Analysis of Quadruped Robot Gaits in Push-and-Slide Interaction Tasks

Emanuele Cuzzocrea, Michele Avagnale, Pierluigi Arpenti, Fabio Ruggiero

***Abstract*—This work investigates the optimal configuration for performing a push-and-slide inspection task with a quadruped robot, focusing on gait type, base orientation, interaction force, and sliding velocity. To isolate the effect of motion type while the robot walks and slides parallel to a wall, a rigid stick is mounted on its base in place of a robotic arm. A control method combining classical control techniques and reinforcement learning is used to enable fair and consistent comparisons across different configurations. A novel gait learning method enforcing a predefined contact sequence is introduced while automatically optimizing all timing parameters during training. Extensive simulation trials are conducted on several morphologically diverse quadruped robots, followed by a four-way analysis of variance (ANOVA) to identify statistically significant performance differences across key metrics. The most effective motion configurations are then validated on real hardware.**

***Index Terms*—Legged Robots, Field Robots, Reinforcement Learning.**

I. Introduction

In recent years, quadruped robots have been increasingly employed in industrial environments for inspection tasks, replacing humans in hazardous and challenging conditions [1], [2]. This trend is largely driven by their advanced locomotion capabilities, which allow them to traverse complex and unstructured terrains with ease [3]–[5]. However, despite this rapid adoption, current deployments are still predominantly limited to *non-contact* sensing (e.g., visual and acoustic), with minimal or no physical interaction with the environment. In contrast, a substantial portion of industrial plant inspections requires *contact-based* Non-Destructive Testing (NDT) [6], i.e., a set of techniques used to assess the structural integrity of materials—such as concrete, metals, and coatings—without causing damage or requiring material removal. In many practical scenarios (e.g., thickness assessment and corrosion monitoring on long pipes, ducts, and large structures), the measurement quality critically depends on establishing and maintaining a reliable probe–surface coupling while traversing extended assets. Currently, these operations are typically performed manually by specialized operators or partially assisted by drones [7]–[9] to reach hazardous or high-altitude locations, yet they still require careful contact management and are difficult to sustain over long paths and durations. A quadrupedal robot capable of autonomously performing *contact-based scanning* could therefore significantly enhance automation, repeatability, and safety in dangerous or hard-to-reach areas, enabling persistent inspections over large industrial facilities and long linear infrastructures.

Manuscript received: January 2026.
Undisclosed projects acknowledgments.
Undisclosed authors affiliations and emails.

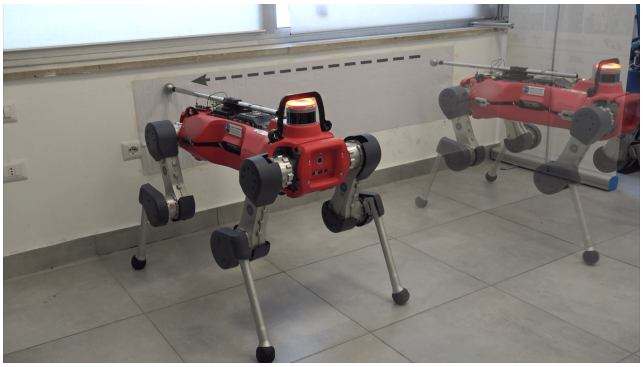

Fig. 1. A quadruped robot performing a push-and-slide task using a stick rigidly attached to its base. The transparent robot indicates the starting pose; the opaque one indicates the final pose. The dotted arrow denotes the sliding path of the end-effector on the wall.

Compared to wheeled and aerial robots, quadrupeds exhibit unique locomotion capabilities, such as walking with various gait patterns. While much of the existing research has focused on studying gaits in terms of stability and walking energy efficiency [10]–[12], less attention has been given to how the choice of gait and base orientation can optimize performance in loco-manipulation tasks. In this context, many existing approaches to loco-manipulation rely on the integration of a robotic arm mounted on the quadruped's base [13], [14]. While this extends the manipulation capabilities, it also introduces significant complexity in managing the overall system. Moreover, the addition of a robotic arm partially decouples locomotion from manipulation. Although such a separation can be advantageous in certain scenarios [15], it makes it difficult to determine the most effective leg movement strategies for executing physical tasks, as the locomotion system primarily serves to position the manipulator rather than actively contributing to the interaction process [16]. Besides, only a few market available solutions integrate a quadruped robot with a robotic arm so far.

By performing a comprehensive statistical analysis, this work aims to study the optimal motion configuration for a push-and-slide inspection task using a quadruped robot. A rigid stick attached to the robot's base is deployed, as shown in Fig. 1. This design choice ensures that locomotion directly influences manipulation capabilities. The key contributions of this work are: (i) The identification of the most effective gait pattern, base orientation, interaction forces, and sliding velocities for performing a push-and-slide task among the set of configurations considered, based on a comprehensive statistical analysis that includes four different robots; and (ii)

