2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Exploring Constrained Reinforcement Learning Algorithms for Quadrupedal Locomotion

Joonho Lee[1,2,*], Lukas Schroth[1,*], Victor Klemm[1], Marko Bjelonic[1], Alexander Reske[1], and Marco Hutter[1]

***Abstract*— Shifting from traditional control strategies to Deep Reinforcement Learning (RL) for legged robots poses inherent challenges, especially when addressing real-world physical constraints during training. While high-fidelity simulations provide significant benefits, they often bypass these essential physical limitations. In this paper, we experiment with the Constrained Markov Decision Process (CMDP) framework instead of the conventional unconstrained RL for robotic applications. We evaluated five constrained policy optimization algorithms for quadrupedal locomotion using three different robot models. Our aim is to evaluate their applicability in real-world scenarios. Our robot experiments demonstrate the critical role of incorporating physical constraints, yielding successful sim-to-real transfers, and reducing operational errors on physical systems. The CMDP formulation streamlines the training process by separately handling constraints from rewards. Our findings underscore the potential of constrained RL for the effective development and deployment of learned controllers in robotics.**

I. INTRODUCTION

The use of Deep Reinforcement Learning (RL) for robotic control is on the rise, revolutionizing the way control policies are created for legged robots and other complex dynamic systems. Particularly, model-free approaches have gained prominence, replacing traditional optimization-based methods. This paradigm shift can be attributed to the high-capacity neural network models, effective model-free algorithms that can solve complex problems, and efficient tools for data-generation (i.e. simulations). As a result, the synthesis of locomotion policies for legged robots has become more straightforward and accessible, as evidenced by the growing number of RL-based controllers in recent literature.

The so-called sim-to-real approach is commonly employed, where policy training solely relies on simulated data. This is due to the inherent requirements of widely-used algorithms such as Proximal Policy Optimization (PPO) [1] and Soft Actor Critic (SAC) [2], which demand random exploration and a significant number of samples. As a result, training policies directly on hardware is both impractical and hazardous. In recent years, diverse approaches have emerged to enhance simulation fidelity (e.g., actuator modeling [3], hybrid simulator [4], [5]), and to robustify policies against domain shifts (e.g., dynamics randomization [6], [7], privileged training [8]).

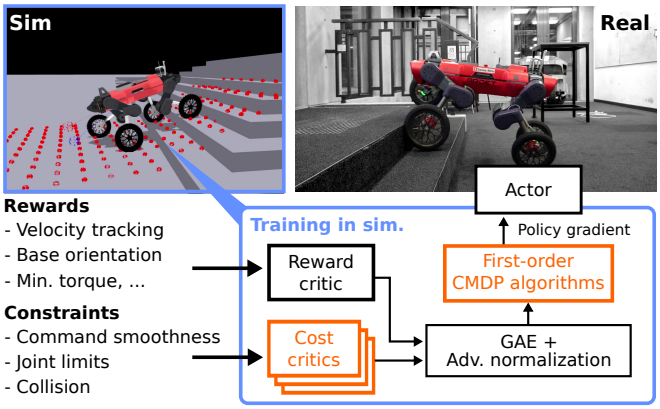

Fig. 1: Training a wheeled-legged quadruped using constrained policy optimization. Additional components to conventional PPO are highlighted.

Notably, while most existing research emphasizes enhancing simulation accuracy and regularizing policies for sim-to-real transfer, a gap persists in the literature — a lack of attention to physical constraints. Despite the studies done in understanding and simulating the physical properties of hardware, the incorporation of essential physical constraints during training remains under-explored.

These constraints can be physical, such as limits on joint velocities, torque limits, or safety regulations. Considering such constraints is a common practice in model-based approaches [9], [10]. Existing literature provides compelling evidence of its significance. For instance, Gangapurwala et al. [11] first utilized a Constrained Proximal Policy Optimization (CPPO) algorithm to train a locomotion controller for a quadrupedal robot, achieving both constraint-consistency and high performance. Kim et al. [12] also experimented with a modified version of Interior-point Policy Optimization (IPO) [13] algorithm and showed rough-terrain locomotion with a generalizable Constrained Markov Decision Process (CMDP) formulation.

In this paper, we evaluate various first-order constrained policy optimization methods, focused on the application to legged locomotion. We formulate velocity-tracking locomotion as a CMDP [14], effectively isolating the physical constraints from the reward function. Additionally, we intro-

This work was supported by the Mobility Initiative grant funded through the ETH Zurich Foundation, European Union's Horizon 2020 research and innovation programme under grant agreement No 101070405, No 852044 and No 101070596, Swiss National Science Foundation through the National Centre of Competence in Digital Fabrication (NCCR dfab) and Apple Inc.

* *Joonho Lee and Lukas Schroth contributed equally.*
[1] The authors are with Robotic Systems Lab; ETH Zurich, Switzerland
[2] The author is now with Neuromeka, Korea
Corresponding author: Joonho Lee (jolee@ethz.ch)

duce a modification to existing algorithms to enhance both stability and final performance.

