# Learning and Deploying Robust Locomotion Policies with Minimal Dynamics Randomization

**Luigi Campanaro**     LUIGI@ROBOTS.OX.AC.UK
**Siddhant Gangapurwala**     SIDDHANT@ROBOTS.OX.AC.UK
**Wolfgang Merkt**     WOLFGANG@ROBOTS.OX.AC.UK
**Ioannis Havoutis**     IOANNIS@ROBOTS.OX.AC.UK
*Department of Engineering Science, University of Oxford*

## Abstract

Training Deep Reinforcement Learning (DRL) locomotion policies often require massive amounts of data to converge to the desired behavior. In this regard, simulators provide a cheap and abundant source. For successful sim-to-real transfer, exhaustively engineered approaches such as system identification, dynamics randomization, and domain adaptation are generally employed. As an alternative, we investigate a simple strategy of *random force injection* (RFI) to perturb system dynamics during training. We show that the application of random forces enables us to emulate dynamics randomization. This allows us to obtain locomotion policies that are robust to variations in system dynamics. We further extend RFI, referred to as extended random force injection (ERFI), by introducing an episodic actuation offset. We demonstrate that ERFI provides additional robustness for variations in system mass offering on average a 53% improved performance over RFI. We also show that ERFI is sufficient to perform a successful sim-to-real transfer on two different quadrupedal platforms, ANYmal C and Unitree A1, even for perceptive locomotion over uneven terrain in outdoor environments.

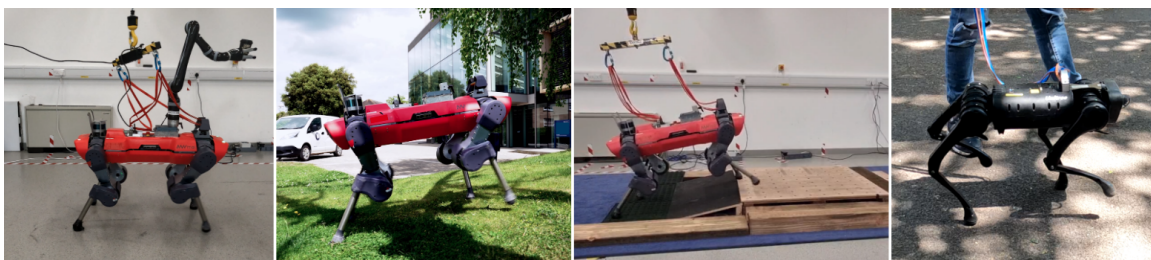**Keywords:** Sim-to-real; Legged Locomotion; Reinforcement Learning.

Figure 1: Deployment of the perceptive and blind locomotion policies on the ANYmal C and Unitree A1 quadrupedal platforms trained using our proposed ERFI-50 strategy without requiring actuation modeling or explicit randomization of dynamics or actuation properties.

## 1. Introduction

Deep reinforcement learning (DRL) has emerged as a promising approach for legged robotic control enabling highly dynamic and sophisticated locomotion capabilities (Lee et al., 2019; Yang et al., 2020; Kumar et al., 2021). The sample complexity associated with high-dimensional problems such as locomotion makes the use of physics simulators (Hwangbo et al., 2018a; Makoviychuk et al.,

2021b) appealing for training DRL control policies. This convenience, however, often requires addressing the *reality gap* between the simulated training domain and the physical target domain.

Strategies to address this reality gap often include identification of sensory noise which is then modeled and introduced in simulation during training (Jakobi et al., 1995; Hwangbo et al., 2019), or alternatively by including dropout during rollouts (Campanaro et al., 2023); accurate parameter identification (of properties such as Center of Mass (CoM), mass and inertia of robot links, impedance gains, system communication delays, and friction) for system modeling in addition to identification of relevant distributions suitable for domain randomization (Tan et al., 2018; Lee et al., 2019); and training a Neural Network (NN) to model the actuation dynamics of specific actuators, e.g. Series Elastic Actuators SEAs (Hwangbo et al., 2019; Lee et al., 2020).

As an alternative to exhaustive system identification and distribution identification for dynamics randomization, Valassakis et al. (2020) demonstrated captivating performance in sim-to-real for manipulation tasks using an extremely simple RFI strategy. RFI enables emulation of dynamics randomization through perturbation of system dynamics with randomized forces. However, as presented in Section 8, locomotion policies trained using RFI exhibit subpar robustness to policies trained with explicit dynamics randomization. To address this loss of performance, we present ERFI: ERFI allows to transfer locomotion controllers trained in simulation to the hardware by randomizing only two parameters: a random episodic actuation offset and random perturbations at each step. First, we show the efficacy of the approach proposed on legged systems, not covered in previous studies, second, we compare it to its predecessor RFI (Valassakis et al., 2020), to variations of the same method detailed in the following chapters and to standard domain randomization. Furthermore, we demonstrate with simulation experiments a significant performance improvement over RFI (mass variations' success rate +53%) especially scenarios unseen during training, which involves adding a manipulator arm on top of the robot at test time (mass variations' success rate +61%). Finally, we successfully deploy perceptive and blind policies trained in simulation with ERFI on to the physical ANYmal C and Unitree A1 quadrupeds. We show that training of actuator networks (mainly adopted for robots containing SEAs) and performing significant dynamics randomization (Peng et al., 2018), currently accepted as a standard for sim-to-real transfer can be substituted by a simple ERFI strategy. We test the controller's locomotion performance over flat and uneven terrain and further evaluate its robustness to additional mass and variation in CoM by mounting a Kinova arm on the robot's base.