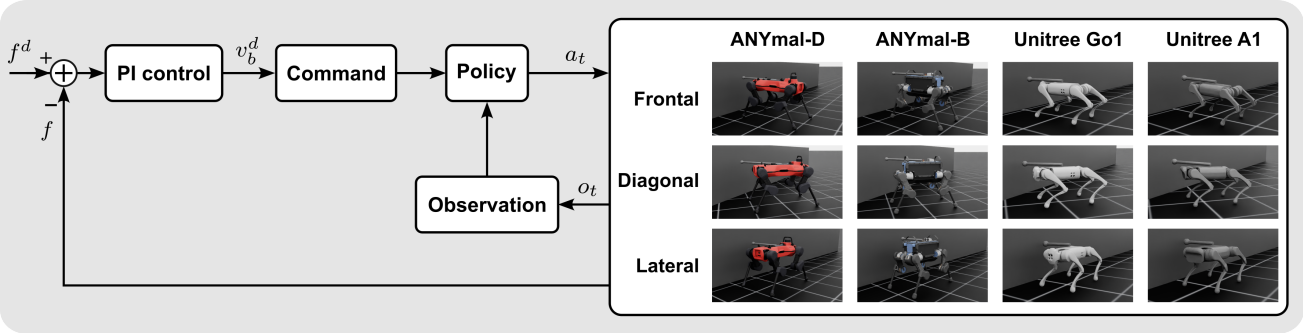

Fig. 2. Block diagram of the control approach used in this work, which combines classical control techniques with RL. The force error is transformed into base velocity commands by a PI controller. The policy receives these commands and, together with proprioceptive observations, outputs desired joint positions.

A novel gait learning approach that enforces a desired gait by specifying only the foot contact sequence, while timing parameters are automatically learned during training.

II. Related Work

Several studies in the literature explore the integration of classical control techniques with RL. In [16], a control framework is proposed for a legged mobile manipulator in which the legs are controlled via an RL policy, while the robotic arm is governed by model predictive control (MPC). In [17], a hybrid control framework that integrates model-based whole-body control with reinforcement learning is presented. The approach combines precise torque optimization with learned feedback corrections, enabling robust and accurate loco-manipulation tasks. In [18], a hybrid position-force control strategy is introduced for a fixed-base manipulator, where position control is handled by an RL policy and force control is managed by a PD controller. In all these works, the model-based and learning-based components either operate in a decoupled manner or the RL module serves as a residual correction or additional feedback. In contrast, the approach proposed in this paper features a coordinated cooperation between the two methods. Specifically, the RL policy learns to regulate contact forces by tracking reference velocities generated by a proportional-integral (PI) controller acting on real-time force feedback, as depicted in Fig. 2. Several recent works demonstrate that loco-manipulation tasks can be successfully executed without the use of articulated arms. For instance, in [19] quadruped robots are trained to use the front legs not only to walk, but also to climb walls, press buttons and move obstacles. In [20], a hierarchical learning-based framework that enables a quadrupedal robot to manipulate large and heavy objects directly with its floating base is proposed. In [21], prongs that are small, rigid extensions attached underneath the robot's base, are used. These allow the robot to stably rest its torso on the ground, enabling it to use two legs simultaneously for manipulation tasks. A learning-based method for a quadrupedal robot to climb ladders efficiently and reliably is presented in [22], where its standard point-contact feet are replaced with hooked end-effectors. Recent advances have demonstrated that RL can be effectively employed to generate locomotion gaits for legged robots. In [11], an RL-based gait generator is combined with a low-level MPC controller to enable autonomous transitions from a slow crawl to a trot, and eventually to a fly-trot gait as the reference velocity increases, aiming to enhance locomotion efficiency. In [23], central pattern generators are integrated with RL. A novel approach introduced in [12], termed RM-based locomotion learning, leverages reward machines (RM) to enable the learning of diverse gaits without relying on predefined trajectory references. However, these studies do not consider physical interactions with the environment and do not provide mechanisms for enforcing specific gait patterns without manually tuning all timing-related parameters.

The works most closely related to ours are the following. In [13], an RL controller capable of managing both end-effector position and contact forces for a legged manipulator is presented. In [24], the previous work is extended: force and position control are no longer handled independently, but a unified policy is proposed that integrates both, thereby enabling tasks such as push-and-slide. In [25], a control strategy designed to regulate force interactions directly through the floating base of a hydraulic legged robot is introduced. However, these studies do not address key challenges such as applying constant contact forces against a rigid surface during locomotion, or studying how gait strategies influence interaction capabilities—issues that are central to this paper.

III. Method

This work aims to statistically identify the most advantageous configurations for performing a push-and-slide inspection task using a quadruped robot. A rigid stick is mounted in place of a robotic arm to isolate the oscillations of the floating base across different configurations. The analysis varies gait pattern (trot, crawl, pace, bound), base orientation relative to the wall (frontal, diagonal, lateral), interaction force (0–60 N), and sliding velocity (0.0–0.2 m/s). For generalization purposes, four quadruped platforms with different characteristics are considered: ANYmal-D, ANYmal-B, Unitree Go1, and Unitree A1. All four robots differ in the size, shape, and weight of both the base and the legs. Moreover, the ANYmal robots feature inward-bent knee joints, whereas the Unitree robots exhibit alternating inward–outward knee orientations. Combined with their different actuation systems, these factors result in distinct stability, compliance, and interaction dynamics across platforms.

TABLE I
DEFINITION OF THE REWARD FUNCTION R (ANYMAL-D)

Description	Definition	Weight
Push-and-Slide Task		
Velocity tracking	$\exp(-\|v^d_{b,xy} - v_{b,xy}\|^2/0.15)$	$3.0dt$
Yaw tracking	$\exp(-\|0 - \psi\|^2/0.15)$	$2.0dt$
Keep wall contact	$\mathrm{I}(f > 0)$	$0.5dt$
Penalties		
Vertical velocity	$\|v_{b,z}\|^2$	$-2.0dt$
Angular velocity	$\|\omega_{b,xy}\|^2$	$-0.05dt$
Flat orientation	$\|g_z\|^2$	$-5.0dt$
Joint torques	$\|\tau_j\|^2$	$-2.5\mathrm{e}{-5}dt$
Joint acceleration	$\|\ddot{q}_j\|^2$	$-2.5\mathrm{e}{-7}dt$
Action rate	$\|a - a_{last}\|^2$	$-0.01dt$
Collisions	n_c	$-1.0dt$
Joint deviation	$\|q_{j,nom} - q_j\|$	$-1.0dt$