Our main results can be summarized as follows:

1) We evaluate five first-order constrained RL algorithms on quadrupedal locomotion task in simulation, using two robot models of different sizes and capabilities. We identify suitable algorithms for practical applications based on constraint violations and final performance.
2) We demonstrate the effectiveness of the constrained RL approach in handling physical constraints with the wheeled-legged robot shown in Fig. 1.

From our experiments, we found out that the constrained RL formulation yields fewer constraint violations compared to the commonly used unconstrained approach. Additionally, this reduces the reward-shaping effort for physical limitations, a common practice in the existing research.

II. BACKGROUND

A. Constrained Policy Optimization

In RL, a control problem is typically modeled as a Markov Decision Process (MDP), which is described by a tuple (S, A, r, p, μ). Here, S is the set of states, A is the set of actions, $r : S \times A \times S \to \mathbb{R}$ is the reward function, $p : S \times A \times S \to [0, 1]$ is the state transition probability and μ is the initial state distribution. To solve an MDP, we aim to find a policy $\pi : S \mapsto \mathcal{P}(A)$ that maximizes

$$J_R(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t, s_{t+1})\right], \tag{1}$$

where $\gamma \in [0, 1)$ is the discount factor. Here, the expectation $\mathbb{E}[\ldots]$ represents the empirical average over a finite batch of samples. s_0 is sampled from an initial state distribution μ and trajectories are sampled using π.

To address constrained problems, this framework is extended into a CMDP. The MDP is augmented with a set C of cost functions that capture constraint violations $\{c_1, c_2, \ldots, c_n\}$ and corresponding limits $E = \{\epsilon_1, \epsilon_2, \ldots, \epsilon_n\}$ [14], [15]. Each $c_i : S \times A \times S \to \mathbb{R}$ maps state-action-state triplets to the cost of the state transition. In the constrained setting, an optimal policy maximizes the expected discounted return in Eq. 1, while keeping the discounted sum of future costs c_i below their respective threshold ϵ_i, yielding the constrained optimization problem:

$$\begin{aligned} \max_{\pi} \quad & J_R(\pi) \\ \text{s.t.} \quad & \forall i \in \{1, \ldots, n\},\ J_{C_i}(\pi) \leq \epsilon_i, \end{aligned} \tag{2}$$

where

$$J_{C_i}(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t c_i(s_t, a_t, s_{t+1})\right]. \tag{3}$$

While many constrained RL problems in the literature consider a single constraint (e.g [16], [15], [17]), the CMDP framework is not limited to the single constraint setup [12].

Derived by the performance difference lemma by Shen et al. [16], the constrained optimization problem in Eq. 2 can be reformulated as follows:

$$\max_{\pi'} \quad \mathbb{E}\left[A^{\pi}_{R,t}(s, a)\right] \tag{4a}$$

$$\text{s.t.} \quad \underbrace{J_{C_i}(\pi) + \frac{1}{1-\gamma}\mathbb{E}_{\pi'}\left[A^{\pi}_{C_i,t}(s, a)\right]}_{J_{C_i}(\pi')} \leq \epsilon_i \quad \forall i. \tag{4b}$$

where $A^{\pi}_{R,t}(s, a)$ and $A^{\pi}_{C_i,t}(s, a)$ are estimators of the reward advantage function and cost advantage function for the i-th constraint at timestep t, respectively.

B. First-order Optimization Methods for CMDPs

We compare five first-order policy optimization algorithms in order to identify a method that is performant and stable. As higher-order algorithms typically require resource-intensive computation of the inverse Hessian or inverse Hessian-vector products (see, e.g., CPO [15], PCPO [18], TRPO-Lagrangian [17]), we restrict our scope to first-order algorithms. We considered practical aspects such as the number of hyperparameters, availability of an implementation and the presented empirical results.

1) P3O: Shen et al. [16] proposed to augment the PPO objective with penalties on the constraint violations. The objective function for Penalized Proximal Policy Optimization (P3O) is defined as:

$$L^{\text{CLIP}}_R(\theta') - \sum_i \kappa_i \cdot \max\left\{0, J_{C_i}(\pi') - \epsilon_i\right\}, \tag{5}$$

κ_i controls the weight of each constraint.

The first term $L^{\text{CLIP}}_R(\theta')$ is the clipped surrogate objective by Schulman et al. [1], defined as:

$$L^{\text{CLIP}}_R(\theta') = \mathbb{E}\left[\min(r_t(\theta')\tilde{A}^{\pi_\theta}_{R,t}, \text{clip}(r_t(\theta'))\tilde{A}^{\pi_\theta}_{R,t}\right], \tag{6}$$

where $r_t(\theta')$ denote the probability ratio $\frac{\pi'(a_t|s_t)}{\pi(a_t|s_t)}$, and the operation $\text{clip}(\cdot)$ clips the value between $1 - \delta$ and $1 + \delta$ with δ controlling the magnitude of policy updates. $\tilde{A}_{R,t}$ denotes the normalized reward advantage.