## 2. Related Works

Actuators are an essential part of legged systems: they can be hydraulic (Semini et al., 2011), electric (Seok et al., 2013) and contain compliant elements (Hutter et al., 2016). Their dynamics is difficult to model involving nonlinear/nonsmooth dissipation, feedback loops and several internal states which are not directly observable. To accurately approximate SEAs, the authors of (Hwangbo et al., 2019) trained an actuator network able to output an estimated torque at the joints given as inputs a history of joint position errors and joint velocities recorded from the hardware. Modeling the actuation dynamics with NNs, for robots adopting SEAs, is now considered a standard and other works employed derivations of the same approach (Gangapurwala et al., 2020; Miki et al., 2022; Bohez et al., 2022). The limitations of learning the actuators' dynamics can be summarized in the need of recording motors' torques (not directly measurable for direct drives), training and testing the NN. In this context it is important to underline that direct drives motors are simpler to model

compared to SEAs and adopting an actuator network is not necessary, since classic system identification is enough. However, ERFI also removes the need for system identification by randomizing motors' torques.

Alongside actuator networks, the rise of highly dynamic controllers is driven by domain randomization, particularly dynamics randomization. Initially introduced in (Tobin et al., 2017; Peng et al., 2018), the approach consists in the randomization of some of the parameters of the robot's dynamics or of the environment. The additional robustness achieved can then compensate for discrepancies between simulation and the real world. In (Hwangbo et al., 2019) the domain randomization involves adding noise to the center of mass positions, the masses of links, and joint positions. In (Bohez et al., 2022) the randomization covers: mass, center of mass position, joint position, joint damping, joint friction, joint position tracking gains $K_p$, torque limits; while regarding the observations' perturbations: delays, joint position noise, angular velocity noise, linear acceleration noise and base orientation noise. Alongside with dynamics randomization, observations were also perturbed during training (Bohez et al., 2022) by adding delay, injecting noise into the joint positions, angular velocity, linear acceleration and base orientation. A significant randomization was also adopted in (Gangapurwala et al., 2020): gravity, actuation torque scaling, robot link mass scaling, robot link length scaling, random external forces at the base, gravity, actuation torque scaling, link mass scaling, link length scaling, actuation damping gain.

Considering the long list of parameters affected by randomization, the additional robustness it offers requires substantial efforts in system identification: especially in selecting the factors responsible of the reality gap (Valassakis et al., 2020) and in defining their randomization range; which if done incorrectly can severely affect the real world performances of the controller, leading to overly conservative policies (Xie et al., 2020).

An alternative technique – Random Force Injection – was proposed in (Valassakis et al., 2020), it aims to transfer policies trained in simulation to real systems without further tuning, with a limited number of parameters and it consists of injecting random forces into the simulator's dynamics. This method was tested on manipulation tasks, where it performed comparably to domain randomization. However, its potential was not evaluated for floating-base systems, especially when the overall stability is compromised by external perturbations.

## 3. Preliminaries

### 3.1. System Model

We model a quadrupedal system as a floating base $B$. The robot state is represented w.r.t. a reference frame $W$. We assume the $z$-axis of $W$, $\mathbf{e}_z^W$, aligns with the gravity axis. The base position is then expressed as $r_B \in \mathbb{R}^3$, and the orientation, $q_B \in SO(3)$, is represented by a unit quaternion. The corresponding rotation matrix is expressed as $\mathbf{R}_B \in SO(3)$. The angular positions of the rotational joints in each of the limbs are described by the vector $q_j \in \mathbb{R}^{n_j}$. For the quadrupeds considered in this work, $n_j = 12$. The linear and angular velocities of the base w.r.t. the global frame are written as $v_B \in \mathbb{R}^3$ and $\omega_B \in \mathbb{R}^3$ respectively. The generalized coordinates and velocities are thus expressed as q and u where

$$q = \begin{bmatrix} r_B \\ q_B \\ q_j \end{bmatrix} \in SE(3) \times \mathbb{R}^{n_j}, \quad u = \begin{bmatrix} v_B \\ \omega_B \\ \dot{q}_j \end{bmatrix} \in \mathbb{R}^{6+n_j}. \tag{1}$$

### 3.2. Impedance Control

In the context of this work, we consider a quadrupedal system is actuated using the joint control torques $\tau_j \in \mathbb{R}^{n_j}$. These torques are computed using the impedance control model given by

$$\tau_j = K_p(\mathrm{q}_j^* - \mathrm{q}_j) + K_d(\dot{\mathrm{q}}_j^* - \dot{\mathrm{q}}_j) + \tau_{j_{FF}}, \tag{2}$$

where $K_p$ and $K_d$ refer to the position and velocity tracking gains respectively, $\mathrm{q}_j^*$ is the vector representing desired joint positions, $\dot{\mathrm{q}}_j^*$, the desired joint velocities, and $\tau_{j_{FF}}$ refers to the feed-forward joint torques.