A. Observation and Action Space

The observation vector is defined as $o = (v_b, \psi, \omega_b, g_z, q_j - q_{j,nom}, \dot{q}_j, a_{last}, u) \in \mathbb{R}^{50}$, where $v_b \in \mathbb{R}^3$ and $\omega_b \in \mathbb{R}^3$ denote the base linear and angular velocities, respectively; $\psi \in \mathbb{R}$ is the base yaw expressed in radians; $g_z \in \mathbb{R}^3$ is the projected gravity vector; $q_j \in \mathbb{R}^{12}$ and $\dot{q}_j \in \mathbb{R}^{12}$ are joint positions and velocities, respectively; $q_{j,nom} \in \mathbb{R}^{12}$ are nominal joint positions; $a_{last} \in \mathbb{R}^{12}$ is the previous action; and $u \in \mathbb{R}^4$ is the current RM state, encoded as a one-hot vector (Section III-C). The desired yaw is always set to zero, since it is the wall that is rotated across different configurations—not the robot itself. Additionally, the policy receives the commanded planar linear velocities, $v^d_{b,xy} \in \mathbb{R}^2$, which are computed by the PI controller (Section III-D). The policy action $a \in \mathbb{R}^{12}$ represents joint deviations from the robot's nominal configuration. Target positions are computed as $q^d_j = \sigma_a a + q_{j,nom}$, where σ_a is a standard scaling factor. Specifically, σ_a is set to 0.5 for ANYmal platforms (i.e., ANYmal-D and ANYmal-B) and to 0.25 for Unitree platforms (i.e., Unitree Go1 and Unitree A1).

B. Reward Function

The reward function R for ANYmal-D is detailed in Table I. The first three terms correspond to the push-and-slide task, while the remaining penalties encourage safety and smoothness. The rewards are multiplied by dt to make the training independent of the simulation time step. To ensure the generality of the results, the first three terms were used unchanged for the other three robots. As for the penalties, all four robots employed the default terms and weights provided by Isaac Lab (with the exception of the joint deviation penalty which was added equally for all robots with the same weight). This setup minimizes sensitivity to hyperparameters: not only are four different robots tested, but each also operates with slightly different rewards. Using RL, the task can thus be encoded with only a few high-level parameters, allowing the policy to learn autonomously without being biased towards a specific gait, base orientation, or even the particular robotic platform being used. Notably, no explicit force-tracking term is included, *e.g.*, $\exp(-\|f^d - f\|^2/k)$ [13], where $f \in \mathbb{R}$ denotes the actual force, $f^d \in \mathbb{R}$ the desired force, and $k \in \mathbb{R}$ is a scalar gain. While such a term could potentially improve task performance, it makes valid comparisons across configurations very difficult, compromising statistical analysis. This occurs because the force and velocity rewards tend to compete strongly, and each training session fails to balance them evenly according to the reward weights (one often dominates the other in an unpredictable manner). A better approach is to regulate force indirectly through velocity control. Although this slightly reduces task performance, it allows for a single tracking reward (velocity) while managing forces via a PI controller. This setup is far better suited to assessing how different gaits influence force and velocity tracking, which constitutes the main objective of this paper.

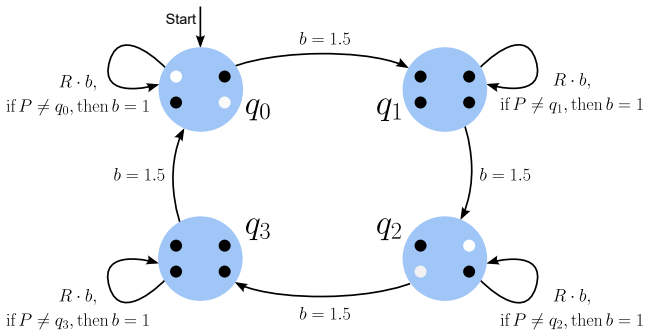

Fig. 3. Finite state automaton for the trot gait. In each state, black circles represent stance feet and white ones swing feet. The cycle has four states, including two intermediate ones (q_1 and q_3) where all feet contact the ground to help re-establish balance and reduce angular oscillations. The same strategy is applied to pace and bound gaits, resulting in all chosen gaits having four states for a fair comparison. The selected gaits capture all the main oscillation patterns that a quadruped robot can exhibit during locomotion: diagonal oscillations for the trot, lateral ones for the pace, frontal for the bound, and single-leg movements for the crawl.

C. Gait Learning

One of the main objectives of this work is to evaluate different gait patterns and identify the one that yields the best performance. To this end, predefined contact sequences are assumed, focusing on enforcing them effectively without requiring manual specification of timing-related parameters, such as gait cycle and phase durations. To address this challenge, a modified version of RM-based locomotion learning is proposed, inspired by [12]. In Fig. 3, the finite-state automaton used to enforce the trot gait is shown, while the automata used for crawl, pace, and bound are provided in the attached video. The variable $P = \{P_{FL}, P_{FR}, P_{BL}, P_{BR}\} \in \mathbb{R}^4$ is defined as a boolean vector indicating whether the front-left (FL), front-right (FR), back-left (BL), and back-right (BR) feet are in contact with the ground. Let $u \in \mathbb{R}^4$ be the current RM state, represented as a one-hot vector, and $q_i \in \mathbb{R}^4$ be the desired foot contact configuration for the i-th state (*e.g.*, $q_0 = \begin{bmatrix}1 & 0 & 0 & 1\end{bmatrix}^\top$), indicating that the FL and BR feet must be in contact. Different from [12], reward amplification is not applied during state transitions. Instead, it is applied during the preservation of a desired contact configuration associated with each state. Specifically, whenever the robot enters a new RM state, the amplification variable b is set to 1.5. If the robot keeps the required foot contact configuration (*i.e.*, if $P = q_i$), the overall reward is multiplied by this factor. However, if the robot loses the desired configuration even for