Similarly, the final objective of P3O is obtained using importance sampling and clipping of the importance ratios of the cost advantages:

$$L^{P3O}(\theta') = L^{\text{CLIP}}_R(\theta') - \sum_i \kappa_i \cdot \max\left\{0, L^{\text{VIOL}}_{C_i}(\theta')\right\}, \tag{7}$$

with

$$L^{\text{VIOL}}_{C_i}(\theta') = L^{\text{CLIP}}_{C_i}(\theta') + (1 - \gamma)(J_{C_i}(\pi_\theta) - \epsilon_i)$$

$$L^{\text{CLIP}}_{C_i}(\theta') = \mathbb{E}\left[\max(r_t(\theta')A^{\pi_\theta}_{C_i,t}, \text{clip}(r_t(\theta'))A^{\pi_\theta}_{C_i,t})\right].$$

2) PPO-Lagrangian: Chow et al. [19] proposed to utilize the Lagrangian relaxation. The Lagrangian method approaches constraint problems with objective $f(\theta)$ and constraint $g(\theta)$ by minimizing the Lagrange dual with dual variable λ, resulting in the unconstrained objective:

$$\min_{\lambda \geq 0} \max_{\theta} \mathcal{L}(\theta, \lambda) \doteq f(\theta) - \lambda g(\theta). \tag{9}$$

Approximate solutions of this minimax objective can be obtained via the iterative primal-dual method, which alternates between updates on the primal variable θ and the dual variable λ [20].

OpenAI researchers [17] suggested utilizing the iterative primal-dual method with the PPO objective to derive the following update:

$$\theta' = \theta + \alpha_\theta \nabla_\theta \left(L_R^{\text{CLIP}}(\theta) - \sum_i \lambda_i L_{C_i}^{\text{CLIP}}(\theta) \right), \tag{10}$$

$$\lambda_i' = \lambda_i + \alpha_{\lambda_i}(J_{C_i}(\theta) - \epsilon_i). \tag{11}$$

Here, α_θ and α_{λ_i} are the learning rates of the gradient ascent and descent steps, respectively. λ_i' is typically cut off at zero, to ensure non-negativity of the penalty parameter.

3) IPO: Inspired by the interior-point method for constrained optimization problems, IPO uses logarithm barrier functions $\phi(x) = \log(-x)/k$, with the hyperparameter k to achieve an infinitely large penalty as the estimated cost returns approach the constraint threshold ϵ_i. This results in the objective:

$$L^{IPO}(\theta') = L_R^{\text{CLIP}}(\theta') + \sum_{i: J_{C_i}(\theta') < \epsilon_i} \phi(J_{C_i}(\theta') - \epsilon_i), \tag{12}$$

where $J_{C_i}(\theta')$ can be estimated based on the advantages using Eq. 4b.

4) CRPO: Constraint-Rectified Policy Optimization (CRPO) [21] alternates between maximizing the objective and minimizing the constraint violations whenever the constraints are violated:

$$L^{CRPO}(\theta') = \mathbb{1}_{J_C(\theta) \leq \epsilon_i} \cdot L_R^{\text{CLIP}}(\theta') - \mathbb{1}_{J_C(\theta) > \epsilon_i} \cdot L_C^{\text{CLIP}}(\theta'). \tag{13}$$

$\mathbb{1}(\cdot)$ denotes the indicator function.

5) FOCOPS: First-Order Constrained Optimization in Policy Space (FOCOPS) solves the constrained optimization problem in policy space and then projects the solution back into parameter space, effectively also leading to an objective function with a constraint penalty [22]. For a detailed derivation we refer to the original paper of Zhang at al. [22].

The algorithms P3O [16], PPO-Lagrangian [17], and IPO [13] relax the constrained optimization problem in Eq.2 into an unconstrained one using additional penalties to the PPO objective. CRPO takes a simpler approach and alternates between PPO updates with reward and cost advantages [21]. FOCOPS [22] solves the constrained optimization problem in policy space.

III. METHOD

We define a CMDP to train policies for velocity-tracking perceptive locomotion. The training environment and MDP inherit from the quadruped environment by Rudin et al. [23].

Task Rewards	
Linear Velocity	$\exp(-2.0 \cdot \|\|v_{xy}^{targ} - v_{xy}\|\|^2)$
Yaw Rate	$\exp(-2.0 \cdot \|\|\omega_z^{targ} - \omega_z\|\|^2)$
Style Rewards	
Base Stability	$\exp(-v_z^2) + \exp(-\|\|\omega_{x,y}\|\|^2)$
Height	$-0.5 \ \|h^{targ} - h^{robot}\|$, $h^{targ} = 0.5$
Joint Torque Minimization	$1\text{e}{-6} \ \|\|\tau\|\|^2$
Joint Motion	$1\text{e}{-5} \ \|\|\dot{q}\|\|^2 + 1\text{e}{-6} \ \|\|\ddot{q}\|\|^2$
Constraint Rewards (Removed for CMDP)	
Command Smoothness 1	$-0.01 \ \|\|q_t^{des} - q_{t-1}^{des}\|\|^2$
Command Smoothness 2	$-0.01 \ \|\|q_t^{des} - 2q_{t-1}^{des} + q_{t-2}^{des}\|\|^2$
Joint Torque Limits	$-0.01 \ \sum \max(\|\tau_{i,t}\| - \tau_i^{lim}, 0)^2$
Joint Speed Limits	$-0.1 \ \sum \max(\|\dot{q}_{i,t}\| - \dot{q}_i^{lim}, 0)^2$
Joint Position Upper Limits	$-10.0 \ \sum \max(q_{i,t} - q_i^{ub}, 0)^2$
Joint Position Lower Limits	$-10.0 \ \sum \max(q_i^{lb} - q_{i,t}, 0)^2$
Body Contact	$-$(Number of non-wheel contacts)

TABLE I: Reward Functions. q and τ are joint position and torque vectors.