For locomotion, we train DRL control policies that modulate the joint actuation torques by generating $\mathrm{q}_j^*$. Additionally, we set $\dot{\mathrm{q}}_j^* = 0$ and $\tau_{j_{FF}} = 0$. Peng and van de Panne (2017) presented that such an approach offers more stable training and better performance than a torque controller. Equation 2 can thus be simplified to

$$\tau_j = K_p(\mathrm{q}_j^* - \mathrm{q}_j) - K_d\dot{\mathrm{q}}_j. \tag{3}$$

### 3.3. Rigid Body Dynamics Model

The rigid body dynamics model of a quadrupedal system can be expressed in the form of generalized equations of motion expressed as

$$\mathrm{M}\dot{\mathrm{u}} + \mathrm{h} = \mathbf{S}^T\tau_j + \mathrm{J}^T\lambda, \tag{4}$$

where $\mathrm{M} \in \mathbb{R}^{(6+n_j)\times(6+n_j)}$ is the mass matrix relative to the joints, $\mathrm{h} \in \mathbb{R}^{6+n_j}$ comprises Coriolis, centrifugal and gravity terms, $\mathbf{S}^T = [\mathbf{0}_{n_j\times 6} \ \ \mathbf{I}_{n_j\times n_j}]^T$, and J is the Jacobian which maps the contact forces $\lambda \in \mathbb{R}^{n_f}$ at $n_f = 4$ feet to generalized forces.

## 4. Extended Random Force Injection

Valassakis et al. (2020) investigated the effects of introducing random perturbations to a manipulation system. These *random force injections* aimed to diversify the visited states during training of DRL policies. In this regard, their implementation augmented the generalized equations of motion, similar to Equation 4, by random forces $f_r \sim \mathcal{U}(-f_r^{lim}, f_r^{lim})$ sampled from a uniform distribution $\mathcal{U}$ with limits $-f_r^{lim}$ and $f_r^{lim}$. These forces are sampled and applied at each time step to perturb the state transition $P$. In this work, we adapt this approach for quadrupedal systems and write Equation 4 with RFI as

$$\mathrm{M}\dot{\mathrm{u}} + \mathrm{h} = \mathbf{S}^T\tau_j + \mathrm{J}^T\lambda + f_r. \tag{5}$$

It is important to note that Valassakis et al. (2020) used this approach for a fixed-base system. In our preliminary experiments, for a mobile-base quadrupedal system, we observed that perturbing the robot's base with even small forces and torques resulted in convergence to undesired locomotion behavior. For the ANYmal C quadruped, forces and torques on the base sampled from distributions with $f_{r_b}^{lim} > 5\,\mathrm{N}$ and $\tau_{r_b}^{lim} > 3\,\mathrm{N\,m}$ respectively resulted in pronking behavior. Although this behavior was robust to external disturbances on the base, the pronking gait is energy inefficient and unsuitable for transfer to the physical system. Therefore, to better handle uncertainty in the system, we only introduce perturbations to the rotary joints of the quadruped and randomize forces on DoFs that we directly control.

The impedance controller described by Equation 2 is often executed at the actuation level at a higher frequency compared to the locomotion controller which is described by the DRL control policy mapping robot state information to desired joint positions. In this article, we refer to these frequencies as impedance control frequency and locomotion control frequency. We introduce perturbations at the impedance control frequency. We then split Equation 5, describing **RFI**, into the generalized equations of motion given by Equation 4 and an augmented impedance controller given by

$$\tau_j^r = K_p(\mathrm{q}_j^* - \mathrm{q}_j) - K_d \dot{\mathrm{q}}_j + \tau_{r_j}, \tag{6}$$

where $\tau_{r_j}$ refers to the random joint torque injections sampled from $\mathcal{U}(-\tau_{r_j}^{lim}, \tau_{r_j}^{lim})$ at each impedance control update step. Note that Equation 6 is only utilized during training. For deployment, we consider the actuation is governed by Equation 3.

In this work, we also investigate the effects of introduction of episodic actuation offsets during training. As opposed to randomizing $\tau_{r_j}$ at each impedance control step, we sample joint torque offsets $\tau_{o_j}$ from $\mathcal{U}(-\tau_{o_j}^{lim}, \tau_{o_j}^{lim})$, at the beginning of each training episode and apply them at each impedance control step. We refer to this as random actuation offset (**RAO**). This can be represented similarly as our implementation of RFI and is written as

$$\tau_j^o = K_p(\mathrm{q}_j^* - \mathrm{q}_j) - K_d \dot{\mathrm{q}}_j + \tau_{o_j}. \tag{7}$$

This constant offset enables us to emulate a shift in the robot's mass, inertia, impedance gains and contact Jacobian. However, unlike RFI, wherein the dynamics vary at each impedance control step resulting in a more reactive control behavior robust to temporally local perturbations, with RAO, the policy learns an implicit adaptive behavior for temporally global variations in system dynamics.

We also introduce an extended variant of RFI by combining RFI and RAO to learn control policies which can be robust to temporally local and global variations in system dynamics. We refer to this as **ERFI-Cumulative** (**ERFI-C**). In this case, we inject both a randomized force sampled at each impedance control step and an episodic actuation offset. The impedance controller with ERFI-C can be then written as

$$\tau_j^c = K_p(\mathrm{q}_j^* - \mathrm{q}_j) - K_d \dot{\mathrm{q}}_j + \tau_{r_j} + \tau_{o_j}. \tag{8}$$

We further explore another strategy with the same motivation as for ERFI-C. In this case, we only utilize RFI with 50% of the parallelized DRL training environments. The remaining environments employ RAO. We refer to this approach as **ERFI-50**. In comparison to ERFI-C, which can be considered as RFI with randomized distribution mean thereby resulting in a possibility of a learning bias for robustness to temporally local perturbations, ERFI-50 promotes unbiased learning of both local and global variations in system dynamics.

