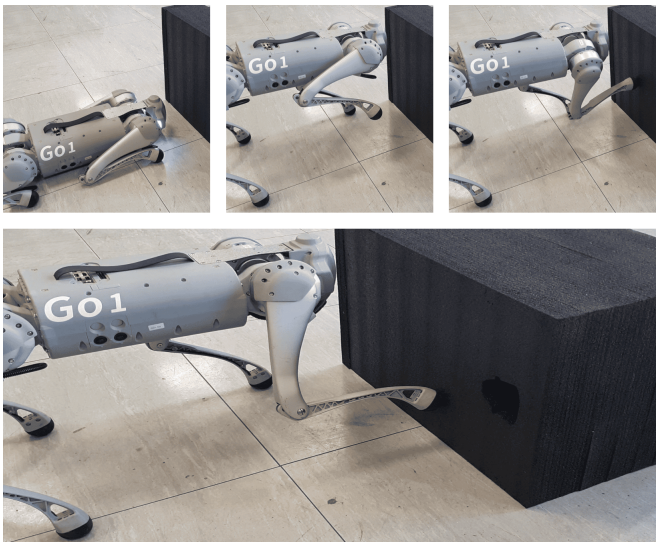

Fig. 5. The robot inspects an obstacle in front of it using the front right leg.

Fig. 6. The robot uses the front right leg to opens a half-closed door, making it possible to pass.

V. Discussion

From the results obtained while testing the IK solver in MATLAB simulations, the initially unreachable target was subsequently identified as actually reachable. This error was due to the target being located in a critical area where the forward phase positioned the elbow in a location that, upon projection, returned it to its previous iteration point, leaving the convergence error unchanged. This highlights that, despite the proposed modifications, certain points remain challenging to reach. Nonetheless, such challenging targets represent uncommon scenarios and do not impede the practical applicability of the proposed methods. The results

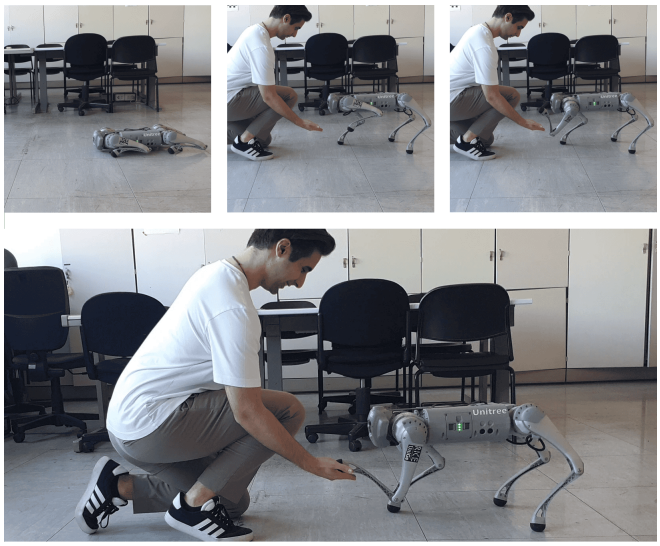

Fig. 7. The robot gives its paw to the human standing in front of it.

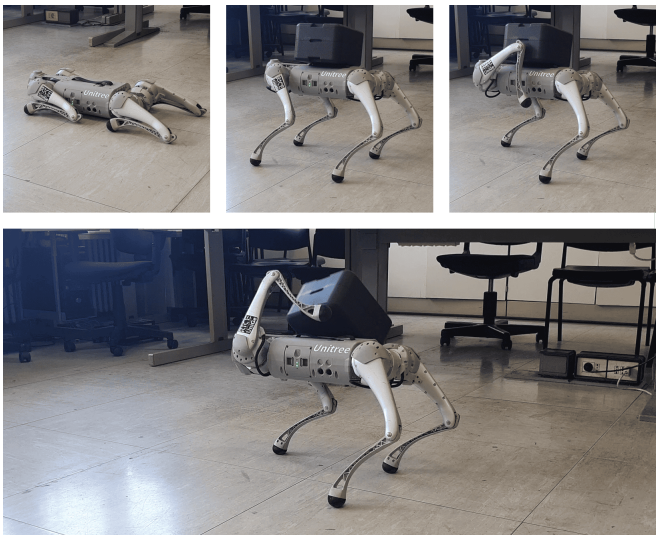

Fig. 8. The robot uses the front left leg to remove an object on its back.

