TARC: Time-Adaptive Robotic Control *

Arnav Sukhija[1], Lenart Treven[1], Jin Cheng[1], Florian Dörfler[2], Stelian Coros[1], Andreas Krause[1]

[1]Department of Computer Science, ETH Zurich, Switzerland
[2]Dept. of Information Technology and Electrical Engineering, ETH Zurich, Switzerland
{asukhija, trevenl, jicheng, scoros, krausea}@ethz.ch, dorfler@ethz.ch

Abstract

Fixed-frequency control in robotics imposes a trade-off between the efficiency of low-frequency control and the robustness of high-frequency control, a limitation not seen in adaptable biological systems. We address this with a reinforcement learning approach in which policies jointly select control actions and their application durations, enabling robots to autonomously modulate their control frequency in response to situational demands. We validate our method with zero-shot sim-to-real experiments on two distinct hardware platforms: a high-speed RC car and a quadrupedal robot. Our method matches or outperforms fixed-frequency baselines in terms of rewards while significantly reducing the control frequency and exhibiting adaptive frequency control under real-world conditions.

1 Introduction

Efficient and adaptive control is a hallmark of intelligent systems. Biological systems, for instance, masterfully adapt their control effort to the complexity of a situation [Crevecoeur et al., 2020, Babič et al., 2016]. Walking on a sidewalk requires little conscious control, our steps are automatic, and corrections are infrequent. In contrast, walking on a slack line requires constant attention, frequent corrections, and adjustments to maintain balance. This ability to dynamically modulate control frequency is fundamental to the robustness and efficiency of human motor skills [Crevecoeur et al., 2020].

In sharp contrast, most existing control methods in robotics rely on a fixed control frequency in all scenarios [Hwangbo et al., 2019, Di Carlo et al., 2018, Wu et al., 2023, Schuchert and Karimi, 2024, Oleiwi et al., 2025]. For example, a self-driving car is controlled at the same frequency when driving straight compared to when performing a complicated drift maneuver. Although adaptability has been extensively studied in human-robot interaction and collaborative contexts [Schneider and Kummert, 2021, Tanevska et al., 2020, Nikolaidis et al., 2017], this dynamic modulation of control frequency remains under-addressed.

The fixed-frequency approach, in turn, forces a fundamental compromise. A too low control frequency, while computationally cheap, risks failure when the controller must react to sudden changes in its state or environment. Conversely, a too high frequency ensures robustness and reactivity but comes at a significant and often unnecessary computational cost. In robotics, this trade-off leads to a preference for the latter approach, accepting a high computational burden to guarantee performance in the worst-case scenario [Hwangbo et al., 2019, Di Carlo et al., 2018, Wu et al., 2023, Schuchert and Karimi, 2024, Oleiwi et al., 2025].

In this work, we present **T**ime-**A**daptive **R**obotic **C**ontrol (TARC), a reinforcement learning (RL) approach that enables robots to modulate their control frequency. Our policies learn to jointly output

*Project website: https://arnavsukhija.github.io/projects/tarc/. Supplementary video: https://youtu.be/w0y6uusnPYc

2nd Workshop on Aligning Reinforcement Learning Experimentalists and Theorists (ARLET 2025).

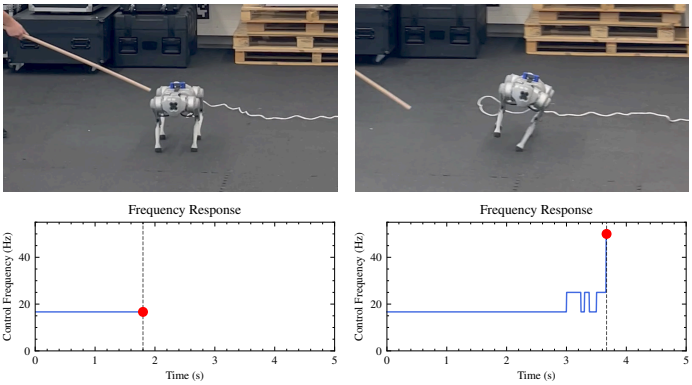

Figure 1: The performance of TARC on scenarios requiring adaptation in control frequency. The *quadruped*'s control frequency spikes when experiencing a push, demonstrating state-dependent frequency modulation.

a control action and its application duration, building on the TaCoS framework [Treven et al., 2024], which allows this joint output to be learned using any standard RL algorithm. We demonstrate the effectiveness and generalizability of TARC via zero-shot sim-to-real deployment on two distinct dynamic platforms. Its core capability of modulating control frequency in response to situational demands is illustrated in Fig. 1. Our key contributions are summarized as follows:

- We introduce an adaptive frequency control method and validate it on two dynamically distinct robotic tasks: a high-speed RC car drifting and quadrupedal locomotion.
- We demonstrate that such adaptive policies achieve task performance comparable to fixed-frequency baselines while requiring less control interventions.
- We show that our learned policies autonomously modulate their control frequency in response to situational demands.

2 Related Work

2.1 Choice of Control Frequency in Robotic Systems

Traditional robotic control schemes often operate at a fixed frequency, which introduces several practical limitations. Fixed-frequency controllers can be wasteful if the frequency is too high, and their limitations have been extensively addressed in recent literature.Park et al. [2021] and Amin et al. [2021] show that variable action durations can significantly improve learning performance. Although high frequencies can ensure reactivity, they incur significant computational costs and can hinder training [Karimi et al., 2023, Wang and Beltrame, 2024a]. In contrast, low-frequency control is efficient and can improve latency insensitivity [Gangapurwala et al., 2023], but it risks instability when rapid reactions are needed. These challenges are not unique to learning-based methods; Nguyen et al. [2025] challenge the "one frequency for all" intuition by using different control frequencies for different scenarios. They consider a dynamic jumping maneuver on legged robots using Model Predictive Control (MPC) and observe more robust results with different control frequencies for the in-flight phase and the ground phase. Similarly, Li et al. [2025b] use a two-agent framework for humanoid locomotion, where a high-frequency agent stabilizes the upper body while a lower-frequency agent controls the gait, highlighting the benefit of specialized frequencies for tasks with mismatched dynamics. These previous works highlight the trade-offs of frequency selection and motivate the need for more flexible, adaptive control strategies.