## 5. Why does ERFI work?

In Figure 2(a) and Figure 2(b), respectively, we show the effects of adding RFI and RAO as a feed-forward term of the PD controller ($K_p = 15$, $K_d = 1$) when commanding a step position change of $0.17 \, \mathrm{rad}$ ($\approx 10 \, \mathrm{deg}$) to the hind right knee.
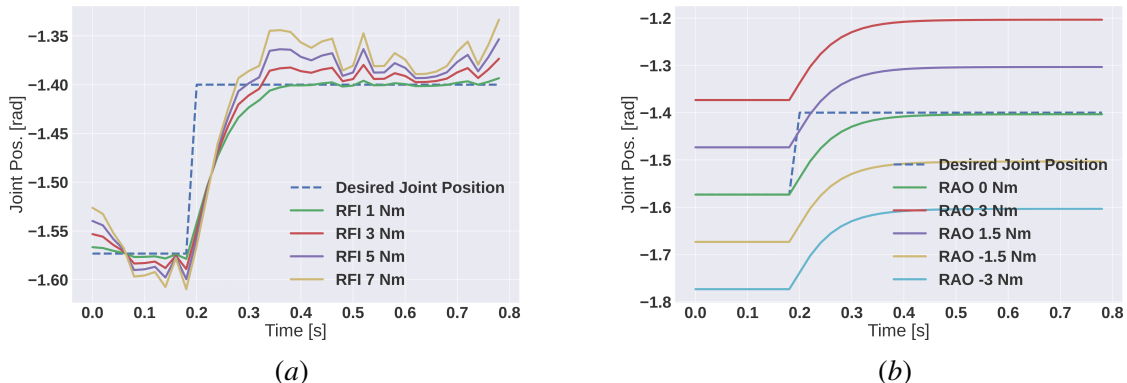
Figure 2: The magnitudes of $\tau_{r_j}^{lim}$ and $\tau_{o_j}^{lim}$ affect the dynamics of the system.

### 5.1. How does RFI model delays?

As can bee seen from Figure 2(a), the yellow line reaches the desired position faster than the green line, although the green line settles earlier. This implies that RFI adds stochasticity to the rise and settling times, i.e. it either increases or reduces the rise and settling times. The increase or decrease depends on the direction of the perturbation. This allows us to implicitly randomise actuation dynamics, especially parameters that relate to delays, friction and inertia (Section "ERFI robustness to delays" of the accompanying website).

### 5.2. How does RAO model mass and kinematic variations?

In Figure 2(b), the additional torque shifts the desired position of the joint and implicitly models offsets in the joint position (kinematics variations) or in the payload supported by the robot. Evidences of these effects can be found in Figure 4(a) and Figure 4(d) and video 3, 4, and 10 (on the accompanying website), demonstrating the robustness of the controllers even when the unmodelled payload reaches 42% of the total weight of the robot.

## 6. Problem Definition

The complex SEAs present on the ANYmal C quadruped exhibit a highly nonlinear behavior (Gehring et al., 2016). To address their complex dynamics, networks modeling the actuation became common practice in the community (Section 2). Evaluating the effectiveness of ERFI on such a platform thus provides a measure of the robustness of the method and of its generalization abilities.

Conversely, Unitree's A1 adopts quasi-direct drive actuators, which are affected by high levels of delay, signal noise and inaccurate tracking. Given the different technology adopted compared to SEAs and the canonical role that Unitree A1 has played in recent research works (Shao et al., 2022; Yang et al., 2022), we also investigated the effects of ERFI for obtaining locomotion policies for A1.

### 6.1. Perceptive Quadrupedal Locomotion

The ANYmal C robot is used to track a velocity command $[v_x, v_y, \dot{\gamma}]_{\mathcal{B}}$ on uneven ground using proprioceptive and exteroceptive information. The state is represented as $s := \langle s_r, s_v, s_{j_p}, s_{j_v}, s_a, s_m, s_c \rangle$,

where $s \in \mathbb{R}^{259}$, $s_r^{\mathcal{B}} \in \mathbb{R}^3$ is the second row of the rotation matrix, $s_v \in \mathbb{R}^6$ is the base linear and angular velocities, $s_{j_p}^{\mathcal{B}} \in \mathbb{R}^{24}$ is the sparse history of joint position errors and $s_{j_v}^{\mathcal{B}} \in \mathbb{R}^{24}$ is the sparse history of joint velocities, $s_a \in \mathbb{R}^{12}$ is the previous action, $s_m \in \mathbb{R}^{187}$ are measurements from the height-map around the robot's base and $s_c^{\mathcal{B}} \in \mathbb{R}^3$ is the velocity command. The actions $a \in \mathbb{R}^{12}$ are interpreted as the reference joint positions $q_j^*$. The state $s$ is fed to an MLP network made by three layers respectively of size $[512, 256, 128]$ and the action $a$ is subsequently tracked by the low level PD controller ($K_p = 80.$, $K_d = 2.$).