A. CMDP for Perceptive Locomotion

1) Reward Functions: Our reward function is a sum of different reward terms provided in Table I. We define three categories:

- Task Reward: This defines the main task objective. In our experiment, the main task is to track linear velocity command in horizontal direction (v_{xy}) and yaw rate (ω_z).
- Style Reward: There can be many solutions for the velocity tracking, e.g., different gait, base height, or different orientation. We use extra rewards to guide natural-looking motions. Kim et al. [12] similarly achieved this by applying constraints to gait and other physical quantities.
- Constraint Reward: High penalty is given when the physical limits are violated. The constraint rewards are replaced by the constraints in CMDP.

2) Constraints: For all constraints, we set $\epsilon_i = 0$ and defined cost functions such that each cost encapsulates a specific physical quantity:

- Command Smoothness: For the sim-to-real transfer, it is crucial to consider the tracking bandwidth of the physical actuators [9]. Existing works regularize the output with negative rewards on the first or second order derivative of the commands [8], [23], [12]. This prevents infeasible commands, reduces sim-to-real discrepancy in the joint space, and vibration on the hardware.

  We define two constraint functions as:

  $$c_{c1,i} = \max(0, |(q_{t,i}^{des} - q_{t-1,i}^{des})/dt| - \dot{q}^{des,*})$$
  $$c_{c2,i} = \max(0, |(q_{t,i}^{des} - 2q_{t-1,i}^{des} + q_{t-2,i}^{des})/dt^2| - \ddot{q}^{des,*})$$

  for each joints ($i \in$ joints). dt is the timestep and $\dot{q}^{des,*}$ and $\ddot{q}^{des,*}$ are thresholds.

  The discounted sum of both costs are restricted to be below the desired thresholds by setting $\epsilon = 0.0$. $\dot{q}^{des,*}$ and $\ddot{q}^{des,*}$ are hyperparameters, with $\dot{q}^{des,*}$ set as half of the joint speed limit, and $\ddot{q}^{des,*} = \dot{q}^{des,*}/dt$.
- Joint Speed: The constraint function is defined as an

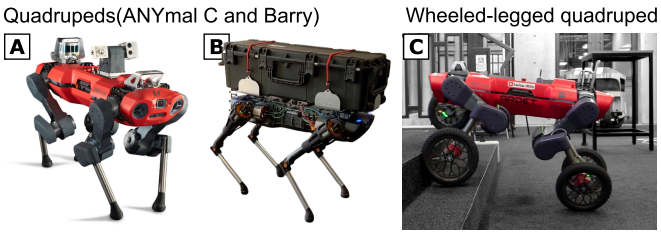

Fig. 2: Robot models used for experiments. (A) ANYmal C (B) Barry [24] (C) Wheeled-legged quadruped [25]

indicator function:

$$c_{qv} = \mathbb{1}(\sum_{i \in \text{joints}} \mathbb{1}(|\dot{q}_{t,i}| > \dot{q}_i^*) > 0.0).$$

In other words, $c_{qv} = 1$ if any of the joints violates the speed limitation. $\dot{q}^*$ is the physical limit of the actuator.

- Joint Torque: Joint torque constraint is defined similarly to the joint speed constraint

$$c_\tau = \mathbb{1}(\sum_{i \in \text{joints}} \mathbb{1}(|\tau_{t,i}| > \tau_i^*) > 0.0).$$

- Joint Position: Each joint has different upper bound (q^{ub}) and lower bound (q^{lb}) positions. We only set the limit angle for the hip joints to avoid self-collision.

$$c_q = \mathbb{1}(\sum_{i \in \text{hip joints}} (\mathbb{1}(q_{t,i} > q_i^{ub}) + \mathbb{1}(q_{t,i} < q_i^{lb})) > 0.0).$$

- Undesirable Body Contact: The cost is 1.0 when there is any contact at the body parts except for the wheel or foot, including self-collision.

B. Normalizing Cost Advantages

Advantage normalization is a widely used heuristic to improve the stability of policy gradient algorithms [26]. This technique is also applicable for constrained RL algorithms.