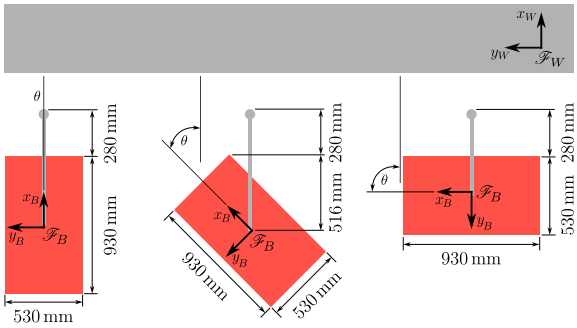

Fig. 4. Visual representation of the wall-aligned and robot-aligned reference frames for ANYmal-D, showing the angle θ between them, and the stick lengths for the frontal, diagonal, and lateral configurations.

a single time-step, b is immediately reset to 1. This strategy effectively forces the robot to chase the reward amplification by continuously progressing through the finite-state automaton and removes the need to predefine a fixed transition frequency between states. Also, a deliberately lower value of b than that used in [12] was employed, since the amplification does not occur sparsely during the transition from one state to another, but rather continuously. To the best of our knowledge, this is the first strategy that enables enforcing an arbitrary gait in RL training without requiring predefined timing parameters.

D. Hybrid Force-Velocity Control

The actual force control is handled by a simple PI controller, which computes the desired velocity towards the wall, required to apply the target force. As discussed in Section III-B, this approach was preferred over learning force tracking directly through the reward function. To define reference base velocities consistently, a wall-aligned reference frame $\mathscr{F}_W : \{O_W, x_W, y_W\}$ and a robot-aligned reference frame $\mathscr{F}_B : \{O_B, x_B, y_B\}$ are defined, both planar and parallel to the ground, as illustrated in Fig. 4. The angle between these two frames is denoted by $\theta \in \mathbb{R}$. Specifically, $\theta = 0°$ corresponds to the frontal configuration, $\theta = 45°$ to the diagonal configuration, and $\theta = 90°$ to the lateral configuration. Then, two velocities are defined, both expressed in $\mathscr{F}_W$: the *sliding velocity* $v_s \in \mathbb{R}$, which represents the velocity the robot should keep parallel to the wall, and the *pushing velocity* $v_p \in \mathbb{R}$, which is the output of the PI controller and is defined as

$$v_p = \mathrm{P}\,(f^d - f) + \mathrm{I} \int_0^T (f^d - f)\, dt, \tag{1}$$

where P and I are the proportional and integral gains, respectively. To express these velocities in $\mathscr{F}_B$ as desired base velocities $v^d_{b,xy}$, a proper rotation matrix depending on θ is applied.

E. Curriculum Learning

By leveraging the massive parallelization capabilities of the Isaac Lab framework [26], 4096 robots were trained simultaneously. To maximize performance, a two-phase training approach was adopted. In the first phase, lasting 1000 iterations, half of the robots are trained solely to walk in random directions at varying speeds, while the other half interacts with the wall by performing the push-and-slide task. After 1000 iterations, the second phase begins, during which all robots are trained to interact for another 500 iterations. This strategy proved particularly beneficial, as it first trains a fairly general neural network, followed by a specialization phase to maximize the target task.

F. Training Details

For robots required to walk during the first phase of training, a velocity is randomly sampled every 2 seconds within the range $[-0.5, 0.5]$ m/s for both $v^d_{b,x}$ and $v^d_{b,y}$. For robots required to interact with the environment, a wall is spawned perpendicular to the stick and positioned 0.15 m from its tip. As shown in Fig. 4, the stick extends the same length beyond the robot's base in all three orientations (280 mm for ANYmal-D, 240 mm for ANYmal-B, 195 mm for Unitree Go1, and 150 mm for Unitree A1). These lengths were chosen proportionally to the robot dimensions and also ensure a safe distance from the wall. The wall has a width of 10 m, a depth of 0.5 m, and a height of 1 m. It is not perfectly rigid, but is modeled with a stiffness coefficient of 5×10^5 N/m to improve contact stability between the stick and the wall. The sliding velocity v_s is randomly sampled in the range $[-0.2, 0.2]$ m/s, while the desired force is randomly chosen in the range $[0, 60]$ N for ANYmal robots and $[0, 30]$ N for Unitree robots (which are lighter).

Each episode has a maximum duration of 20 s but is terminated early if the robot's thighs come into contact with the environment (indicating a fall) or if a self-collision occurs. The low-level control runs at 400 Hz, while the policy is executed at 50 Hz. Each policy was trained for 1500 iterations. During training, the network is updated every 24 policy steps per environment, resulting in a batch size of 98304 due to parallelization. Proximal Policy Optimization (PPO) [27] is exploited as the learning algorithm, using the hyperparameters from [28]. Both the actor and critic are modeled as three-layer multilayer perceptrons (MLPs) with 128 hidden units per layer and exponential linear units (ELUs) as activation functions. Simulations were conducted on an Intel i9-14900HX CPU with an NVIDIA RTX 4090 GPU, with each policy requiring approximately 30 minutes of training time.