## 6.2. Blind Quadrupedal Locomotion

The A1 quadruped robot is required to follow a velocity command $[v_x, v_y, \dot{\gamma}]_{\mathcal{B}}$ on flat ground using proprioceptive information. The state is represented as $s := \langle s_r, s_v, s_{j_p}, s_{j_v}, s_a, s_c \rangle$, where $s \in \mathbb{R}^{192}$, $s_r^{\mathcal{B}} \in \mathbb{R}^3$ is the second row of the rotation matrix, $s_v \in \mathbb{R}^6$ is the base linear and angular velocities, $s_{j_p}^{\mathcal{B}} \in \mathbb{R}^{84}$ is the history of joint position errors and $s_{j_v}^{\mathcal{B}} \in \mathbb{R}^{84}$ is the history of joint velocities, $s_a \in \mathbb{R}^{12}$ is the previous action, velocity and action and $s_c^{\mathcal{B}} \in \mathbb{R}^3$ is the velocity command. The actions $a \in \mathbb{R}^{12}$ are interpreted as the reference joint positions $q_j^*$. The state $s$ is fed to an MLP network formed by two layers respectively of size $[512, 512]$ and the action $a$ is tracked by the low level PD controller ($K_p = 15.$, $K_d = 1.$). The base linear velocity in $s_v$ is not provided by the onboard state estimator and it was estimated similarly to (Ji et al., 2022) through an MLP network of size $[128, 128]$.
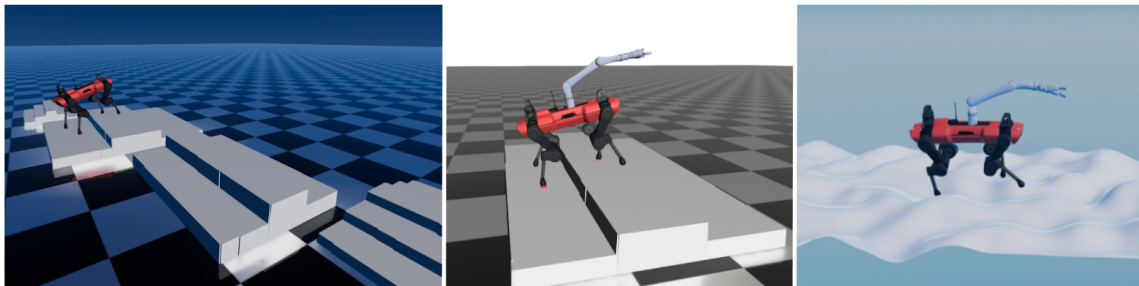


Figure 3: (Left) Examples of stairs with varying step-height and step-depth used for evaluation. (Center) ANYmal C walking on stairs with an unmodeled Kinova manipulator. (Right) ANYmal C walking on rocky terrain during tests.

## 7. Experimental Setup

To evaluate our method, we employed ANYmal C as a reference platform and we trained different policies for 10,000 iterations using IsaacGym (Makoviychuk et al., 2021a) each adopting one among RFI, RAO, ERFI-50, ERFI-C, and ActNetRand, where ActNetRand represents the present state-of-the-art approach implementing both actuation network and extensive domain randomization, as in (Rudin et al., 2021). The environment settings used are described in Section 6.1.

The performances of the policies trained with the different methods were assessed by addressing perceptive locomotion over stairs and rocky terrain as relevant case study (Figure 3). Moreover, to obtain more realistic results the experiments were conducted in a different simulator (RaiSim (Hwangbo et al., 2018b)) and we included an actuator network to reproduce the dynamics of SEAs (which was not used during the training of RFI/RAO/EFRI policies). The robot is always

deployed at the same position, the velocity command is fixed to $0.5\,\mathrm{m\,s^{-1}}$ and it has $8\,\mathrm{s}$ to go up the stairs; the attempt is considered a failure when the robot falls on the ground or when it is not able to move forward for at least $2.5\,\mathrm{m}$. We generated 50 random stairs as in (Gangapurwala et al., 2020), which are placed just in front of the robot, and walking for $2.5\,\mathrm{m}$ from the spawning point requires tackling at least one step.

To assess the robustness of the policies to unseen conditions we introduced perturbations to the simulation environment: the application of external forces $[0 ; 150]\,\mathrm{N}$ to the base (fixed value during training: 0.) for a duration of $3\,\mathrm{s}$, the application time of an external force of $50\,\mathrm{N}$ varies between $[0 ; 3]\,\mathrm{s}$ (fixed value during training: 0.), the application of external torque $[0 ; 75]\,\mathrm{N\,m}$ to the base (fixed value during training: 0.) for a duration of $1\,\mathrm{s}$, the friction coefficient between ground and feet in the range $[0.2, 0.8]$ (fixed value during training: 0.5), the gravitational acceleration was modified between $[-18 ; -2]\,\mathrm{m/s^2}$ (fixed value during training: $-9.81\,\mathrm{m/s^2}$), the position of the knees' motors was shifted by $[-0.15 ; -0.15]\,\mathrm{m}$ (fixed value during training: 0.) and the mass of the base changed between $[22 ; 65]\,\mathrm{kg}$ (fixed value during training: $27\,\mathrm{kg}$). Throughout the evaluation we alter only one parameter at the time and for each of them we run 50 experiments with different terrains. Furthermore, we replicate the set of experiments above with a robotic arm mounted on top of ANYmal C, this introduces significant variations in the mass matrix $M$ which the robot never explicitly experienced during training. Following the thorough validation presented in simulation, the best performing controller (resulted to be trained with ERFI-50) was deployed on the hardware, tested on rough and uneven terrain, both in the laboratory and outdoor environments to validate the feasibility of the method.

In addition, we demonstrate the effectiveness of ERFI-50 with hardware experiments also on Unitree A1, this time performing blind locomotion in challenging conditions. We present results of extensive hardware evaluation in Figure 5 and on our accompanying website https://sites. google.com/view/erfi-video.