Consider the simplified objective for P3O:

$$L(\theta') = \mathbb{E}\left[r(\theta')\left(A_R^\theta - \kappa \cdot A_C^\theta\right)\right]$$

The un-normalized advantages A_R^θ and A_C^θ can have different magnitudes, depending on the reward, constraints, and the current policy's behavior. With normalized advantages,

$$L(\theta') = \mathbb{E}\left[r(\theta')\left(\tilde{A}_R^\theta - \kappa \cdot \tilde{A}_C^\theta\right)\right] \tag{14}$$

then the weighting of the constraints (κ) remains unchanged regardless of the reward and cost functions. E.g., $\kappa = 1$ always corresponds to equal weighting of the reward and cost advantages. This makes the algorithm more stable and improves generalization across tasks, also as evidenced by Kim et al. [12]. Furthermore, this prevents the cost advantages from vanishing when cost violation is low.

For P3O and IPO, we need to reformulate the objectives in Eq. 7 and Eq. 12. We start by expressing the constraint in Eq. 4b in terms of normalized advantages:

$$\frac{(1-\gamma)(J_{C_i}(\pi) - \epsilon_i) + \mu_{C_i}}{\sigma_{C_i}} + \mathbb{E}\Big[\underbrace{\frac{A^\pi_{C_i,t} - \mu_{C_i}}{\sigma_{C_i}}}_{\tilde{A}^\pi_{C_i,t}}\Big] \leq 0 \quad \forall_i. \tag{15}$$

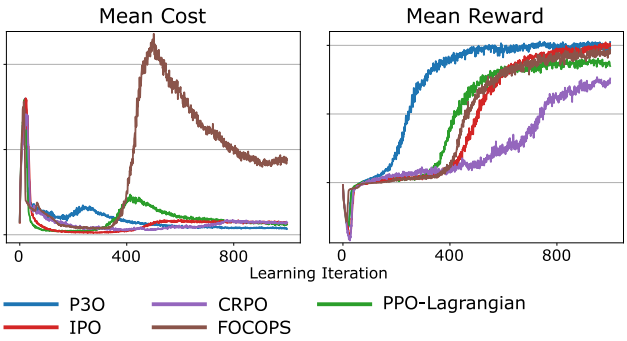

Fig. 3: Learning curves of the selected CMDP algorithms for ANYmal C.

	Reward	Violations per episode
PPO (unconstrained)	24.96 (± 0.67)	533.44 (± 108.94)
P3O	24.13 (± 1.14)	**0.49** (± 0.88)
IPO	**24.67** (± 0.84)	1.33 (± 1.69)
PPO-Lagrangian	23.68 (± 1.87)	0.99 (± 1.31)
CRPO	22.28 (± 1.70)	0.96 (± 1.22)
FOCOPS	22.65 (± 3.02)	15.82 (± 11.67)

(a) ANYmal C

	Reward	Violations per episode
PPO (unconstrained)	30.03 (± 1.91)	88.392 (± 38.01)
P3O	**29.25** (± 3.04)	0.53 (± 2.41)
IPO	28.87 (± 1.49)	**0.31** (± 0.97)
PPO-Lagrangian	28.19 (± 2.86)	1.09 (± 7.60)
CRPO	25.70 (± 3.15)	0.41 (± 1.73)
FOCOPS	25.43 (± 3.89)	21.96 (± 14.14)

(b) Barry

TABLE II: Comparison of the CMDP algorithms for different quadrupedal robots.

Here, μ_{C_i}, σ_{C_i} are the mean and standard deviations of the cost advantages. $\tilde{A}_\pi^{C_i}$ denotes the normalized advantages. Using importance sampling with clipping, one obtains

$$L_{C_i}^{\text{VIOL,N}}(\theta') = L_{C_i}^{\text{CLIP,N}}(\theta') + \frac{(1-\gamma)(J_{C_i}(\pi_\theta) - \epsilon_i) + \mu_{C_i}}{\sigma_{C_i}} \leq 0. \tag{16}$$

The superscript N indicates the usage of normalized advantage estimates. Penalizing violations of Eq. 16, leads to the objectives

$$L^{\text{P3O, N}}(\theta') = L_R^{\text{CLIP,N}}(\theta') - \sum_i \kappa_i \cdot \max\left\{0, L_{C_i}^{\text{VIOL,N}}(\theta')\right\},$$
$$L^{\text{IPO, N}}(\theta') = L_R^{\text{CLIP,N}}(\theta') + \sum_i \phi(L_{C_i}^{\text{VIOL,N}}(\theta')).$$

We will use these normalized versions of P3O and IPO throughout the rest of the paper.