also showed that the initial configuration of the leg plays a crucial role in determining the quality of the solution. It was observed that an extended arm configuration tends to yield the best performance in terms of success in finding the solution. Nevertheless, the choice of the initial configuration does not affect the feasibility of the resulting motion. As long as all joint constraints are strictly respected and the reference for the corresponding task is defined in joint space, any valid solution remains reachable from any other admissible configuration.

Regarding case studies in simulation and with the actual Go1 robot, it should be noted that target definition strictly depends on the type of action defined by the user. Since modules for obstacle detection and avoidance are currently absent, users are responsible for defining which targets to reach and their positions.

It is also worth noting that this approach aims to achieve pedipulation tasks by using the quadruped robot's leg as if it were a fixed-base manipulator. During execution, the framework calculates the desired end-effector position, formulates the inverse kinematics problem, and, upon solving it, determines the trajectory to be followed in joint space. This methodology is sensitive to base displacements; therefore, the base equality task for this type of action is set to minimize base movement as much as possible. In practice, the Unitree State Estimator exhibited insufficient precision, resulting in drift that negatively impacted the convergence of the end-effector toward the goal. Nevertheless, the results obtained are highly encouraging, showing relatively low positioning errors. Higher errors were recorded during more complex movements, such as removing an object from the robot's back and opening a door, with the latter naturally increasing resistance and thus amplifying the error.

VI. Conclusions And Future Work

In this work, we presented an architecture that relies on a heuristic inverse kinematics solver specifically designed for pedipulation tasks.

This form of pedipulation enables a higher level of interaction for this class of quadruped robots, which are increasingly being considered for operation in urban and unstructured environments. In addition to providing solutions for interacting more effectively with human-inhabited environments (e.g., by pushing light objects out of its way or opening doors), pedipulation may increase the social acceptability of robotic dogs by fostering an emotional bond with their owners.

The results from simulations and with an actual Go1 robot demonstrate that the implemented architecture was effective in the majority of the proposed scenarios. However, certain cases revealed vulnerabilities in handling specific target configurations, both in terms of target positions to be reached and initial configurations from which the problem is solved, highlighting directions for future work.

Exploring alternative control formulations for this type of manipulation will be of great interest. In particular, by reformulating the problem to integrate potential contributions from the remaining available degrees of freedom, it would be possible to achieve true whole-body pedipulation, thereby further enhancing the robot's capabilities to interact with both environments and people.

References

[1] A. Aristidou and J. Lasenby, "Fabrik: A fast, iterative solver for the inverse kinematics problem," *Graphical Models*, vol. 73, pp. 243–260, 9 2011.

[2] F. Farshidian, E. Jelavi´c, A. Satapathy, M. Giftthaler, and J. Buchli, *Real-Time Motion Planning of Legged Robots: A Model Predictive Control Approach*. IEEE, 2018.

[3] D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, *ALMA-Articulated Locomotion and Manipulation for a Torque-Controllable Robot*, 2019.

[4] J.-P. Sleiman, F. Farshidian, M. V. Minniti, and M. Hutter, "A unified mpc framework for whole-body dynamic locomotion and manipulation," 3 2021. [Online]. Available: http://arxiv.org/abs/2103.00946

[5] M. V. Minniti, R. Grandia, F. Farshidian, and M. Hutter, "Adaptive clf-mpc with application to quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 7, pp. 565–572, 1 2022.

[6] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback mpc for torque-controlled legged robots," 5 2019. [Online]. Available: http://arxiv.org/abs/1905.06144

[7] G. Xin, W. Wolfslag, H. C. Lin, C. Tiseo, and M. Mistry, "An optimization-based locomotion controller for quadruped robots leveraging cartesian impedance control," *Frontiers in Robotics and AI*, vol. 7, 4 2020.

