Mechanical Intelligence-Aware Curriculum Reinforcement Learning for Humanoids with Parallel Actuation

Yusuke Tanaka[1*], Alvin Zhu[2,3*], Quanyou Wang[1], Dennis Hong[1]

***Abstract*— Reinforcement learning (RL) has enabled significant advances in humanoid robot locomotion, yet most learning frameworks do not account for mechanical intelligence embedded in parallel actuation mechanisms due to limitations in simulator support for closed kinematic chains. This omission can lead to inaccurate motion modeling and suboptimal policies, particularly for robots with high actuation complexity. This paper presents an end-to-end curriculum RL framework for BRUCE, a kid-sized humanoid robot featuring three distinct parallel mechanisms in its legs: a differential pulley, a 5-bar linkage, and a 4-bar linkage. Unlike prior approaches that rely on simplified serial approximations, we simulate all closed-chain constraints natively using GPU-accelerated MJX (MuJoCo), preserving the hardware's physical properties during training. We benchmark our RL approach against a Model Predictive Controller (MPC), demonstrating better surface generalization and performance in real-world zero-shot deployment. This work highlights the computational approaches and performance benefits of fully simulating parallel mechanisms in end-to-end learning pipelines for legged humanoids.**

I. Introduction

Humanoid robots have achieved significant advancements in mobility and manipulation through various control strategies, particularly Model Predictive Control (MPC) [1], [2] and Reinforcement Learning (RL) [3], [4]. However, purely learning-based approaches have yet to leverage the mechanical intelligence embedded in robot designs fully [5]. In particular, parallel mechanisms have been shown to offer advantages such as higher combined motor output power, reduced inertia, and greater transmission ratio [6], [7]. Despite benefits, the parallel mechanisms are less commonly modeled in robotics physics simulations due to the challenges of handling closed kinematic chains, where a single child body is connected to more than one parent body [3].

Existing simulation frameworks often sidestep the complexity of closed-chain systems by simplifying or approximating the parallel linkages as serial chains [8]. Such assumptions undermine the natural mechanical intelligence inherent in the structure, requiring a lower-level controller that handles the actual parallel mechanisms [5]. Consequently, controllers trained or designed under these simplified conditions cannot fully exploit the system's real-world mechanical properties. Furthermore, these approximations can lead to inaccurate and suboptimal motion [5] in the simulation, as well as an inability to simulate singularities, which is a notable issue in parallel mechanisms [9].

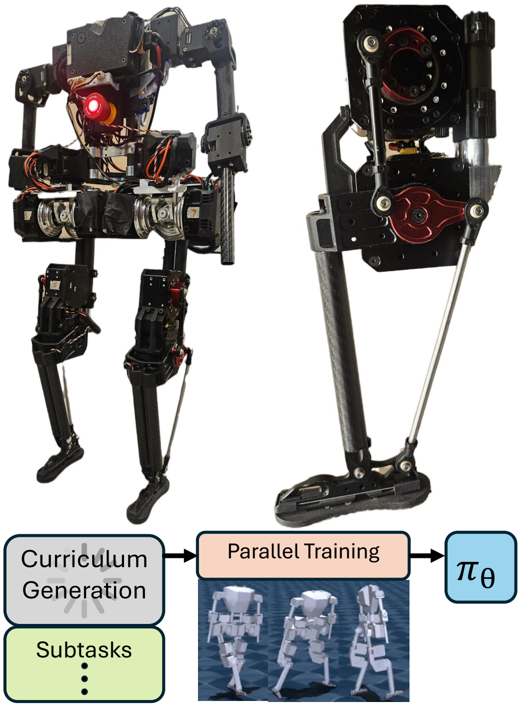


Fig. 1: BRUCE [2] hardware with three distinct parallel mechanisms, which are simulated, and an RL policy is deployed on the hardware zero-shot.

This paper presents an end-to-end policy trained through curriculum RL for a kid-size humanoid robot, BRUCE [2], which features three distinct parallel mechanisms in each leg. We incorporate constraint-based modeling of closed-chain constraints in our simulation environment, ensuring that the learned control policies capture the mechanical intelligence that parallel linkages offer. This provides a pure end-to-end control policy, where all action spaces directly correspond to the physical hardware actuators. We further compare the performance of these learned policies to that of the MPC controller, examining the benefits and costs of parallel linkage modeling in simulation and hardware deployments.