IV. Results and Discussion

A. Simulation Results and Statistical Analysis

A total of 48 training runs were performed (12 for each of the four robots), as all force and sliding velocity values can be addressed within a single training. For the statistical analysis, three different force values were selected (10 N, 35 N, and 60 N for ANYmal robots, and 10 N, 20 N, and 30 N for Unitree robots), along with three sliding velocities (0.05 m/s, 0.1 m/s, and 0.2 m/s). To test all possible combinations of these parameters, a total of 432 different tests were carried out (108 per robot). Static and dynamic friction coefficients of the wall were set to 0.5. A PI controller with gains $\mathrm{P} = 1 \times 10^{-5}$ and $\mathrm{I} = 5 \times 10^{-6}$ was used in all simulations. These low gains ensure a constant pushing velocity v_p towards the wall in the absence of disturbances. Choosing even lower gains

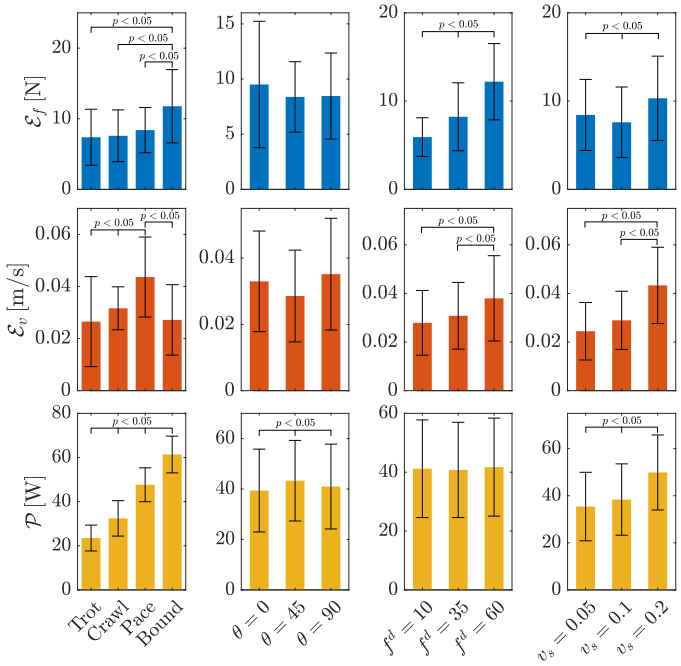

Fig. 5. Performance evaluation results for the ANYmal-D robot. Top: average force tracking error $\mathcal{E}_f$; Middle: average velocity tracking error $\mathcal{E}_v$; Bottom: average power consumption $\mathcal{P}$. Statistically significant differences between levels ($p < 0.05$) are indicated.

does not affect performance but simply increases the transient time. Tuning the gains to maximize performance for a specific configuration would have made comparisons across different configurations meaningless.

1) Statistical Analysis: Three metrics were considered to evaluate control performance. The first metric is the mean absolute error (MAE) between the desired force and the actual force, which quantifies the average force tracking error $\mathcal{E}_f = \frac{1}{T-t_0}\int_{t_0}^{T} \|f^d - f\|\, dt$; the second metric accounts for the average tracking error of the sliding velocity $\mathcal{E}_v = \frac{1}{T-t_0}\int_{t_0}^{T} \|v_s - v_b\|\, dt$; while the third metric concerns the average power consumption, computed as $\mathcal{P} = \frac{1}{T-t_0}\int_{t_0}^{T} \|\tau \cdot \dot{q}_j\|\, dt$. Metrics were computed over a fixed 10 s window, from $t_0 = 10$ s to $T = 20$ s, to isolate steady-state performance. The first 10 s, during which the robot approaches and establishes contact with the wall, were excluded from the analysis. A four-way ANOVA with a significance level of $\alpha = 0.05$ was performed. Assumptions of independence, normality (Kolmogorov–Smirnov test), and homogeneity of variances (Levene's test) were satisfied. Fig. 5 shows the mean and standard deviation of the three metrics across all factor levels for the ANYmal-D robot, with p-values reported for statistically significant pairwise comparisons. The plots for the other three robots are provided in the attached video.

For $\mathcal{E}_f$, values remain low for trot and crawl, worsen for bound, and stay good for pace on ANYmal but not Unitree robots. The metric is stable across stick orientations and increases with higher desired forces. Regarding sliding velocity, performance degrades at 0.2 m/s, while differences between 0.05 m/s and 0.1 m/s are not always statistically significant, although mean errors at 0.05 m/s are never lower than 0.1 m/s. For $\mathcal{E}_v$, the crawl exhibits a significantly lower

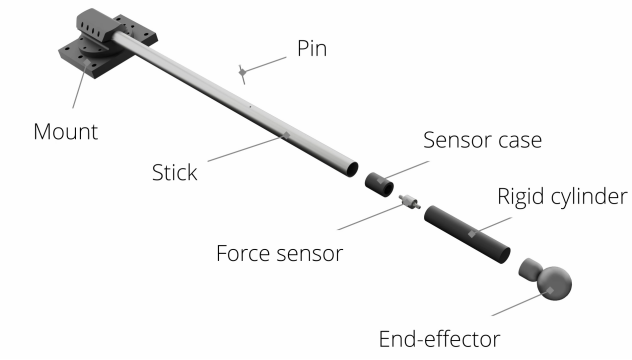

Fig. 6. The tool used to perform the push-and-slide experiments.

standard deviation, while the trot performs very well in some cases but poorly in others. Regarding the pace and bound, they tend to fail on $\mathcal{E}_v$ whenever they perform well on $\mathcal{E}_f$, and vice versa, indicating that these gaits are unsuitable for the push-and-slide task. No significant difference is found for stick orientation, while higher interaction forces and sliding speeds degrade performance. For $\mathcal{P}$, results mainly depend on gait, with trot the most efficient and bound the least. The frontal configuration is the most efficient for all robots, and power consumption slightly increases with force and speed. In conclusion, despite minor differences among robots, overall rankings remain stable and results fully consistent.