## 8. Results

We compared ERFI-50, ERFI-C, and RAO against two baselines, RFI and ActNetRand (policy trained using dynamics randomization and actuator network). The metric adopted to assess their performances is the success rate described in Section 7. The first row of Figure 4 (Figures 4(a) to 4(c)) shows the robustness of the different approaches to changes in the base mass, in the application of external forces, or to different friction coefficients between feet and ground; in this first batch of experiments, the arm was not included. From these plots, it is evident that the standard RFI is the least performing method, while still providing decent robustness especially close to the training domain. Conversely, RAO and ERFI-50 are the better-performing ones (providing on average 53% better success rate than RFI on mass variations, Figure 4(a)), they are often very close and sometimes better than ActNetRand, which is currently the standard approach to deploy controllers on the hardware. Regarding ERFI-C, it does better than standard RFI (on average 41% better success rate on mass variations, Figure 4(a)), but still not as well as ERFI-50 and RAO (on average 12% worse success rate on mass variations, Figure 4(a)). The analysis presented above was repeated after mounting a fixed Kinova manipulator arm on top of the robot; the same policies, perturbation, and set of stairs were considered during the experiments. The objective of this last study is to test the robustness of the controller in real-world scenarios never encountered during training. The resulting performances are depicted in Figures 4(d) to 4(f), where we observe the gap between
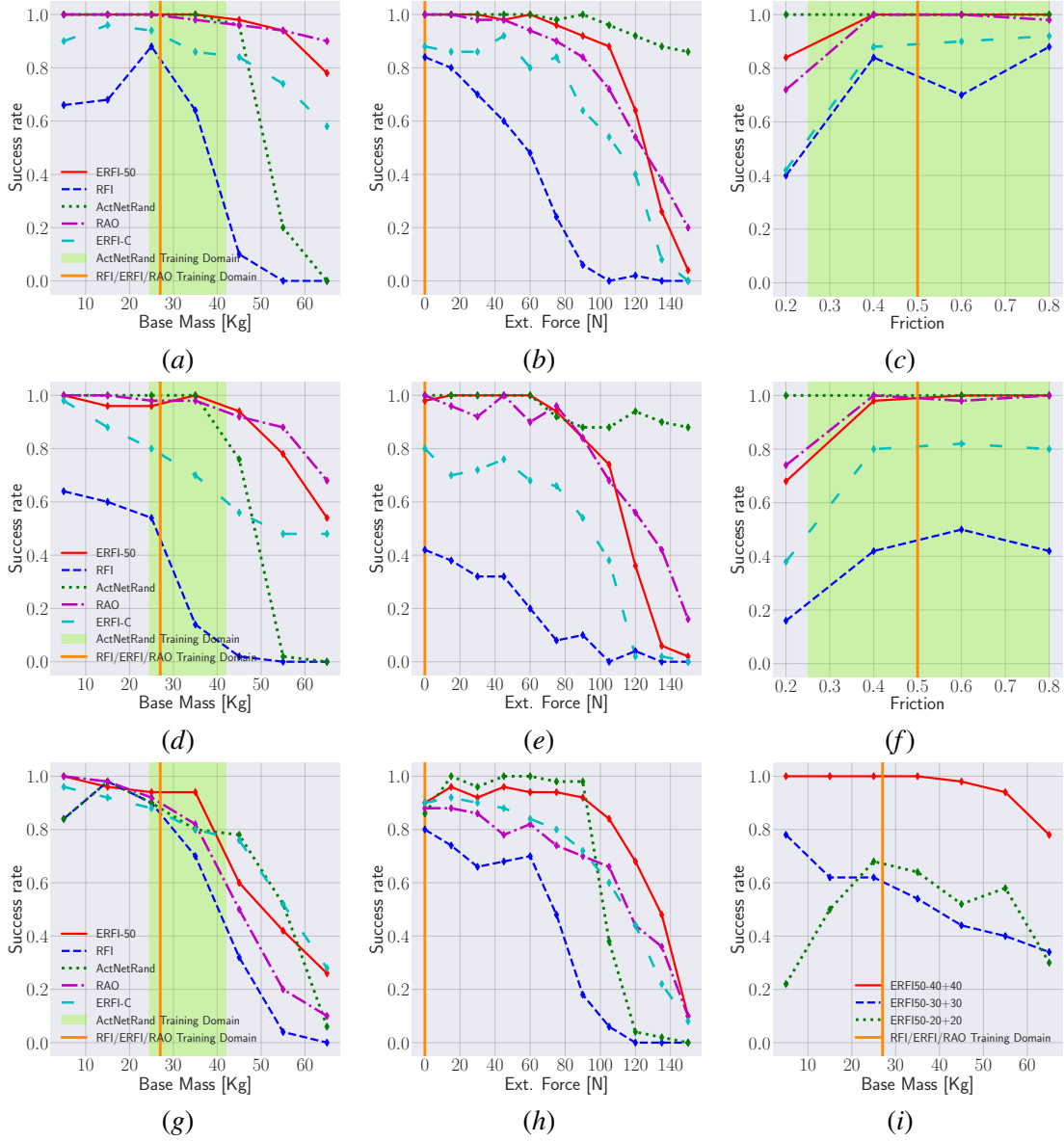
Figure 4: Figures 4(a) to 4(c) show how RFI, ERFI-50, ERFI-C, RAO and ActNetRand resist to variations of the base mass, to external forces, or to different frictions. In Figures 4(d) to 4(f) the same experiments are replicated with a Kinova manipular on top of the robot. In Figures 4(g) and 4(h) we investigated the effects of the perturbations also on the rocky terrain in Figure 3. While, in Figure 4(i) we studied how different $\tau_{o_j}^{lim}$ and $\tau_{r_j}^{lim}$ affected the robustness of the controller.