2.2 Adaptive Frequency Control

To overcome these limitations, recent research has explored methods that adapt the control frequency during run-time. A significant amount of research has been conducted on repetitive action reinforce-

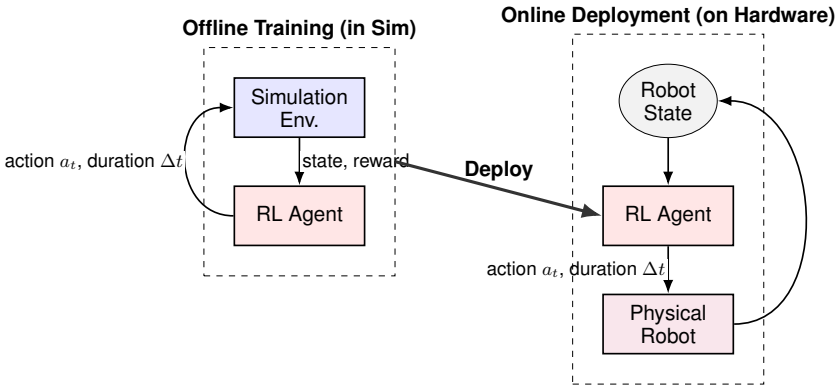

Figure 2: Overview of our framework. The RL agent outputs an action a_t and an application duration Δt, which allows the policy to modulate control frequency. We train completely offline in simulation (left), and deploy the policy zero-shot on the hardware (right).

ment learning, which enables agents to repeat actions over multiple time steps [Hafner et al., 2020, Braylan et al., 2015, Sharma et al., 2017, Metelli et al., 2020]. Furthermore, Wang and Beltrame [2024b,a] explore variable time-step reinforcement learning with adaptive control frequencies where the agent adapts the control frequencies based on task requirements. Their method extends Soft-Actor-Critic (SAC) [Haarnoja et al., 2018] with variable time steps and reward shaping, enabling the agent to select both which action to take and its execution duration using additional hyperparameter tuning. Similar concepts have appeared in MPC, where neural-augmented methods simultaneously optimize step location, step duration, and contact forces for natural variable-frequency locomotion [Li et al., 2025a].

In contrast, our work generalizes these concepts. We use the flexible TaCoS framework [Treven et al., 2024], which formalizes an extended Markov Decision Process (MDP) that any standard RL algorithm can solve without extensive reward shaping or hyperparameter adjustment.

3 Method

In this section, we describe our methodology for learning time-adaptive control policies that jointly predict the next action and their application durations. Fig. 2 provides an overview of our methodology.

3.1 Time-Adaptive Control

To obtain time-adaptive control policies, we adopt the interaction cost setting of the ***T**ime-**a**daptive **Co**ntrol and **S**ensing* (TaCoS) framework [Treven et al., 2024]. TaCoS transforms a continuous-time MDP to an extended discrete-time MDP which any standard RL algorithm can solve.

Although the intricate system dynamics behind both of our robotic platforms is continuous-time, their hardware implementations work in discrete-time, which means that control is applied at a fixed frequency f_{max}. Thus, we adapt the framework to a discrete-time setting in which the agent may choose to repeat actions over multiple control steps. This allows the policy to implicitly select the applied control frequency.

3.1.1 Policy Class

Let $i \in \mathbb{N}$ denote the maximum repetition count. In our setting, this corresponds to the maximum number of control steps that the agent can select as the action's application duration. The policy class we consider extends a standard action-selection policy to include time-adaptive duration selection:

Definition 1 (Time-Adaptive Policy π) *Let $\mathcal{X}$ be the state space, $\mathcal{A}$ the action space, and $T \in \mathbb{N}$ the episode horizon. At step $t \in \{0, 1, .., T\}$, let the agent observe the state $s_t = (x_t, t) \in \mathcal{X} \times \{0, 1, .., T\}$, where $x_t \in \mathcal{X}$ is the environment state. A **time-adaptive policy** is a function*

$\pi : \mathcal{X} \times \{0, 1, .., T\} \to \mathcal{A} \times \{1, 2, .., i\}$ *such that*

$$\pi(s_t) = (a_t, \Delta t) \tag{1}$$

where $a_t \in \mathcal{A}$ *is the action and* $\Delta t \in \{1, 2, .., i\}$ *the application duration.*

We refer to a controller using such a policy as **TARC-**i. Since control actions are kept constant for the Δt time steps, the resulting control frequency dynamically varies as

$$f = \frac{f_{max}}{\Delta t} \in \left[\frac{f_{max}}{i}, f_{max} \right] \tag{2}$$

3.1.2 Switch Cost and Reward

To encourage sparser control intervention and thus lower control frequencies, we penalize the agent for each action switch with a fixed cost $c \in \mathbb{R}$ (see the interaction cost formulation in the TaCoS paper [Treven et al., 2024]). The goal is to discourage high-frequency control actions unless they are necessary for task performance.

Definition 2 (Reward function R**)** *Let* $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ *denote the original reward function of the underlying MDP,* $\gamma \in (0, 1]$ *be the discount factor,* $c \in \mathbb{R}$ *the switch cost,* $s_t = (x_t, t)$ *the augmented state at time t,* $u_t = (a_t, \Delta t)$ *the augmented action. We define the reward function* R *of our discrete-time augmented MDP as:*

$$R(s_t, u_t) = \left(\sum_{k=0}^{\Delta t - 1} \gamma^k r(x_{t+k}, a_t) \right) - c \tag{3}$$

where

- x_{t+k} *is the state observed at time* $t + k$ *by repeatedly executing* a_t
- a_t *is held constant for* Δt *steps*

This reward structure encourages the agent to maximize task performance while minimizing unnecessary interventions through the switch penalty. This hyperparameter is tuned to align with the specific control requirements, the degree to which low control frequencies are encouraged, and the characteristics of the robotic platform.