The main contributions of this work are:

- Simulating three distinct parallel mechanisms in the simulation to account for closed kinematic chain constraints and analysis.
- A curriculum RL trained fully end-to-end locomotion policy on a humanoid robot, BRUCE.
- A hardware validation and performance comparison

[1] Department of Mechanical and Aerospace Engineering, UCLA, Los Angeles, CA, USA.
[2] Department of Computer Science, UCLA, Los Angeles, CA, USA.
[3] Electrical and Computer Engineering Department, UCLA, Los Angeles, CA, USA.
Emails: {yusuketanaka, alvister88, richardlin, mehtank, dennishong}@g.ucla.edu

*Equal contribution.

against a baseline MPC.

By integrating parallel mechanism modeling directly into the RL training process and evaluating the results in hardware, this study highlights the importance and feasibility of capturing the full mechanical dynamics of complex humanoid robots, providing a way to embrace mechanical intelligence in machine learning.

II. Related Work

A. Parallel Mechanism Leg and Humanoid

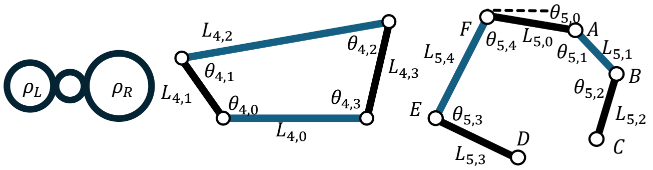
Parallel linkage mechanisms have been widely adopted in legged robotics thanks to their mechanical advantages, including transmission and structural benefits, combined actuator outputs, etc. [6]. These advantages have been leveraged in agile dynamic [6], [10], [11] and power-intensive domains [9], [12], enabling novel locomotion performance.

In humanoid robot leg designs, parallel linkages reduce leg inertia and aggregate mass closer to the torso, approaching the idealized single-mass spring-loaded inverted pendulum model [7], [13]. Various research humanoid legged robots have employed hybrid serial-parallel linkages [14] to realize higher DoF in the leg [15], [16] or through tendon-driven mechanisms [17]. Commercially developed humanoids (e.g., Unitree H1, Fourier GR1, and Optimus [5]) also utilize parallel linkages, yet simulating such closed-chain mechanisms remains challenging.

Those parallel mechanism-based humanoid platforms have been successful using both model-based [1] and RL [3] methods. However, simulating the closed kinematics chain has been challenging [8], [18]. The GPU-accelerated Isaac Gym RL framework lacks native support for closed-chain kinematics, requiring custom implementations or approximations [3]. Researchers have resorted to strategies such as adopting a kinematic rather than actuator joint space or approximating the parallel mechanism as a serial chain [5]. Although these approaches enable simulation, they can yield suboptimal results and limit direct use of mechanical advantages. Efforts to accurately simulate parallel linkages include extending URDF with graph-based pre-processing and constraint embedding [8] or treating closed chains as contact constraints with full differentiation for optimal control [5].

B. Reinforcement Learning on Humanoid

RL has emerged as an effective framework for robots to acquire complex control policies directly through interaction, without the need for explicit system modeling. Deep RL algorithms—such as Proximal Policy Optimization (PPO) [19], Trust Region Policy Optimization (TRPO) [20], and Deep Deterministic Policy Gradient (DDPG) [21]—have demonstrated impressive performance on high-dimensional locomotion tasks in both bipedal and quadrupedal robots [22]. These algorithms enable agents to learn robust and adaptive behaviors by optimizing policies over time through experience, often in the presence of non-linear dynamics, contact-rich environments, and unstructured terrain [23].



(a) Differential Gear (b) 4-bar Linkage (c) 5-bar Linkage

Fig. 2: general topology definitions of three parallel mechanisms in BRUCE. The five-bar linkage forms a closed chain where Points C and D are at the same location.

However, the application of RL to robotics presents significant challenges [24]. These methods typically require large-scale data collection and finely tuned reward functions to ensure stable learning [25]. In humanoid robotics, this is especially difficult due to the system's high degrees of freedom, underactuation, and sensitivity to small control errors [26]. To bridge the gap between simulation and the real world, sim-to-real transfer has become an essential focus [27], [28]. Techniques such as domain randomization, dynamics perturbation, and injecting latency into the environment are commonly employed to expose policies to a wide range of conditions during training, improving their robustness to real-world discrepancies [29]. Recent advancements also explore zero-shot policy transfer, where agents trained entirely in simulation can generalize to real-world deployment through domain randomization [30] without additional fine-tuning [31], [32]. Curriculum learning is also often used to improve convergence by structuring training from simple to more complex tasks, helping guide policy development progressively and stably [33]–[35].

In this work, we present a GPU-accelerated MuJoCo-based simulation (MJX) of three parallel mechanisms within BRUCE [2], utilizing soft equality constraints to preserve the intrinsic advantages of parallel hardware. We then demonstrate the trained locomotion policies through the curriculum RL directly on these fully simulated parallel chains.