RFI and RAO/ERFI-50 enlarging with a performance loss for RFI -even in the training domain- of roughly 50%, while RAO and ERFI-50 achieved roughly 62% higher success rate than RFI on this task, Figure 4(d).

Furthermore, we investigated the effects of $\tau_{o_j}^{lim}$ and $\tau_{r_j}^{lim}$ on the overall performances of ERFI-50, we show the outcomes of different limits on the success rate when the base mass is increased, Figure 4(i). The curves in Figure 4(i) show that high $\tau_{o_j}^{lim}$ provides greater robustness in combination

with high $\tau_{r_j}^{lim}$ (ERFI50-40+40, red line), when compared to $\tau_{o_j}^{lim} = 20[\text{N m}]$ and $\tau_{r_j}^{lim} = 20[\text{N m}]$. However, for values of $\tau_{o_j}^{lim} = 30[\text{N m}]$ and $\tau_{r_j}^{lim} = 30[\text{N m}]$ the performance improves in one portion of the domain and it remains comparable to $\tau_{o_j}^{lim} = 20[\text{N m}]$ and $\tau_{r_j}^{lim} = 20[\text{N m}]$ in the remaining one.

To provide a comprehensive assessment, we present the performance on the rocky terrain in Figure 3, which complements the results obtained in the staircase environment. Notably, the survival rates in high perturbation regimes are partially reduced due to the absence of rocky terrain in the training environment. Nonetheless, the relative performances of the methods remain comparable to those observed in the staircase evaluation, Figures 4(*g*) and 4(*h*).

The robustness to further perturbations –as varying the duration of the external force, applying an external torque, varying the gravitational acceleration and shifting the knee motors' positions– were investigated and the results are consistent with what is already shown in Figure 4.
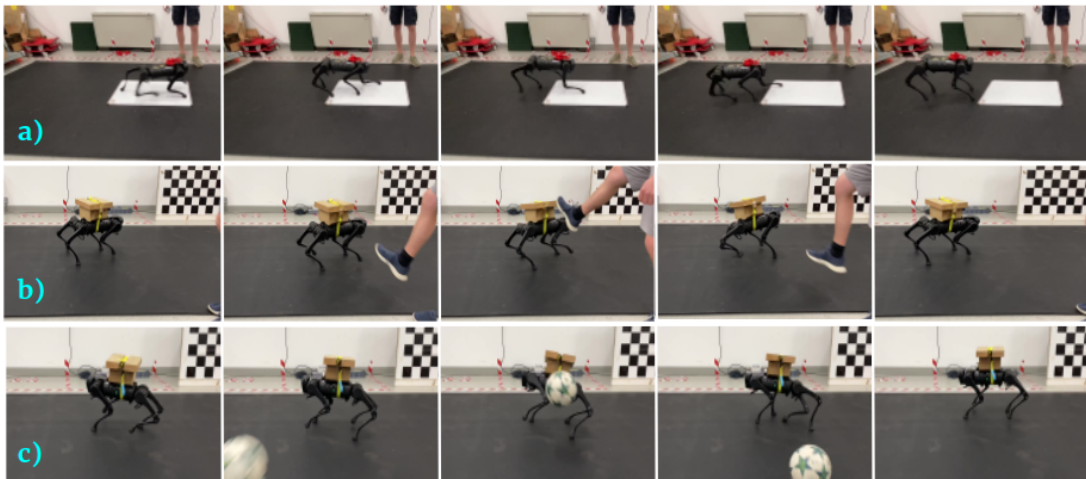


Figure 5: This figure shows some of the experiments on Unitree A1 adopting ERFI: a) Walking on wet terrain and recovering from slipping, b) resisting to kicks, and c) withstanding impulsive forces. More experiments and videos available on our accompanying website https://sites.google.com/view/erfi-video.

## 9. Conclusion

In this work we showed that transferring policies trained in simulation to real systems is possible without defining the domain randomization's parameters and their ranges, without further system identification to measure the noise to inject in the observations and without recording any of hardware data to train an additional NN to model the motors' dynamics. Instead we proposed to use a blend of episodic and continuously changing random force perturbations (ERFI-50), which has competitive performance compared to state of the art extensive domain randomization (ActNetRand) and which only requires tuning two parameters ($\tau_{o_j}^{lim}$ and $\tau_{r_j}^{lim}$); hence reducing the sim-to-real transfer efforts compared to previous approaches by a large margin. We further demonstrated the validity of our approach by transferring the controllers to the hardware and showing stable locomotion with Unitree A1, uneven terrain locomotion and mounting an unmodelled manipulator on top of the robot with ANYmal C.

# References

Steven Bohez, Saran Tunyasuvunakool, Philemon Brakel, Fereshteh Sadeghi, Leonard Hasen-clever, Yuval Tassa, Emilio Parisotto, Jan Humplik, Tuomas Haarnoja, Roland Hafner, Markus Wulfmeier, Michael Neunert, Ben Moran, Noah Siegel, Andrea Huber, Francesco Romano, Nathan Batchelor, Federico Casarini, Josh Merel, Raia Hadsell, and Nicolas Heess. Imitate and repurpose: Learning reusable robot movement skills from human and animal behaviors, 2022. URL https://arxiv.org/abs/2203.17138.