IV. EXPERIMENTAL RESULTS

We present two experiments using three different robots shown in Fig. 2:

1) **Comparison of first-order CMDP algorithms (ANYmal C and Barry)**: We select an algorithm for our purposes (P3O) based on a comparative study of different first-order CMDP algorithms with two

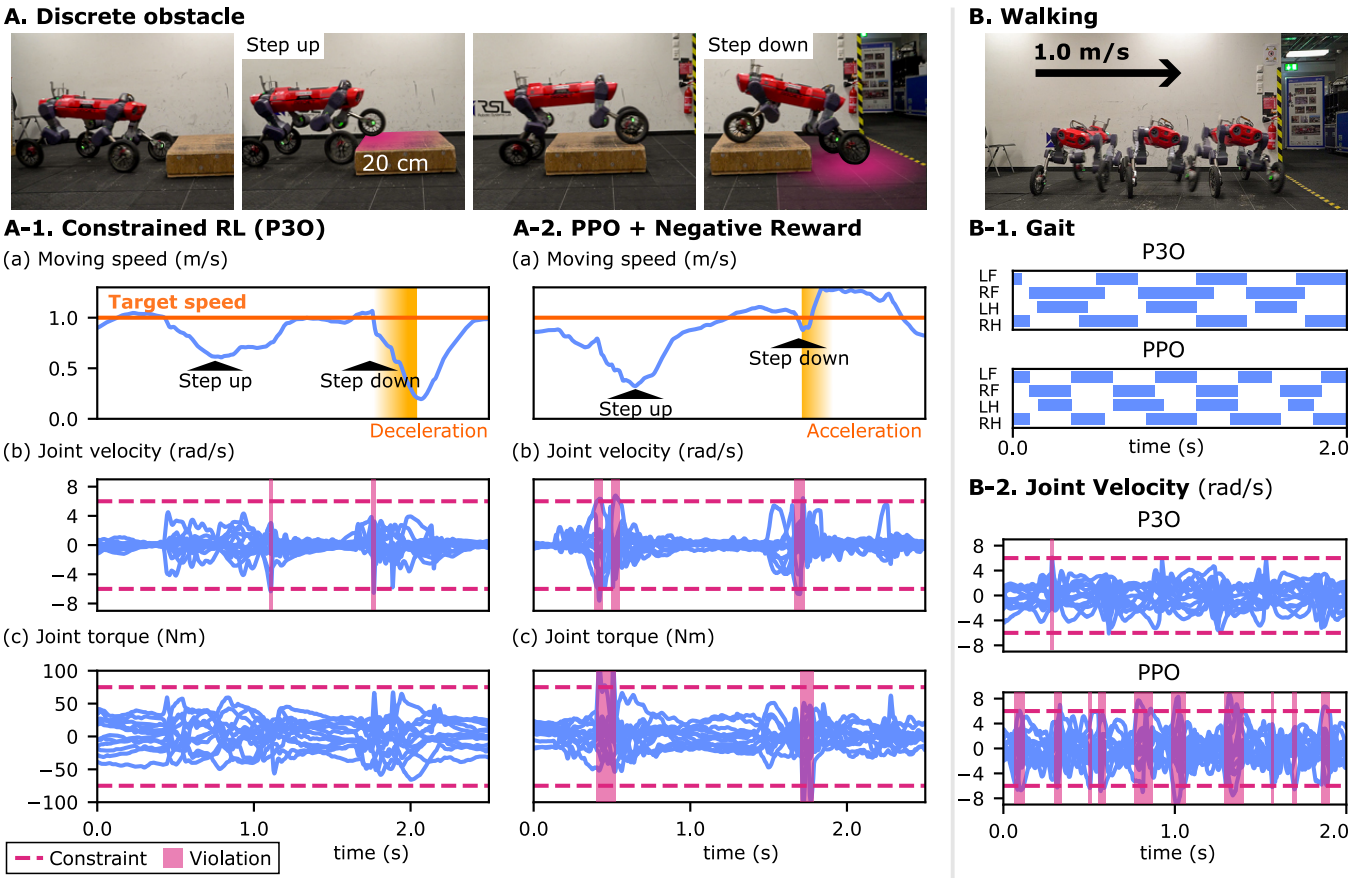

Fig. 4: Robot experiments with constraints. (A) Traversing a 20 cm high block with 1.0 m/s command to the front. (B) Walking in y-direction at the maximum speed.

different quadrupedal robots of different sizes and joint actuators.

2) **Sim-to-real transfer (Wheeled-legged robot)**: We validate the CMDP framework by training a perceptive locomotion policy for the wheeled-legged quadruped while enforcing tight physical constraints. We compare it to a standard PPO-trained policy.

A. Comparing different CMDP Algorithms

1) Experimental Setup: We consider an example problem of legged locomotion on flat terrain with constrained joint velocities. We constrain the joint velocities to be below 6.0 rad/s. The joint speed limit of 6.0 rad/s is significantly lower than the actual physical capabilities of the robots; $\sim 8.0\,\mathrm{rad/s}$ for ANYmal C and $\sim 20\,\mathrm{rad/s}$ for Barry [24]. The same set of reward and constraint functions are used for both robots. The scales of the reward components were tuned separately for each robot.

We implement all algorithms with normalized advantages. As we aim to obtain zero constraint violations, we used P3O, PPO-Lagrangian and FOCOPS with a threshold (ϵ) of zero. Hereby, the cost return cannot drop below zero since the cost function is non-negative. For IPO and CRPO, we treat the threshold as a hyperparameter. Note that CRPO only applies reward improvement steps if the cost returns are below ϵ, and the logarithm barrier penalty term in IPO also needs constraint satisfaction to be well-defined.