If c is too low, the agent tends to favor high-frequency control, similar to the fixed-rate baseline. In contrast, a very high c can discourage necessary control updates, causing the agent to repeat actions excessively or act conservatively. Selecting an appropriate value of c is important to balance the costs of control intervention with responsiveness.

For each robotic platform, we provide the values of c used in our experiments in section 4, while a complete list is also available in appendix A.

4 Experimental Setup

For both platforms, we compare our TARC policies with a fixed-frequency baseline controller operating at the base control frequency f_{max}. The baseline simply picks an action at each step and acts within the standard, non-augmented MDP.

Training is done offline with the help of simulators and deployed zero-shot on to the real hardware platform. All policies are optimized using Proximal Policy Optimization (PPO) [Schulman et al., 2017, Freeman et al., 2021]. We consider several variants of adaptive policies as per Definition 1, specifically, **TARC-3, TARC-4, TARC-5, TARC-10**.

For our evaluations, we consider the following performance metrics, both directly reported on the hardware in Section 5:

- **Total Reward**: We report the cumulative reward per episode under two conditions. The **penalized reward** includes the switch cost c to evaluate how well the policy balances task performance against the cost of frequent control interventions. The **unpenalized reward** excludes this cost to provide a direct measure of pure task performance.

- **Average Control Frequency**: The average number of control actions applied per second (Hz). This metric directly quantifies the policy's intervention rate.

4.1 RC Car

The RC car is similar to those of Bhardwaj et al. [2024], Sukhija et al. [2023]. The RL task is a reverse parking maneuver: starting approximately 2m from a target position, the car must rotate 180°, typically requiring a drift (see Fig. 3). This scenario is chosen to have high and low control demand, so that a non-adaptive f_{max} baseline can be compared against the adaptive controller. This task is considered over one episode of 200 control steps at base frequency of $f_{max} = 30\text{Hz}$, which corresponds to $\frac{200}{30} = 6.6$ seconds.

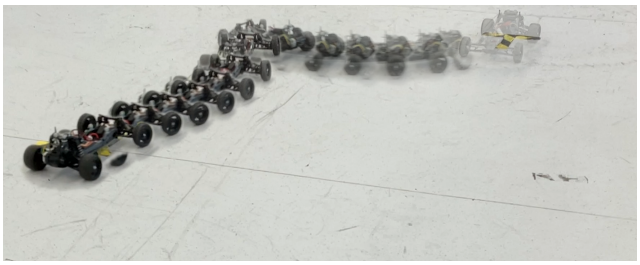

Figure 3: Desired reverse parking maneuver which involves rotating the car 180°and parking approx. 2m away.

The dynamics model is based on Kabzan et al. [2020] and uses the Pacejka tire model [Pacejka and Bakker, 1992] to accurately capture drifting effects. We employ domain randomization over physical parameters such as mass, tire stiffness, and motor power to improve robustness. Furthermore, to account for an observed hardware delay 80ms, we augment the 6D physical state with the last three actions, forming a 12 dimensional base observation vector. As per Definition 1 we augment this 12 dimensional observation vector with an additional time component for the TARC policies. The full details of the state space and randomization ranges are in appendix A.1.

The reward function, based on the work of Rothfuss et al. [2024], is:

$$r(x_t, a_t) = r_{state}(x_t) - w \cdot ||a_t||^2 \tag{4}$$

where r_{state} is a tolerance-based reward that encourages closeness to the target pose. We use a control penalty weight of $w = 0.005$. This reward function is used as the underlying reward function for our method according to Definition 2, using an empirically tuned switch cost of $c = 0.1$. The complete reward formulation and training hyperparameters are detailed in appendix A.1.

4.2 The quadruped

For the quadrupedal locomotion task, we use the Unitree Go1, a robot with a 48-dimensional state space and a 12-dimensional action space. We use MuJoCo Playground [Zakka et al., 2025], known for its robust sim-to-real transfer, allowing policies to be trained in simulation and deployed zero-shot onto the hardware with control actions issued at a base control frequency of $f_{max} = 50\text{Hz}$.

4.2.1 Training Setup

Policies are trained on the joystick locomotion task [Ji et al., 2022, Rudin et al., 2022], where the agent learns to map high-level velocity commands to low-level joint actions. Our training is based on the *JoystickFlatTerrain* environment from MuJoCo Playground, adopting its reward function and base hyperparameters. Our TARC method extends this setup, enabling the policy to dynamically select its intervention rate using an empirically tuned switch cost of $c = 0.005$. Full details on the training procedure and TARC implementation are provided in appendix A.2.

4.2.2 Evaluation Scenarios

To evaluate generalization and the effectiveness of our time-adaptive policies, we deploy our policies on the hardware and test them on three distinct scenarios, each conducted over a 20-second episode.

Importantly, these specific command profiles were not encountered during training as such, the agent was simply trained to map joystick commands to actual joint targets, and we test the performance of this in these three different scenarios:

- **Gentle Curve**: The robot receives time-varying joystick commands that guide it along a smooth and predictable low-speed curve.
- **Velocity Changes**: The robot is commanded to walk straight at varying speeds, exposing the agent to different magnitudes of the same direction command.
- **Run Then Turn** The robot begins with a running command, followed by an abrupt switch to a in-place turn command, testing the agent's reactivity and adaptability to sudden changes.

Representative command profiles for each scenario are shown in Fig. 4.