Luigi Campanaro, Daniele De Martini, Siddhant Gangapurwala, Wolfgang Merkt, and Ioannis Havoutis. Roll-drop: accounting for observation noise with a single parameter. In Nikolai Matni, Manfred Morari, and George J. Pappas, editors, *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, volume 211 of *Proceedings of Machine Learning Research*, pages 718–730. PMLR, 15–16 Jun 2023. URL https://proceedings.mlr.press/v211/campanaro23a.html.

Siddhant Gangapurwala, Mathieu Geisert, Romeo Orsolino, Maurice Fallon, and Ioannis Havoutis. RLOC: Terrain-Aware Legged Locomotion using Reinforcement Learning and Optimal Control. *arXiv e-prints*, art. arXiv:2012.03094, December 2020.

Christian Gehring, Stelian Coros, Marco Hutter, Carmine Dario Bellicoso, Huub Heijnen, Remo Diethelm, Michael Bloesch, Peter Fankhauser, Jemin Hwangbo, Mark Hoepflinger, and Roland Siegwart. Practice makes perfect: An optimization-based approach to controlling agile motions for a quadruped robot. *IEEE Robotics & Automation Magazine*, 23(1):34–43, 2016. doi: 10.1109/MRA.2015.2505910.

Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. Anymal - a highly mobile and dynamic quadrupedal robot. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 38–44, 2016. doi: 10.1109/IROS.2016.7758092.

Jemin Hwangbo, Joonho Lee, and Marco Hutter. Per-contact iteration method for solving contact dynamics. *IEEE Robotics and Automation Letters*, 3(2):895–902, 2018a. URL www.raisim.com.

Jemin Hwangbo, Joonho Lee, and Marco Hutter. Per-contact iteration method for solving contact dynamics. *IEEE Robotics and Automation Letters*, 3(2):895–902, 2018b. URL www.raisim.com.

Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *CoRR*, abs/1901.08652, 2019. URL http://arxiv.org/abs/1901.08652.

Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In Federico Morán, Alvaro Moreno, Juan Julián Merelo, and Pablo Chacón, editors, *Advances in Artificial Life*, pages 704–720, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. ISBN 978-3-540-49286-3.

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 Letters*, 7(2):4630–4637, 2022. doi: 10.1109/LRA.2022.3151396.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: Rapid motor adaptation for legged robots. In *Robotics: Science and Systems*, 2021.

Joonho Lee, Jemin Hwangbo, and Marco Hutter. Robust recovery controller for a quadrupedal robot using deep reinforcement learning. *CoRR*, abs/1901.07517, 2019. URL http://arxiv.org/abs/1901.07517.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *CoRR*, abs/2010.11251, 2020. URL https://arxiv.org/abs/2010.11251.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021a. URL https://arxiv.org/abs/2108.10470.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021b.

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *CoRR*, abs/2201.08117, 2022. URL https://arxiv.org/abs/2201.08117.

Xue Bin Peng and Michiel van de Panne. Learning locomotion skills using deeprl: Does the choice of action space matter? In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–13, 2017.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2018. doi: 10.1109/icra.2018.8460528. URL https://doi.org/10.1109%2Ficra.2018.8460528.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. *CoRR*, abs/2109.11978, 2021. URL https://arxiv.org/abs/2109.11978.

C Semini, N G Tsagarakis, E Guglielmino, M Focchi, F Cannella, and D G Caldwell. Design of hyq – a hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 225(6): 831–849, 2011. doi: 10.1177/0959651811402275. URL https://doi.org/10.1177/0959651811402275.

Sangok Seok, Albert Wang, Meng Yee Chuah, David Otten, Jeffrey Lang, and Sangbae Kim. Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot. In *2013

*IEEE International Conference on Robotics and Automation*, pages 3307–3312, 2013. doi: 10. 1109/ICRA.2013.6631038.

Yecheng Shao, Yongbin Jin, Xianwei Liu, Weiyan He, Hongtao Wang, and Wei Yang. Learning free gait transition for quadruped robots via phase-guided controller. *IEEE Robotics and Automation Letters*, 7(2):1230–1237, 2022. doi: 10.1109/LRA.2021.3136645.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *CoRR*, abs/1804.10332, 2018. URL http://arxiv.org/abs/1804.10332.

Joshua 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. *CoRR*, abs/1703.06907, 2017. URL http://arxiv.org/abs/1703.06907.

Eugene Valassakis, Zihan Ding, and Edward Johns. Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics. *CoRR*, abs/2008.06686, 2020. URL https://arxiv.org/abs/2008.06686.

Zhaoming Xie, Xingye Da, Michiel van de Panne, Buck Babich, and Animesh Garg. Dynamics randomization revisited:a case study for quadrupedal locomotion, 2020. URL https://arxiv.org/abs/2011.02404.

Chuanyu Yang, Kai Yuan, Qiuguo Zhu, Wanming Yu, and Zhibin Li. Multi-expert learning of adaptive legged locomotion. *Science Robotics*, 5(49):eabb2174, 2020.

Yuxiang Yang, Tingnan Zhang, Erwin Coumans, Jie Tan, and Byron Boots. Fast and efficient locomotion via learned gait transitions. 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 773–783. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/yang22d.html.