2) Learned Gait Timing Analysis: For all trainings, the learned full gait cycles are relatively short, ranging from 0.18 s to 0.28 s. Having a sufficiently fast gait cycle is indeed the most effective way to accomplish the task, as will be further discussed in Sec. IV-C2. All learned trot gaits consistently spend more time in two-leg support (q_0, q_2) than in four-leg support (q_1, q_3), reflecting their natural balance on diagonal limbs. The crawl gaits are evenly divided across all four phases, as only one foot is lifted at a time. In contrast, the bound gaits spend more time in four-leg support than in two-leg support, struggling to keep balance on two legs. Finally, the timing of the pace gait is similar to that of the crawl gait in the frontal and diagonal configurations, while in the lateral configuration, it more closely matches the bound gait, which is inherently more challenging because of its lateral oscillations. If a fixed frequency had been assigned for all gaits, the results would not have allowed for a valid and balanced comparison.

B. Experimental Setup, Sim-to-Real Transfer, and Results

The policies described in Sec. III were deployed on the available ANYmal-D robot. To facilitate sim-to-real transfer, static and dynamic friction of the robot, ground, and wall were randomized within the range $[0.5, 1.25]$. Additionally, slight modifications were made to the robot's base mass from the interval $[-5, 5]$ kg, and external forces were applied to the base in the range $[-10, 10]$ N. Observation noise is also added [28]. The real robot was equipped with a simple yet effective custom-designed tool for push-and-slide testing, shown in Fig. 6. The tool consists of an aluminium tube rigidly attached to the robot's base via a 3D-printed mount. A DYMH-106 force sensor was embedded inside the tube to provide force feedback during pushing. The control algorithm runs at 50 Hz

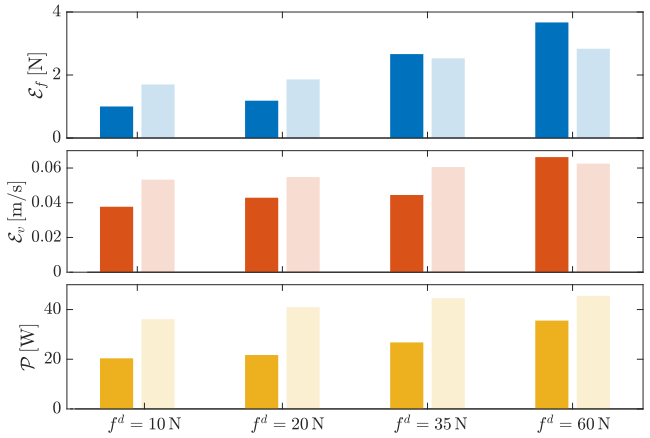

Fig. 7. Comparison of performance metrics between trot and crawl gaits in experiments conducted at $v_s = 0.1$ m/s, with desired forces $f^d = 10$ N, 20 N, 35 N, and 60 N. Top: force tracking error, $\mathcal{E}_f$; Middle: velocity tracking error, $\mathcal{E}_v$; Bottom: average power consumption, $\mathcal{P}$. Trot results are shown in dark blue/red/yellow, and crawl results in light blue/red/yellow.

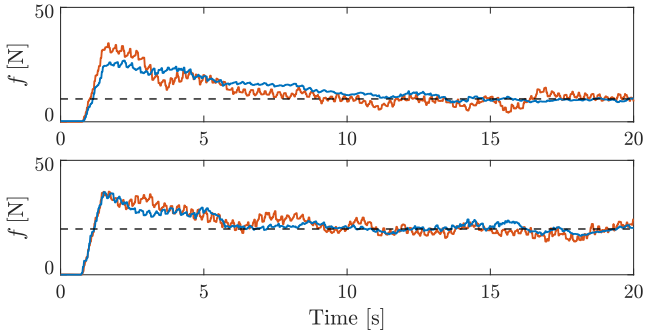

Fig. 8. Interaction force magnitude between the tool and the wall in the experiment. Blue is the trot gait, red is the crawl gait. Top: comparison with a force reference of 10 N. Bottom: comparison with a 20 N reference. Since the desired force is low, the trot exhibits better performance than the crawl, as the high-frequency oscillations induced by locomotion are more pronounced in the crawl gait than in the trot.

directly on the locomotion PC, which is equipped with an Intel i7 Core processor. The force sensor is appropriately integrated into the control loop, interfacing with an Arduino UNO R4 WiFi at 80 Hz. The same PI gains as in simulation were used. A standard concrete wall was considered for pushing (Fig. 1).

Experiments were carried out to validate the simulation results and statistical analysis. The most promising configurations from Sec. IV-A (*i.e.*, frontal stick, trot and crawl gaits) were considered.

1) Trot and Crawl Comparison: Fig. 7 shows the experimental performance of the trot and crawl gaits with a frontal stick across all three metrics, considering $v_s = 0.1$ m/s and several values of f^d. Regarding $\mathcal{E}_f$ and $\mathcal{E}_v$, it is particularly interesting to note that the trot performs better at low forces, while the crawl becomes more effective at higher forces. The trot proves capable of more delicate interactions at low contact forces, whereas the crawl benefits from its inherently stable three-leg support when larger forces are required. This trend is consistent with the statistical analyses, where the crawl exhibited lower standard deviations than the trot for both $\mathcal{E}_f$ and $\mathcal{E}_v$. Concerning $\mathcal{P}$, the crawl consistently consumed more power than the trot, although this difference gradually decreases as f^d increases. Figure 8 shows examples of the time evolution of the measured interaction force.