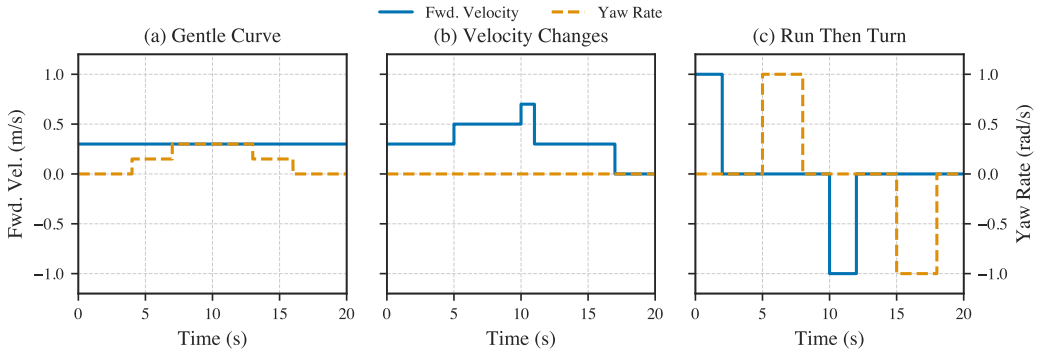

Figure 4: The three evaluation scenarios for the Unitree Go1, defined by their command profiles over a 20-second episode. Each plot shows the commanded forward velocity (solid blue line, left axis) and yaw rate (dashed orange line, right axis) over time for each scenario (a) Gentle Curve, (b) Velocity Changes, and (c) Run Then Turn.

5 Results and Discussion

In this section, we present the results of TARC applied to the RC car and Unitree Go1. For both platforms, we evaluate policy performance on total reward (with and without switch cost penalties) and average control frequency, reporting the mean and standard error over 5 random seeds.

5.1 RC Car

We first analyze the hardware performance of the RC car using the reward formulation defined in (4) and the formulation of Definition 2. Our results are summarized in Fig. 5. The TARC policies achieve a higher penalized reward than the baseline (Fig. 5a), a direct result of their lower control frequencies (Fig. 5c) which incur fewer switching penalties. In the unpenalized setting (Fig. 5b), while the mean rewards are comparable, the TARC policies exhibit significantly smaller variance, indicating more robust and consistent hardware performance.

The notable exception is TARC-10, which shows a clear degradation in hardware reward. This result is counterintuitive; According to our formulation in Definition 1 and (2), a higher maximum repetition count i provides the policy with the greatest flexibility by giving it access to a larger range of control frequencies. With this flexibility, the TARC-10 agent could have chosen to operate at a higher frequency if it were optimal. However, the simulation and hardware comparison in Fig. 6 reveals why this did not translate to better real-world performance. Although all TARC policies perform strongly in simulation (Fig. 6(a,b)), the figure vividly illustrates the amplified sim-to-real gap for TARC-10. This confirms that its longer open-loop control intervals (up to 10 steps) are more sensitive to inaccuracies in the approximated dynamics.

Finally, Fig. 5c shows that TARC policies operate at less than half the frequency of the baseline. Furthermore, Fig. 6c reveals that the policy's learned control frequency strategy transfers almost

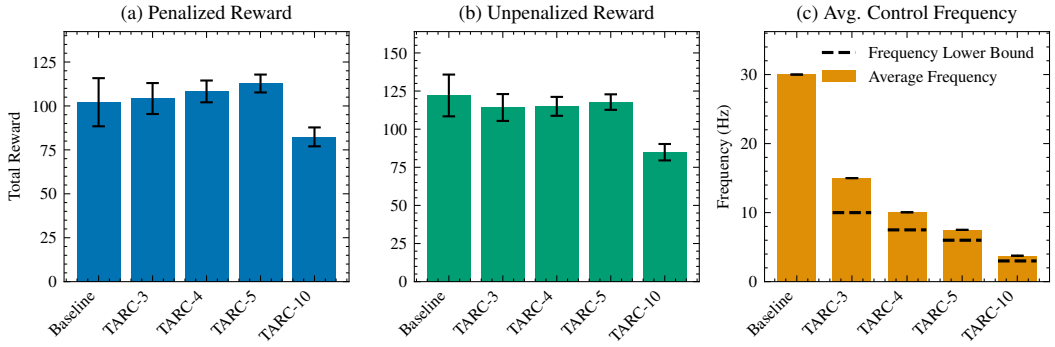

Figure 5: Performance comparison of controllers on the RC Car task averaged over 5 seeds. The subplots show: (a) Total reward penalized with $c = 0.1$. (b) Total reward excluding the penalty (c) Average control frequency over the episode

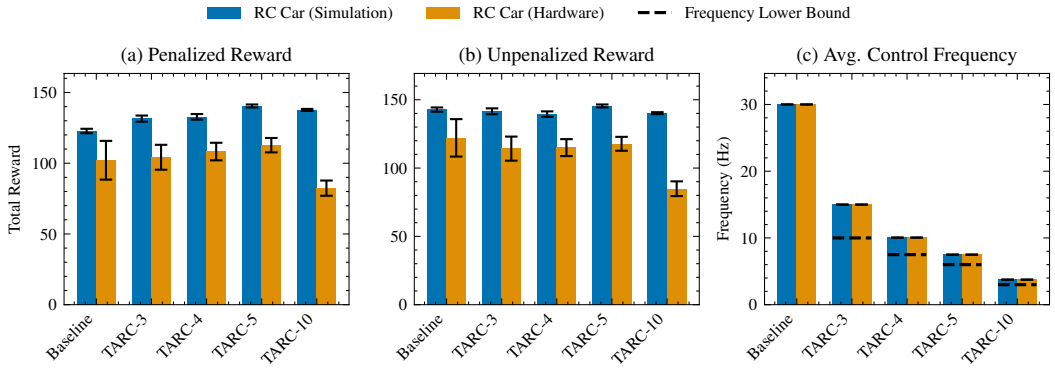

Figure 6: Comparison of Simulation and Hardware performance for the TARC policies on the RC car. (a, b) A sim-to-real gap in reward is visible for all controllers and is most pronounced for TARC-10, highlighting the effect of longer open-loop execution. (c) In contrast, the learned average control frequency transfers almost perfectly from simulation to hardware, demonstrating the robustness of the learned frequency modulation strategy.

perfectly. The average frequencies selected by the TARC agents are nearly identical between the simulation and the hardware, with negligible variance. For this task, agents in both domains converge to a low-frequency, consistent strategy as the optimal solution. However, unlike fixed action repetition methods, our approach allows the policy to discover the optimal rate autonomously, illustrating the benefit of learning state-dependent control durations.

