Unified Legged Locomotion: A Single Policy for Millions of Morphologies

Nico Bohlinger[1], Jan Peters[1,2]

***Abstract*— We present a single, general locomotion policy trained on a diverse collection of 50 legged robots. By combining an improved embodiment-aware architecture (URMAv2) with a performance-based curriculum for extreme Embodiment Randomization, our policy learns to control millions of morphological variations. Our policy achieves zero-shot transfer to unseen real-world humanoid and quadruped robots.**

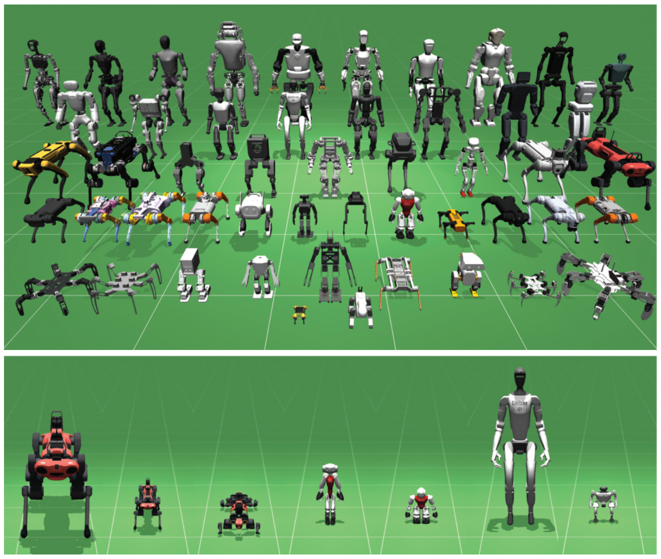
Fig. 1. We collected a diverse set of 50 legged robots. We train the policy on all robots simultaneously using 25600 parallel environments. The performance-based curriculum on extreme Embodiment Randomization leads to the policy seeing gradually more challenging embodiments throughout training.

I. INTRODUCTION

With more and more robot platforms being developed and finding their way into research labs and real-world applications, the current paradigm of training a control policy tailored to a specific robot can become increasingly inefficient [1], [2]. Robot platforms change, adapt, and evolve over time, but many current training approaches do not consider robot morphologies as a key factor. Their learning process is agnostic or simply unaware of the specific characteristics and capabilities of the robot's embodiment, making cross-embodiment transfer difficult or even impossible. We build upon the recently introduced Unified Robot Morphology Architecture (URMA) [3] and train a single unified embodiment-aware policy across 50 different legged robots with massive Embodiment Randomization (ER). This results in a curriculum of up to 10 million embodiments per training run, in order to learn a robust and adaptive general locomotion policy, that can be directly zero-shot transferred to unseen humanoid and quadruped robots in the real world.

[1]Technical University of Darmstadt. [2]hessian.AI. & German Research Center for AI (DFKI). & Robotics Institute Germany (RIG).

II. METHOD

We build upon the original URMA and scale it to the massively multi-embodiment setting (see Figure 2), which we call Unified Robot Morphology Architecture v2 (URMAv2).

A. URMAv2 Architecture

Inputs: Following the original URMA architecture, the inputs are split into three categories: per-joint description vectors $\{d_j\}_{j \in \mathcal{J}}$ for the set $\mathcal{J}$ of all joints in a given robot that uniquely describe a joint's static properties (e.g., rotation axis, torque limits), per-joint observations $\{o_j\}_{j \in \mathcal{J}}$ containing dynamic state information (e.g., position, velocity), and general robot observations $\mathbf{o}_g$ (e.g., trunk velocity, gravity vector). URMAv2 includes an additional per-joint observation to allow for task-specific joint-level conditioning.