2) Sliding Velocity Comparison: Real-world experiments were also conducted at different sliding velocities. While degraded performance is expected at excessively high speeds (*e.g.*, $v_s = 0.2$ m/s), the relationship between $v_s = 0.05$ m/s and $v_s = 0.1$ m/s is particularly insightful. Considering $f^d = 10$ N to ensure a well-conditioned task and a frontal configuration, the trot achieved $\mathcal{E}_f = 1.07$ N, $\mathcal{E}_v = 0.017$ m/s, and $\mathcal{P} = 15.5$ W at $v_s = 0.05$ m/s, compared to $\mathcal{E}_f = 1.00$ N, $\mathcal{E}_v = 0.038$ m/s, and $\mathcal{P} = 20.4$ W at $v_s = 0.1$ m/s. For the crawl, $\mathcal{E}_f = 1.76$ N, $\mathcal{E}_v = 0.049$ m/s, and $\mathcal{P} = 34.6$ W were obtained at $v_s = 0.05$ m/s, while $\mathcal{E}_f = 1.70$ N, $\mathcal{E}_v = 0.053$ m/s, and $\mathcal{P} = 36.1$ W at $v_s = 0.1$ m/s. Overall, using a lower sliding velocity improved $\mathcal{E}_v$ and $\mathcal{P}$ but not $\mathcal{E}_f$. Slower motion makes it harder to overcome static friction, slightly worsening force-tracking performance compared to a moderate but not excessive velocity such as $v_s = 0.1$ m/s.

C. Ablation Study and Discussion

An ablation study was conducted on key design choices to determine whether, and to what extent, they improve training effectiveness and the robot's task performance. The results reported here were obtained in simulation using the ANYmal-D robot, a trot gait, and a frontal configuration as a representative example. For brevity, only the metrics $\mathcal{E}_f$ and $\mathcal{E}_v$ are considered.

1) Contact Reward Ablation: Performance was compared with and without the contact-preservation reward to enable a fair comparison with a similar policy designed solely for locomotion. For $\mathcal{E}_f$, using the reward yields 3.43 N, 5.42 N, and 8.00 N for $f^d = 10$ N, 35 N, and 60 N, respectively. Without the reward, the values are 5.93 N, 12.03 N, and 18.65 N. It is clear that, without the reward, the robot does not learn to exploit the wall effectively, which instead acts only as an external disturbance.

2) Gait Learning Ablation: Learning the trot gait resulted in a complete cycle of 0.22 s, with 63.63 % of the time in two-leg support (q_0, q_2) and 36.36 % in four-leg support (q_1, q_3). The following analysis compares performance across different contact timing strategies. Specifically, the learned timing is evaluated against cases where all four phases of the trot cycle are fixed to 0.04 s, 0.06 s, 0.08 s, 0.10 s, 0.12 s, and 0.14 s. To enforce a fixed frequency, the method from [12] is used, amplifying the reward during phase transitions rather than during state preservation. Fig. 9 reports the obtained results. For both metrics, performance is superior when timing is learned autonomously. For $\mathcal{E}_f$, excessively fast gaits (*e.g.*, 0.04 s) generate excessive oscillations, leading to poor performance. Instead, for $\mathcal{E}_v$, overly slow cycles (*e.g.*, 0.14 s) prevent accurate velocity tracking. Finally, when no gait is imposed, all robots naturally learn the trot, already included in the statistical analysis.

3) Curriculum Learning Ablation: Two alternative structures are now tested: (i) maintaining the split throughout training (removing the second phase of the training), and (ii) training all robots to interact from the start (removing the first phase). For $\mathcal{E}_f$, the results were 3.43 N (curriculum),

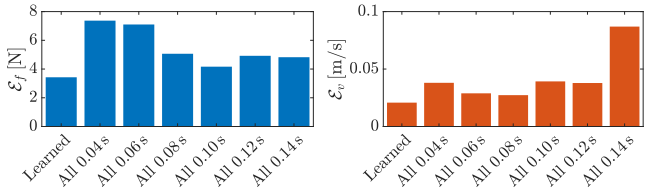

Fig. 9. Force tracking error $\mathcal{E}_f$ (left) and velocity tracking error $\mathcal{E}_v$ (right) for the ANYmal-D robot using a trot gait, frontal stick, and $f^d = 10$ N in simulation, comparing learned gait timing with fixed gait timing.

10.51 N (first alternative), and 5.44 N (second alternative). For $\mathcal{E}_v$, the values were 0.0208 m/s, 0.0385 m/s, and 0.0245 m/s, respectively. The curriculum-based approach achieved the best performance in both metrics. The first alternative performed worst, highlighting the importance of the specialization phase for stable and controlled wall interaction, while the second suffered from limited generalization, leading to poorer disturbance rejection and less coordinated motion.

V. Conclusion

The objective of this work was to identify the most advantageous configurations for a push-and-slide task performed by a quadruped robot. A four-way ANOVA revealed that bound and pace gaits are unsuitable for this task, while the trot gait exhibited the best performance at low force levels, and the crawl gait provided high consistency and stability. Since NDT inspections typically do not require excessive contact forces, the trot gait, combined with a frontal base orientation (for efficiency) and a moderate sliding velocity (to better manage stick–slip behavior), proved to be the most effective configuration. The hybrid control strategy, integrating classical control and RL, enabled consistent comparisons, and the proposed gait timing learning method was successful. Future work could address training new policies on uneven terrain and explore a fully RL-based approach to directly control force and reduce oscillations.