In addition to frequency and reward, we evaluated the smoothness of the motor commands on hardware by measuring the jitter $||a_t - a_{t-1}||$, that is, the change in motor input between two consecutive steps over time. We report the mean and standard error of this data across 3 seeds on the TARC-4 policies. As shown in Fig. 7, TARC policies produce smaller changes in motor command compared to the high-frequency baseline. This reduction in throttle delta demonstrates that high-frequency controllers lead to unnecessarily rapid actuation changes, which can cause increased mechanical wear, energy inefficiency, and control noise in hardware systems. By learning to minimize the intervention rate, TARC not only reduces computational and switching penalties, but also delivers smoother, more hardware-friendly actuation patterns.

5.2 The Unitree Go1

We evaluate our time-adaptive control approach on the Unitree Go1 across the three scenarios represented in Fig. 4: Gentle Curve, Velocity Changes, and Run Then Turn. Using the reward function from MuJoCo Playground [Zakka et al., 2025] and the reward formulation according to

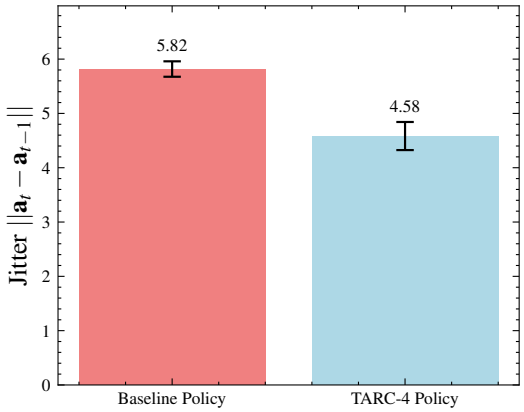

Figure 7: Jitter of the throttle data across time for TARC policies and the fixed-frequency baseline. TARC produces smoother motor commands with lower deltas, showing that reducing high-frequency interventions avoids unnecessary command oscillations.

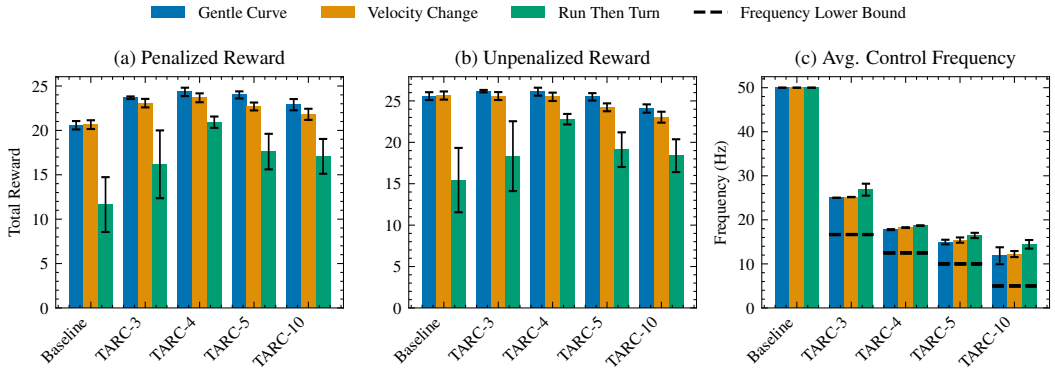

Figure 8: Performance and efficiency metrics for the Unitree Go1 across the three evaluation scenarios. The subplots show a comparison of the baseline controller against the TARC variants on: (a) total reward with a switch cost of 0.005, (b) total reward without the penalty, and (c) the resulting average control frequency.

Definition 2, we assess performance in both penalized and unpenalized settings and measure the average control frequency per episode. The results are summarized in Fig. 8.

Fig. 8a shows that adaptive controllers achieve higher penalized rewards than the baseline in all scenarios. The differences between the TARC policies are smaller than those observed on the RC car, likely due to the lower switch cost $c = 0.005$ employed and the superior sim-to-real transfer to the Go1 using the powerful MuJoCo Playground simulator. Among the adaptive controllers, TARC-4 consistently attains the best performance with the highest penalized reward and the smallest standard error. The baseline performs particularly poorly in the more challenging Run Then Turn scenario, underscoring the robustness benefits of frequency adaptation. Additionally, TARC-4, TARC-5, and TARC-10 exhibit reduced variance compared to the baseline, highlighting their more consistent performance. In the unpenalized setting (Fig. 8b), the rewards are broadly comparable, although TARC-10's performance is slightly lower. This again suggests limitations of open-loop action repetition under greater durations with sim-to-real transfer, the impact although, remains marginal due to the accurate simulation environment.

The frequency analysis in Fig. 8c indicates that the average control frequency can be reduced by at least half relative to the baseline, reflecting what we observe on the RC car. Frequencies are lower on the simpler Gentle Curve scenario and increase for Velocity Changes and Run Then Turn, reflecting adaptive modulation in response to state difficulty. Furthermore, despite greater frequency

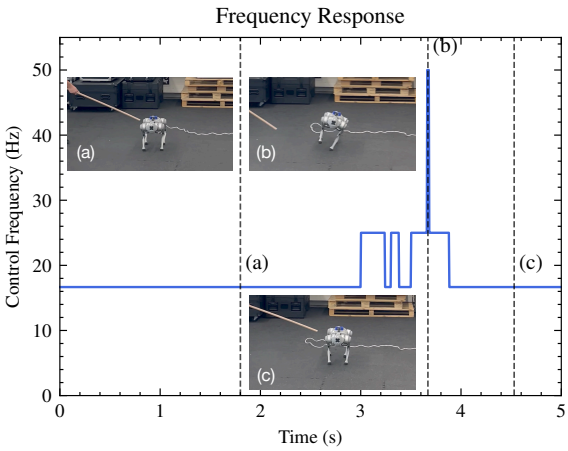

Figure 9: The frequency response of **TARC-4** to an external perturbation on the Go1. We observe the controller uses a low frequency (16.7Hz) in a stable standstill (a, c) but immediately increases its frequency to the maximum (50Hz) to counteract the push while airborne (b). This dynamic modulation is a key benefit of adaptive frequency control.

flexibility, the differences in control frequency between TARC-4, TARC-5, and TARC-10 remain modest, demonstrating the agent's capacity to select higher frequencies when necessary.