Joint Encoder: URMAv2 keeps the same attention-based joint encoder, which processes each joint's description (attention keys) and observation vector (attention values), and aggregates them into the combined joint latent vector

$$\bar{z}_{\text{joints}} = \sum_{j \in \mathcal{J}} f_\psi(o_j) \frac{\exp(f_\phi(d_j)/\tau)}{\sum_{L_d} \exp(f_\phi(d_j)/\tau)}. \quad (1)$$

where f_ϕ (with latent dimension L_d) and f_ψ are the encoders for the joint descriptions and joint observations, respectively, and τ is the learnable temperature parameter. URMAv2 uses a wider Multilayer Perceptron (MLP) for f_ψ (2x 256 units) for the policy network to increase its capacity for the larger number of robots and variations.

Core Network: The joint latent vector is concatenated with the fixed-size general observations and processed by the core network h_θ to generate the action latent vector: $\bar{z}_{\text{action}} = h_\theta(o_g, \bar{z}_{\text{joints}})$. URMAv2 uses a deeper stack of 5 hidden layers to increase the model capacity. For stabilization, WeightNorm layers are used around every Dense layer.

Action Decoder: We replace URMA's universal decoder with an attention-based decoding mechanism. URMAv2 computes the mean action μ_j for each joint via a simple dot product between the action latent vector $\bar{\mathbf{z}}_{\text{action}}$ and the corresponding joint's attention weights α_j that were generated in the encoder: $\mu_j = \bar{\mathbf{z}}_{\text{action}} \cdot \alpha_j$ Also, the per-joint standard deviations are predicted with a linear layer σ_υ from the same joint description encoding calculated for the attention weights, which results in actions being

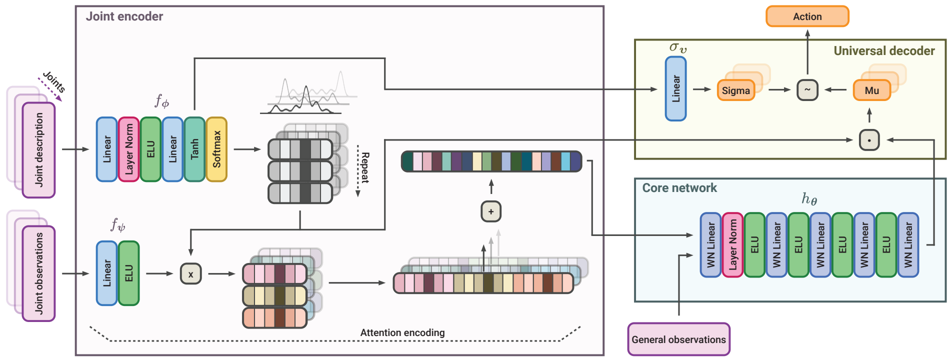

Fig. 2. Overview of the neural network architecture for a URMAv2 policy.

sampled from: $a_j \sim \mathcal{N}(\mu_j, \sigma_v(f_\phi(d_j)))$. This creates a streamlined architecture that is simpler and computationally more efficient than the original URMA decoder.

B. Embodiment Randomization

To improve the generalization across different embodiments, we apply extreme ER online during training (see Figure 1). ER differs from standard Domain Randomization (DR) [4] in that all the generated values are seen by the policy through the description vectors. We use DR after the ER sampling to further modify the sampled parameters but keep them hidden from the policy, to improve sim-to-real transfer. Our ER includes scaling of: body part size and position in every dimension, masses and inertias, joint orientations, IMU position, motor torques and velocities, position limits, PD gains, etc. We sample a new embodiment during every episode step with a probability of 0.2% at the highest curriculum level. This leads to up to 10 million different embodiments per training run.

C. Performance-based Curriculum

We introduce a performance-based curriculum learning strategy that attaches every component of the learning framework to a single curriculum coefficient $\beta \in [0, 1]$. This coefficient is initialized to $\beta = 0$ and is increased whenever an episode is deemed successful, e.g., a return threshold is reached. This significantly helps to speed up the training process for challenging embodiments, especially humanoids, and leads to more stable training runs.