[8] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model predictive control," 8 2022. [Online]. Available: http://arxiv.org/abs/2208.08373

[9] Y. Ouyang, J. Li, Y. Li, Z. Li, C. Yu, K. Sreenath, and Y. Wu, "Long-horizon locomotion and manipulation on a quadrupedal robot with large language models," 4 2024. [Online]. Available: http://arxiv.org/abs/2404.05291

[10] X. Cheng, A. Kumar, and D. Pathak, "Legs as manipulator: Pushing quadrupedal agility beyond locomotion," 3 2023. [Online]. Available: http://arxiv.org/abs/2303.11330

[11] C. Lin, X. Liu, Y. Yang, Y. Niu, W. Yu, T. Zhang, J. Tan, B. Boots, and D. Zhao, "Locoman: Advancing versatile quadrupedal dexterity with lightweight loco-manipulators," 3 2024. [Online]. Available: http://arxiv.org/abs/2403.18197

[12] S. Jeon, M. Jung, S. Choi, B. Kim, and J. Hwangbo, "Learning whole-body manipulation for quadrupedal robot," *IEEE Robotics and Automation Letters*, vol. 9, pp. 699–706, 1 2024.

[13] Z. He, K. Lei, Y. Ze, K. Sreenath, Z. Li, and H. Xu, "Learning visual quadrupedal loco-manipulation from demonstrations," 3 2024. [Online]. Available: http://arxiv.org/abs/2403.20328

[14] Y. Ji, G. B. Margolis, and P. Agrawal, "Dribblebot: Dynamic legged manipulation in the wild," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2023-May. Institute of Electrical and Electronics Engineers Inc., 2023, pp. 5155–5162.

[15] S. Zimmermann, R. Poranne, and S. Coros, "Go fetch! - dynamic grasps using boston dynamics spot with external robotic arm," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2021-May. Institute of Electrical and Electronics Engineers Inc., 2021, pp. 1170–1176.

[16] M. Liu, Z. Chen, X. Cheng, Y. Ji, R.-Z. Qiu, R. Yang, and X. Wang, "Visual whole-body control for legged loco-manipulation," 3 2024. [Online]. Available: http://arxiv.org/abs/2403.16967

[17] F. Shi, T. Homberger, J. Lee, T. Miki, M. Zhao, F. Farshidian, K. Okada, M. Inaba, and M. Hutter, "Circus anymal: A quadruped learning dexterous manipulation with its limbs," 11 2020. [Online]. Available: http://arxiv.org/abs/2011.08811

[18] M. C. Santos, L. Molina, E. A. Carvalho, E. O. Freire, J. G. Carvalho, and P. C. Santos, "Fabrik-r: An extension developed based on fabrik for robotics manipulators," *IEEE Access*, vol. 9, pp. 53 423–53 435, 2021.

[19] G. Dong, P. Huang, Y. Wang, and R. Li, "A modified forward and backward reaching inverse kinematics based incremental control for space manipulators," *Chinese Journal of Aeronautics*, vol. 35, pp. 287–295, 12 2022.

[20] P. C. Santos, R. C. S. Freire, E. A. N. Carvalho, L. Molina, and E. O. Freire, "M-fabrik: A new inverse kinematics approach to mobile manipulator robots based on fabrik," *IEEE Access*, vol. 8, pp. 208 836–208 849, 2020.

[21] A. Aristidou, Y. Chrysanthou, and J. Lasenby, "Extending fabrik with model constraints," *Computer Animation and Virtual Worlds*, vol. 27, pp. 35–57, 1 2016.

[22] A. D. Prete, N. Mansard, O. E. Ramos, O. Stasse, and F. Nori, "Implementing torque control with high-ratio gear boxes and without joint-torque sensors," in *Int. Journal of Humanoid Robotics*, 2016, p. 1550044. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01136936/document

[23] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio c++ library-a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," Tech. Rep., 1 2019. [Online]. Available: https://laas.hal.science/hal-01866228v2

[24] O. E. Ramos, N. Mansard, and P. Soù, "Whole-body motion integrating the capture point in the operational space inverse dynamics control," Tech. Rep.