To further demonstrate adaptability and robustness, we subject the controllers to perturbations and analyze their control frequency responses. Fig. 9 vividly illustrates the resulting adaptive behavior. The policy maintains a low and efficient control rate of 16.7Hz during standstill (Fig. 9a). When pushed, it instantaneously increases its frequency to the maximum 50Hz to handle the more complex critical state in the air and ensure a stable recovery (Fig. 9b). As soon as a stable state is restored, the frequency immediately returns to the low rate (Fig. 9c). In contrast, a fixed-frequency baseline would have used 50Hz throughout the entire scenario, incurring unnecessary costs during periods of stability.

6 Conclusion

In this work, we presented Time-Adaptive Robotic Control (TARC), an adaptive frequency control approach. TARC enables reinforcement learning policies to jointly optimize both control actions and their application durations. This allows the agent to dynamically modulate its control frequency, naturally balancing performance and efficiency without requiring specialized reward shaping.

We validated our approach via zero-shot sim-to-real deployment on two distinct robotic platforms, a drifting RC car and a quadruped. Across these platforms, our method matched or outperformed the baselines in terms of rewards while significantly reducing the control frequency. Furthermore, we exposed the Go1 to perturbations that revealed a state-dependent adaptation in control frequency, similar to how humans increase effort in challenging situations.

A key limitation of our current approach is the reliance on a fixed, empirically-tuned switch cost, which requires manual selection for new platforms. A promising direction for future work is to explore state-action-dependent switch cost functions $c(x_t, a_t)$, which would allow the policy to learn a nuanced trade-off between making cheap exploratory changes in safe states and heavily penalizing large and destabilizing actions in critical ones.

7 Acknowledgements

This project has received funding from the Swiss National Science Foundation under NCCR Automation, grant agreement 51NF40 180545.

References

Susan Amin, Maziar Gomrokchi, Hossein Aboutalebi, Harsh Satija, and Doina Precup. Locally persistent exploration in continuous control tasks with sparse rewards. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 275–285. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/amin21a.html.

Jan Babič, Erhan Oztop, and Mitsuo Kawato. Human motor adaptation in whole body motion. *Scientific Reports*, 6(1):32868, 2016. doi: 10.1038/srep32868. URL https://doi.org/10.1038/srep32868.

Arjun Bhardwaj, Jonas Rothfuss, Bhavya Sukhija, Yarden As, Marco Hutter, Stelian Coros, and Andreas Krause. Data-efficient task generalization via probabilistic model-based meta reinforcement learning. *IEEE Robotics and Automation Letters*, 9(4):3918–3925, 2024. doi: 10.1109/LRA.2024.3371260.

Alexander Braylan, Mark Hollenbeck, Elliot Meyerson, and Risto Miikkulainen. Frame skip is a powerful parameter for learning to play atari. In *AAAI-15 Workshop on Learning for General Competency in Video Games*, 2015. URL http://nn.cs.utexas.edu/?braylan:aaai15ws.

F. Crevecoeur, J. L. Thonnard, and P. Lefèvre. A very fast time scale of human motor adaptation: Within movement adjustments of internal representations during reaching. *eNeuro*, 7(1):ENEURO.0149–19.2019, 2020. doi: 10.1523/ENEURO.0149-19.2019. URL https://doi.org/10.1523/ENEURO.0149-19.2019.

Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi: 10.1109/IROS.2018.8594448.

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=VdvDlnnjzIN.

Siddhant Gangapurwala, Luigi Campanaro, and Ioannis Havoutis. Learning low-frequency motion control for robust and dynamic robot locomotion. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5085–5091, 2023. doi: 10.1109/ICRA48891.2023.10160357.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26), 2019. ISSN 2470-9476. doi: 10.1126/scirobotics.aau5872. URL http://dx.doi.org/10.1126/scirobotics.aau5872.

Gwanghyeon Ji, Juhyeok Mun, Hyeongjun Kim, and Jemin Hwangbo. Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion. *IEEE Robotics and Automation Lettaers*, 7(2):4630–4637, April 2022. ISSN 2377-3774. doi: 10.1109/lra.2022.3151396. URL http://dx.doi.org/10.1109/LRA.2022.3151396.

Juraj Kabzan, Miguel I. Valls, Victor J. F. Reijgwart, Hubertus F. C. Hendrikx, Claas Ehmke, Manish Prajapat, Andreas Bühler, Nikhil Gosala, Mehak Gupta, Ramya Sivanesan, Ankit Dhall, Eugenio Chisari, Napat Karnchanachari, Sonja Brits, Manuel Dangel, Inkyu Sa, Renaud Dubé, Abel Gawel, Mark Pfeiffer, Alexander Liniger, John Lygeros, and Roland Siegwart. Amz driverless:

The full autonomous racing system. *Journal of Field Robotics*, 37(7):1267–1294, 2020. doi: https://doi.org/10.1002/rob.21977. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21977.

Amirmohammad Karimi, Jun Jin, Jun Luo, A. Rupam Mahmood, Martin Jagersand, and Samuele Tosatto. Dynamic decision frequency with continuous options. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7545–7552, 2023. doi: 10.1109/IROS55552.2023.10342408.

Junheng Li, Ziwei Duan, Junchao Ma, and Quan Nguyen. Gait-net-augmented implicit kino-dynamic mpc for dynamic variable-frequency humanoid locomotion over discrete terrains, 2025a. URL https://arxiv.org/abs/2502.02934.

Yitang Li, Yuanhang Zhang, Wenli Xiao, Chaoyi Pan, Haoyang Weng, Guanqi He, Tairan He, and Guanya Shi. Hold my beer: Learning gentle humanoid locomotion and end-effector stabilization control. In *RSS 2025 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*, 2025b. URL https://openreview.net/forum?id=2ajP0TMxi7.