References

[1] S. Halder, K. Afsari, E. Chiou, R. Patrick, and K. A. Hamed, "Construction inspection & monitoring with quadruped robots in future human-robot teaming: A preliminary study," *Journal of Building Engineering*, vol. 65, p. 105814, 2023.

[2] X. Hu, F. He, P. Xiao, T. Wang, D. Zhang, X. Zhou, and Y. Fan, "Design of a quadruped inspection robot used in substation," in *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference*, vol. 4, 2021, pp. 766–769.

[3] X. Zhao, Y. Wu, Y. You, A. Laurenzi, and N. Tsagarakis, "Variable stiffness locomotion with guaranteed stability for quadruped robots traversing uneven terrains," *Frontiers in Robotics and AI*, vol. 9, p. 874290, 2022.

[4] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. arxiv 2019," *arXiv preprint arXiv:1909.06586*.

[5] N. Rudin, J. He, J. Aurand, and M. Hutter, "Parkour in the wild: Learning a general and extensible agile locomotion policy using multi-expert distillation and rl fine-tuning," *arXiv preprint arXiv:2505.11164*, 2025.

[6] S. K. Dwivedi, M. Vishwakarma, and A. Soni, "Advances and researches on non destructive testing: A review," *Materials Today: Proceedings*, vol. 5, no. 2, pp. 3690–3698, 2018.

[7] F. Ruggiero, V. Lippiello, and A. Ollero, "Aerial manipulation: A literature review," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1957–1964, 2018.

[8] S. Marcellini, S. D'Angelo, A. De Crescenzo, M. Marolla, V. Lippiello, and B. Siciliano, "Development of a semi-autonomous framework for ndt inspection with a tilting aerial platform," in *International Symposium on Experimental Robotics*, 2023, pp. 353–363.

[9] S. D'Angelo, M. Selvaggio, V. Lippiello, and F. Ruggiero, "Semi-autonomous unmanned aerial manipulator teleoperation for push-and-slide inspection using parallel force/vision control," *Robotics and Autonomous Systems*, vol. 186, p. 104912, 2025.

[10] J. Wang, I. Chatzinikolaidis, C. Mastalli, W. Wolfslag, G. Xin, S. Tonneau, and S. Vijayakumar, "Automatic gait pattern selection for legged robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 3990–3997.

[11] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on Robot Learning*, 2022, pp. 773–783.

[12] D. DeFazio, Y. Hayamizu, and S. Zhang, "Learning quadruped locomotion policies using logical rules," in *International Conference on Automated Planning and Scheduling*, vol. 34, 2024, pp. 142–150.

[13] T. Portela, G. B. Margolis, Y. Ji, and P. Agrawal, "Learning force control for legged manipulation," in *2024 IEEE International Conference on Robotics and Automation*, 2024, pp. 15 366–15 372.

[14] M. Selvaggio, A. Garg, F. Ruggiero, G. Oriolo, and B. Siciliano, "Non-prehensile object transportation via model predictive non-sliding manipulation control," *IEEE Transactions on Control Systems Technology*, vol. 31, no. 5, pp. 2231–2244, 2023.

[15] H. Ferrolho, V. Ivan, W. Merkt, I. Havoutis, and S. Vijayakumar, "Roloma: Robust loco-manipulation for quadruped robots with arms," *Autonomous Robots*, vol. 47, no. 8, pp. 1463–1481, 2023.

[16] Y. Ma, F. Farshidian, T. Miki, J. Lee, and M. Hutter, "Combining learning-based locomotion policy with model-based manipulation for legged mobile manipulators," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2377–2384, 2022.

[17] J. Cheng, D. Kang, G. Fadini, G. Shi, and S. Coros, "Rambo: Rl-augmented model-based optimal control for whole-body loco-manipulation," *arXiv preprint arXiv:2504.06662*, 2025.

[18] C. Beltran, D. Petit, I. Ramirez-Alpizar, T. Matsubara, and K. Harada, "Hybrid position-force control with reinforcement learning," in *20th Annual Conference of the SICE System Integration Division*. SICE, 2019, pp. 3001–3006.

[19] X. Cheng, A. Kumar, and D. Pathak, "Legs as manipulator: Pushing quadrupedal agility beyond locomotion," in *2023 IEEE International Conference on Robotics and Automation*, 2023, pp. 5106–5112.

[20] 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, no. 1, pp. 699–706, 2023.

[21] W. J. Wolfslag, C. McGreavy, G. Xin, C. Tiseo, S. Vijayakumar, and Z. Li, "Optimisation of body-ground contact for augmenting the whole-body loco-manipulation of quadruped robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 3694–3701.

[22] D. Vogel, R. Baines, J. Church, J. Lotzer, K. Werner, and M. Hutter, "Robust ladder climbing with a quadrupedal robot," *arXiv preprint arXiv:2409.17731*, 2024.

[23] G. Bellegarda and A. Ijspeert, "Cpg-rl: Learning central pattern generators for quadruped locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 547–12 554, 2022.

[24] P. Zhi, P. Li, J. Yin, B. Jia, and S. Huang, "Learning unified force and position control for legged loco-manipulation," *arXiv preprint arXiv:2505.20829*, 2025.

[25] S. Liu, H. Chai, R. Song, Y. Li, Y. Li, Q. Zhang, P. Fu, J. Liu, and Z. Yang, "Contact force/motion hybrid control for a hydraulic legged mobile manipulator via a force-controlled floating base," *IEEE/ASME Transactions on Mechatronics*, vol. 29, no. 3, pp. 2316–2326, 2023.

[26] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[28] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*, 2022, pp. 91–100.