III. EXPERIMENTS

We train URMAv2 on a set of 50 legged robots, including 15 quadrupeds, 23 humanoids, 8 bipeds and 4 hexapods (see Figure 1). We use MuJoCo XLA (MJX) [5] and Proximal Policy Optimization (PPO) [6] which we implement with the RL-X library [7]. With a total of 25600 parallel environments we collect 1.6 million samples for every policy update. We train for a total of 5 billion environment steps.

Figure 3 shows the training performance of URMAv2 and baselines. We measure the performance by the average curriculum coefficient β over all robots. URMAv2 outperforms all baselines in terms of learning speed and final performance, showing the effectiveness of the embodiment-aware architecture in general.

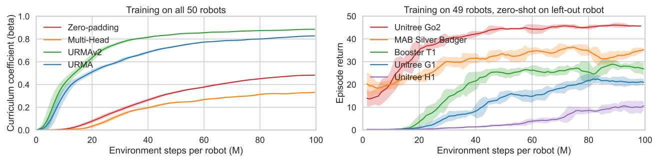

Fig. 3. Comparison of the training performance of URMA, URMAv2, zero-padding and multi-head baselines when training on all 50 robots (left). Different setups of URMAv2 trained on 49 robots and zero-shot performance evaluated on the left-out robot (right).

Figure 3 shows the zero-shot transfer performance of URMAv2 trained on 49 robots and evaluated on the left-out robot. URMAv2 shows strong zero-shot performance on especially the quadruped robots. Zero-shot performance for the humanoids is clearly lower. In simulation, the policy is able to control all three humanoids fairly well, but under severe perturbations it still falls occasionally.

A. Sim-to-Real Transfer

We did zero-shot sim-to-real transfer of URMAv2 trained on 49 robots to the Unitree Go2, MAB Silver Badger and the Booster T1. The policy is able to control both quadrupeds well, even under severe disturbances like pulling on the legs. The policy is able to walk forward and sidewards reliably on the Booster T1, but struggles with turning and walking backwards, leading to regular falls. We could not zero-shot transfer to the Unitree H1 as the policy was not stable enough, but we transfered to URMAv2 policy trained on all 50 robots, which was able to locomote well on the H1.

IV. CONCLUSION

We presented URMAv2, an improved embodiment-aware architecture for learning a general locomotion policy across a diverse set of 50 legged robots with extreme ER and a performance-based curriculum. While URMAv2 shows strong training performance and zero-shot transfer to unseen quadruped and humanoid robots in simulation, sim-to-real transfer to unseen humanoids still remains challenging.

ACKNOWLEDGMENT

This research is funded by the NSC Poland (UMO-2021/43/I/ST6/02711), DFG (PE 2315/17-1), hessian.AI Service Center, and the EuroHPC Joint Undertaking.

References

[1] B. Ai, L. Dai, N. Bohlinger, D. Li, T. Mu, Z. Wu, K. Fay, H. I. Christensen, J. Peters, and H. Su, "Towards embodiment scaling laws in robot locomotion," *Conference on Robot Learning (CoRL)*, 2025.

[2] N. Bohlinger, J. Kinzel, D. Palenicek, L. Antczak, and J. Peters, "Gait in eight: Efficient on-robot learning for omnidirectional quadruped locomotion," *International Conference on Intelligent Robots and Systems*, 2025.

[3] N. Bohlinger, G. Czechmanowski, M. Krupka, P. Kicki, K. Walas, J. Peters, and D. Tateo, "One policy to run them all: an end-to-end learning approach to multi-embodiment locomotion," *Conference on Robot Learning*, 2024.

[4] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *International conference on intelligent robots and systems*, 2017.

[5] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

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

[7] N. Bohlinger and K. Dorer, "Rl-x: A deep reinforcement learning library (not only) for robocup," in *Robot world cup*. Springer, 2023, pp. 228–239.