Alberto Maria Metelli, Flavio Mazzolini, Lorenzo Bisi, Luca Sabbioni, and Marcello Restelli. Control frequency adaptation via action persistence in batch reinforcement learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6862–6873. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/metelli20a.html.

Chuong Nguyen, Abdullah Altawaitan, Thai Duong, Nikolay Atanasov, and Quan Nguyen. Variable-frequency model learning and predictive control for jumping maneuvers on legged robots. *IEEE Robotics and Automation Letters*, 10(2):1321–1328, 2025. doi: 10.1109/LRA.2024.3519864.

Stefanos Nikolaidis, David Hsu, and Siddhartha S. Srinivasa. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36(5-7):618–634, 2017. doi: 10.1177/0278364917690593. URL https://doi.org/10.1177/0278364917690593.

Bashra Kadhim Oleiwi, Mohamed Jasim, Ahmad Taher Azar, Saim Ahmed, and Ahmed Redha Mahlous. Bat optimization of hybrid neural network-fopid controllers for robust robot manipulator control. *Frontiers in Robotics and AI*, Volume 12 - 2025, 2025. ISSN 2296-9144. doi: 10.3389/frobt.2025.1487844. URL https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2025.1487844.

Hans B. Pacejka and Egbert Bakker. The magic formula tyre model. *Vehicle System Dynamics*, 21 (sup001):1–18, 1992. doi: 10.1080/00423119208969994. URL https://doi.org/10.1080/00423119208969994.

Seohong Park, Jaekyeom Kim, and Gunhee Kim. Time discretization-invariant safe action repetition for policy gradient methods. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 267–279. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/024677efb8e4aee2eaeef17b54695bbe-Paper.pdf.

Jonas Rothfuss, Bhavya Sukhija, Lenart Treven, Florian Dörfler, Stelian Coros, and Andreas Krause. Bridging the sim-to-real gap with bayesian inference. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10784–10791, 2024. doi: 10.1109/IROS58592.2024.10801505.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 91–100. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/rudin22a.html.

Sebastian Schneider and Franz Kummert. Comparing robot and human guided personalization: Adaptive exercise robots are perceived as more competent and trustworthy. *International journal of social robotics*, 13(2):169–185, April 2021. ISSN 1875-4791. doi: 10.1007/s12369-020-00629-w.

Philippe Schuchert and Alireza Karimi. Data-driven frequency-based feedforward control design for a robotic arm joint. In *2024 American Control Conference (ACC)*, pages 3853–3858, 2024. doi: 10.23919/ACC60939.2024.10644602.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Sahil Sharma, Aravind S Lakshminarayanan, and Balaraman Ravindran. Learning to repeat: Fine grained action repetition for deep reinforcement learning. In *International Conference on Learning Representations*, 2017.

Bhavya Sukhija, Nathanael Köhler, Miguel Zamora, Simon Zimmermann, Sebastian Curi, Andreas Krause, and Stelian Coros. Gradient-based trajectory optimization with learned dynamics. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1011–1018, 2023. doi: 10.1109/ICRA48891.2023.10161574.

Anja Tanevska, Francesco Rea, Giulio Sandini, Lola Cañamero, and Alessandra Sciutti. A Socially Adaptable Framework for Human-Robot Interaction. *Frontiers in Robotics and AI*, 7:121, 2020. doi: 10.3389/frobt.2020.00121. URL https://doi.org/10.3389/frobt.2020.00121.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018. URL https://arxiv.org/abs/1801.00690.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017. doi: 10.1109/IROS.2017.8202133.

Lenart Treven, Bhavya Sukhija, Yarden As, Florian Dörfler, and Andreas Krause. When to sense and control? a time-adaptive approach for continuous-time rl. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 63654–63685. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/746b0a1e6e3cff8d968ba6d2e6fff049-Paper-Conference.pdf.

Dong Wang and Giovanni Beltrame. Moseac: Streamlined variable time step reinforcement learning, 2024a. URL https://arxiv.org/abs/2406.01521.

Dong Wang and Giovanni Beltrame. Reinforcement learning with elastic time steps, 2024b. URL https://arxiv.org/abs/2402.14961.

Yifan Wu, Wenkai Niu, Linghuan Kong, Xinbo Yu, and Wei He. Fixed-time neural network control of a robotic manipulator with input deadzone. *ISA Transactions*, 135:449–461, 2023. ISSN 0019-0578. doi: https://doi.org/10.1016/j.isatra.2022.09.030. URL https://www.sciencedirect.com/science/article/pii/S0019057822004876.

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. Mujoco playground, 2025. URL https://arxiv.org/abs/2502.08844.

A Experimental Details and Hyperparameters

This appendix provides additional details on the experimental setup and hyperparameters used for training the TARC policies.

A.1 RC Car

A.1.1 State Space

The physical state of the RC car is a 6-dimensional vector, $[p_x, p_y, \theta, v_x, v_y, \omega]$, containing its global position, orientation, local-frame velocity, and angular velocity. To account for the ~80 ms delay, this state is augmented with the three most recent actions, forming a 12-dimensional observation vector x_t that is provided to the policy network. For the TARC policies, this 12-dimensional observation vector is further augmented with an additional time component representing the elapsed number of control steps in this episode. The composition of this observation vector is detailed in Table 1.

Table 1: Components of the 13-D Observation Vector for the RC Car.

Dimension(s)	Description
0, 1	Global Position (p_x, p_y)
2	Orientation (θ)
3, 4	Local Velocity (v_x, v_y)
5	Angular Velocity (ω)
6, 7	Action at $t-1$ $(\delta_{t-1}, \tau_{t-1})$
8, 9	Action at $t-2$ $(\delta_{t-2}, \tau_{t-2})$
10, 11	Action at $t-3$ $(\delta_{t-3}, \tau_{t-3})$
12	Time component t (for TARC policies)

A.1.2 Action Space

The action space is a 2-dimensional continuous vector, $\mathbf{a} = [\delta, \tau]$, where each component is normalized to the range $[-1, 1]$. These normalized values are mapped to physical commands; the steering value δ is multiplied by a maximum steering angle parameter, and the throttle τ is scaled by a maximum throttle parameter.