For IPO, since the barrier penalty cannot be applied if the constraint is already violated, we've introduced an additional penalty to restore constraint satisfaction: $L^{IPO,new}(\theta') = L^{IPO}(\theta') + \lambda_{\text{rec}} \sum_{i:J_{C_i}(\theta')\geq\epsilon_i} L_{C_i}^{\text{CLIP}}(\theta')$.

We've implemented a warm start for all algorithms, wherein constraints are ignored during the initial 200 iterations. In particular, for P3O, PPO-Lagrangian, and FOCOPS, we set κ, λ, and ν to zero, respectively. Likewise, for IPO and CRPO, we ignored all constraint-related terms in the objective. This initial phase allowed the agents to explore freely without constraints, which was crucial for enhancing their final performance.

2) Results: Fig. 3 and Table II show the cost and reward over the learning iterations and the final performance of the best runs. The mean and the standard deviation are computed from 2500 random episodes of the final policy. We include PPO without considering the constraint as a baseline.

P3O consistently achieved high rewards with less than a single constraint violation on average for both robots. For both robots, P3O and IPO emerged as the top performers in terms of the reward. IPO demonstrated a high violation rate for Anymal C robot compared to other top-3 algorithms. The constraint violation is unavoidable due to the non-negative

ϵ by design, but potential improvements could be explored by using different cost functions and advanced scheduling techniques, as proposed by Kim et al. [12]. PPO-Lagrangian showed similar performance, but the variance of the result was higher than P3O and IPO.

*3) **Our choice**:* For our real-world experiment, we decided to use P3O. Among the compared algorithms, P3O required the fewest parameters to adjust in our setup (with ϵ fixed at zero) and achieved low constraint violation. Although IPO showed similar performance, its sensitivity to the threshold parameter made it less suitable.

B. Robot Experiments

We evaluate a perceptive locomotion policy trained using P3O for our wheeled-legged quadrupedal robot [25]. We compare it with the PPO baseline trained with the constraint reward (see Table I).

1) Experimental Setup: The policies are trained to follow velocity commands over rough terrain. The policy observes the terrain scan around the robot as shown by Fig. 1 and outputs joint position and wheel speed commands. We used the rough terrain environment by Rudin et al. [23]. The velocity commands are sampled uniformly within the ranges of [-2.0, 2.0] m/s in the x-direction, [-1.0, 1.0] m/s in the y-direction, and a yaw rate from [-1.5, 1.5] rad/s.

To evaluate the effectiveness of the constrained RL approach, we enforce tight constraints for the leg actuators. We use joint speed limit of 6.0 rad/s, which is significantly lower than the robot's actual physical limit of $\sim$8.0 rad/s. Joint torque is limited to 75 Nm for leg joints. The physical limit is about 100 Nm.

We applied all constraints mentioned in section III-A. We used two cost critic networks - one for command smoothness constraint and the other one for the sum of other costs.

2) Results: In Fig. 4 we show the results from different policies in two scenarios. Both policies violated joint velocity and torque constraints at varying rates in our experiments, while other constraints remained satisfied.

Firstly, we evaluate the policies' behavior when they encounter discrete obstacles (Fig. 4A). A notable qualitative difference in behavior is observed: the P3O policy slows down before stepping down to reduce impact, while the normal PPO policy gains speed (See Fig. 4A-1(a) and A-2(a)). This significantly impacts the rate of constraint violation.

The P3O policy shows two short peaks in the joint velocity that violate constraints, but the joint torque remains within the constrained range (Fig. 4A(b)). On the other hand, the PPO policy exhibits a significantly higher violation rate when stepping up (the front wheel collision) and when stepping down (front legs drop). The P3O policy actively modulates its leg motions and speed in response to discrete events.

Secondly, we evaluate the constraint violation when the robot is stepping at its maximum speed to the y-direction (Fig. 4B). We commanded 1.0 m/s, which is the maximum speed the policy is trained for. Note that this is higher than the nominal operating range for the ANYmal robot ($\sim$0.75 m/s by [8], [27]).

As shown in Fig. 4B-1, the P3O policy shows longer strides and slower gait frequency, resulting in less joint velocity constraint violation (Fig. 4B-2). Additionally, the P3O policy exhibited lower tracking error. The tracking errors are $0.276\ (\pm 0.077)\,\mathrm{m/s}$ and $0.296\ (\pm 0.091)\,\mathrm{m/s}$ for P3O and PPO, respectively. Both policies could not achieve 1.0 m/s due to the hardware limitation.

V. CONCLUSION & DISCUSSION

Our study presents the effectiveness of CMDP formulation for the perceptive locomotion of quadrupedal robots. Through a comparative study of five first-order CMDP algorithms, we identified a normalized version of P3O as one of the most effective algorithms for our task. The additional advantage normalization step further enhanced both the stability and performance of the algorithm.

Real-world experiments on a wheeled-legged quadrupedal robot provide strong evidence for the effectiveness of the constrained RL approach. Utilizing the P3O algorithm, our policies were able to achieve performance metrics on par with the conventional PPO algorithm used by state-of-the-art approaches, but with fewer constraint violations. A distinct advantage we observed was the decoupling of reward and constraint functions, which simplified the tuning processes and led to a better performance in terms of constraint violation.