- δ: Represents the normalized steering angle.
- τ: Represents the normalized motor throttle.

A.1.3 Reward Function

The goal of the movement is a target position $[x_0, y_0, \theta_0]$ that the car must reach. To that extent, we consider the following reward function:

$$r(x_t, a_t) = r_{state}(x_t) - w \cdot ||a_t|| \tag{5}$$

This reward function, the hyperparameters, and the simulation environment are the same as those used by Rothfuss et al. [2024][2]. The reward function is based on the tolerance reward from Tassa et al. [2018]. The tolerance function r_{state} gives higher rewards when the agent is close to a desired state, that is, in the case of the RC car, the target position. The "closeness" is quantified using a margin parameter for the reward function. Rothfuss et al. [2024] use a margin of 20, while also penalizing large steering angles and throttles with a weight factor w. In our experiments, we keep $w = 0.005$. For the tolerance function r_{state}, the "closeness" to the target is first quantified by a combined distance metric, d_{total}. This metric incorporates both the Euclidean distance to the target position (d_{pos}) and the shortest angular difference to the target orientation (d_θ), ensuring the car is rewarded for both correct placement and alignment. The total distance is calculated as $d_{\text{total}} = \sqrt{d_{\text{pos}}^2 + d_\theta^2}$.

This distance is then transformed into a reward using a "long-tail" function of the form:

$$r_{state}(d_{\text{total}}) = \frac{1}{(d_{\text{total}} \cdot s)^2 + 1} \tag{6}$$

where s is a scaling factor derived from the margin. This function provides a maximum reward of 1 when the car is exactly at the target pose ($d_{\text{total}} = 0$) and decays smoothly as the car moves away, creating a soft target zone rather than a hard binary objective.

[2]Official implementation: `https://github.com/lasgroup/simulation_transfer`

A.1.4 Simulation Dynamics

The environment incorporates a dynamics model based on Kabzan et al. [2020], using the Pacejka tire model [Pacejka and Bakker, 1992] to better model drifting and tire-road interaction effects. For better sim-to-real transfer, we use domain randomization: physical parameters such as center of mass, mass, tire stiffness (drifting), and motor power are sampled across episodes [Tobin et al., 2017], with ranges detailed in Table 2.

Table 2: Domain Randomization Ranges for the RC Car Simulation.

Parameter	**Randomization Range**
Mass (m)	[1.6, 1.7] kg
Inertia (I_{com})	[0.03, 0.18]
Front Tire Grip (d_f)	[0.25, 0.6]
Rear Tire Grip (d_r)	[0.15, 0.45]
Motor Constant (c_{m1})	[10.0, 40.0]
Steering Limit	[0.4, 0.75] rad

A.1.5 Hyperparameters

The policies were trained using Proximal Policy Optimization (PPO). The key hyperparameters are listed in Table 3.

Table 3: Training Hyperparameters for the RC Car.

Parameter	**Value**
Environment & Task Parameters	
Episode Steps	200
Number of Environments	2048
Total Timesteps	75×10^6
Number of Random Seeds	5
TARC-Specific Parameters	
Base Control Frequency (f_{max})	30 Hz
Switch Cost (c)	0.1
Lowest Control Frequency (f_{min})	$\{\frac{30}{3}, \frac{30}{4}, \frac{30}{5}, \frac{30}{10}\}$Hz
PPO Parameters	
Learning Rate (lr)	3e-4
Batch Size	1024
Number of Minibatches	32
Updates per Batch	4
Unroll Length	10
Entropy Cost	0.01
Discount Factor (γ)	0.9
GAE Lambda (λ)	0.95
PPO Clipping Epsilon (ϵ)	0.3
Optimizer	Adam
Observation Normalization	True
Advantage Normalization	True
Network Architecture	
Policy Hidden Layers	5×32
Critic Hidden Layers	4×128
Activation Function	Swish

A.2 Unitree Go1

This appendix provides supplementary details for the quadrupedal locomotion experiments.

For training, we use the *JoystickFlatTerrain* environment from MuJoCo Playground. The underlying MDP, including the 48-dimensional state space, 12-dimensional action space, reward function components, and base training hyperparameters (e.g., PPO parameters, network architecture, domain randomization ranges), are adopted from this framework without modification. The training process also leverages an asymmetric actor-critic setup [Tobin et al., 2017], where the policy (actor) and the value network (critic) receive different observations. We refer the reader to the MuJoCo Playground paper for further details of these components.

A.2.1 TARC Implementation Details

Our TARC method extends the base setup with the following modifications, which are crucial for our method to function:

- **State Observation**: For the TARC policies, the 48-dimensional state vector provided by the environment is augmented with an additional feature representing the elapsed time steps in the episode. This results in a final **49-dimensional observation vector** that is input to the policy network. Similarly, for the asymmetric setup, the 123-dimensional privileged state vector from the environment is also augmented with this time component, resulting in a 124-dimensional privileged state vector.
- **Switch Cost Tuning**: The switch cost was empirically tuned in simulation to balance task performance with control frequency. We observed that with switch costs lower than $c = 0.001$, the agent did not reduce its control frequency, while with costs greater than $c = 0.01$, the reward did not improve with more learning time and stayed low throughout the training process. A value of $\mathbf{c = 0.005}$ was found to provide the most effective trade-off.
- **Training Time**: For robustness, we trained the TARC policies for more time steps than the specification from MuJoCo Playground. We trained for 600M timesteps rather than 200M timesteps.

The key TARC-specific experimental parameters are summarized in Table 4.

Table 4: TARC specific parameters for the Go1

Parameter	**Value**
Base Control Frequency (f_{max})	50 Hz
Switch Cost (c)	0.005
Lowest Control Frequency (f_{min})	$\{\frac{50}{3}, \frac{50}{4}, \frac{50}{5}, \frac{50}{10}\}$Hz