This improvement can be attributed to the use of multiple critics in the framework, which has been shown to improve performance in multi-objective problems [28]. While the conventional approach (e.g., PPO) and constrained RL algorithms share similar objectives, separate critics allow for improved credit assignment and far-sighted decision-making regarding constraints.

In conclusion, constrained RL emerges as a promising tool for robotic applications, particularly in sim-to-real transfer scenarios. While our focus was on legged locomotion, the methodology is broadly applicable.

A. Practical Benefits

From a hands-on perspective, the constrained RL algorithms showed clear advantages. The PPO approach necessitated complex adjustments to the scaling coefficients of penalty terms (see Table I). The impact of each coefficient is non-intuitive, often demanding numerous trial and errors. On the other hand, with separate cost critics, this effort is removed by design. We can control the influence of the cost objective using a single parameter κ. Such a streamlined approach accelerates the overall development of learned controllers. While having additional cost critics adds a computational overhead in comparison to PPO (0.07 s more), this is negligible compared to the simulation time ($\sim 0.74\,\mathrm{s}$).

B. Future work

Future works will include different applications such as autonomous navigation or manipulation. Additionally, we only experimented with simple and constant constraints.

More complex systems, such as joints with variable gear ratios [24], may introduce state-dependent constraints. Identifying complex constraints from an unknown or under-modeled systems remains an open question.

REFERENCES

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

[2] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[3] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.

[4] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[5] Y. Jiang, T. Zhang, D. Ho, Y. Bai, C. K. Liu, S. Levine, and J. Tan, "Simgan: Hybrid simulator identification for domain adaptation via adversarial reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2884–2890.

[6] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[7] Z. Xie, X. Da, M. Van de Panne, B. Babich, and A. Garg, "Dynamics randomization revisited: A case study for quadrupedal locomotion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4955–4961.

[8] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.

[9] R. Grandia, F. Farshidian, A. Dosovitskiy, R. Ranftl, and M. Hutter, "Frequency-aware model predictive control," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1517–1524, 2019.

[10] D. Kang, F. De Vincenti, and S. Coros, "Nonlinear model predictive control for quadrupedal locomotion using second-order sensitivity analysis," *arXiv preprint arXiv:2207.10465*, 2022.

[11] S. Gangapurwala, A. Mitchell, and I. Havoutis, "Guided constrained policy optimization for dynamic quadrupedal robot locomotion," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3642–3649, 2020.

[12] Y. Kim, H. Oh, J. Lee, J. Choi, G. Ji, M. Jung, D. Youm, and J. Hwangbo, "Not only rewards but also constraints: Applications on legged robot locomotion," *arXiv preprint arXiv:2308.12517*, 2023.

[13] Y. Liu, J. Ding, and X. Liu, "Ipo: Interior-point policy optimization under constraints," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 4940–4947.

[14] E. Altman, *Constrained Markov decision processes*. Routledge, 2021.

[15] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*. PMLR, 2017, pp. 22–31.

[16] L. Shen, L. Yang, S. Chen, B. Yuan, X. Wang, D. Tao, *et al.*, "Penalized proximal policy optimization for safe reinforcement learning," *arXiv preprint arXiv:2205.11814*, 2022.

[17] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," *arXiv preprint arXiv:1910.01708*, vol. 7, no. 1, p. 2, 2019.

[18] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Projection-based constrained policy optimization," *arXiv preprint arXiv:2010.03152*, 2020.

[19] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, "Risk-constrained reinforcement learning with percentile risk criteria," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6070–6120, 2017.

[20] Q. Liang, F. Que, and E. Modiano, "Accelerated primal-dual policy optimization for safe reinforcement learning," *arXiv preprint arXiv:1802.06480*, 2018.

[21] T. Xu, Y. Liang, and G. Lan, "Crpo: A new approach for safe reinforcement learning with convergence guarantee," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 480–11 491.

[22] Y. Zhang, Q. Vuong, and K. Ross, "First order constrained optimization in policy space," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 338–15 349, 2020.

[23] 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*. PMLR, 2022, pp. 91–100.

[24] G. Valsecchi, N. Rudin, L. Nachtigall, K. Mayer, F. Tischhauser, and M. Hutter, "Barry: a high-payload and agile quadruped robot," *IEEE Robotics and Automation Letters*, 2023.

[25] M. Bjelonic, P. K. Sankar, C. D. Bellicoso, H. Vallery, and M. Hutter, "Rolling in the deep–hybrid locomotion for wheeled-legged robots using online trajectory optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3626–3633, 2020.

[26] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, *et al.*, "What matters in on-policy reinforcement learning? a large-scale empirical study," *arXiv preprint arXiv:2006.05990*, 2020.

[27] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[28] S. Mysore, G. Cheng, Y. Zhao, K. Saenko, and M. Wu, "Multi-critic actor learning: Teaching rl policies to act with style," in *International Conference on Learning Representations*, 2021.