III. Parallel Mechanism and Closed Kinematic Chain

This section covers the general mathematical formulation of the three closed kinematic chain mechanism constraints. BRUCE [2] is an agile and open-source small-sized humanoid platform. BRUCE includes three distinct parallel linkages, making it challenging to simulate the mechanisms entirely. The open-sourced simulation uses the kinematic joint space.

A. Hip Differential Drive Gear Mechanism

BRUCE's hip joint employs a cable-driven differential pulley system [2], which forms a 2-DoF parallel mechanism with minimal backlash. This differential mechanism imposes equality constraints between the actuated input and passive output joints, which can be mathematically expressed as:

$$\begin{bmatrix} \dot{q}_{\text{hip, roll}} \\ \dot{q}_{\text{hip, pitch}} \end{bmatrix} = \frac{1}{\rho_L + \rho_R} \begin{bmatrix} \rho_L & \rho_R \\ \rho_L & -\rho_R \end{bmatrix} \begin{bmatrix} \dot{q}_L \\ \dot{q}_R \end{bmatrix} \tag{1}$$

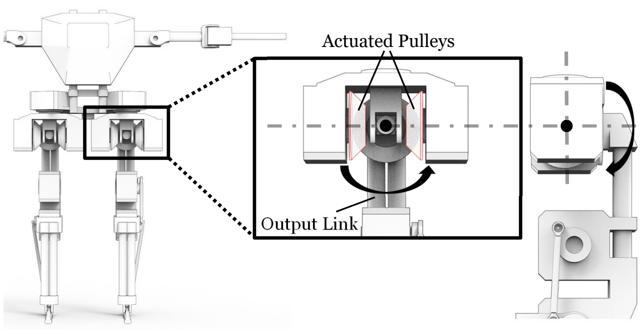


Fig. 3: The BRUCE's cable-driven differential pulley in close up.

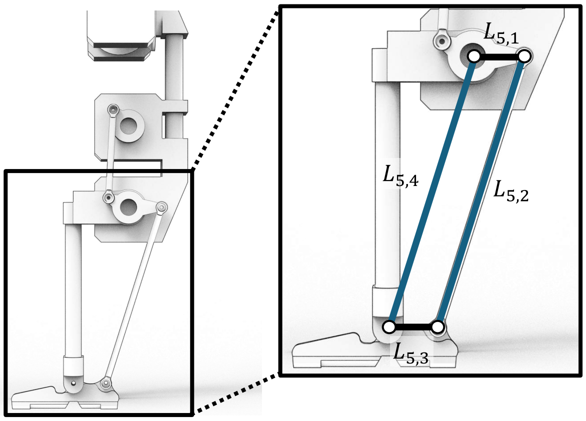


Fig. 4: The BRUCE's 5-bar parallelogram linkage in close up. $L_{5,1}$ and $L_{5,4}$ are actuated by two motors. $L_{5,0}$ in Fig. 2 is zero.

Here, $\dot{q}_{\text{hip, roll}}$ and $\dot{q}_{\text{hip, pitch}}$ denote the roll and pitch joint velocities, respectively. $\dot{q}_{L,R}$ is the actuator velocities, and $\rho_{L,R}$ represents the gear ratios between the left/right drive bevel gears and the central idler gear.

B. Leg 5-Bar Linkage Mechanism

The kinematic simulation of the 5-bar linkage is considered as two serial chains, endpoint constrained at the ends, forming a closed loop. The first endpoint C is given by:

$$\mathbf{p}_C = \mathbf{R}(\theta_{5,1}) \begin{bmatrix} l_{5,1} & 0 \end{bmatrix}^\top + \mathbf{R}(\theta_{5,2}) \begin{bmatrix} l_{5,2} & 0 \end{bmatrix}^\top \tag{2}$$

Similarly, the endpoint D of the second chain is:

$$\mathbf{p}_D = \mathbf{R}(\theta_{5,4}) \begin{bmatrix} l_{5,4} & 0 \end{bmatrix}^\top + \mathbf{R}(\theta_{5,3}) \begin{bmatrix} l_{5,3} & 0 \end{bmatrix}^\top \tag{3}$$

Note that the rotation matrix is $\mathbf{R}(\theta) \in \mathrm{SO}(2)$. These points are projected into 3D space:

$$\tilde{\mathbf{p}}_C = {}^{\mathrm{b}}T_A \begin{bmatrix} \mathbf{p}_C & 0 & 1 \end{bmatrix}^\top, \quad \tilde{\mathbf{p}}_D = {}^{\mathrm{b}}T_F \begin{bmatrix} \mathbf{p}_D & 0 & 1 \end{bmatrix}^\top \tag{4}$$

Where the transformation matrices ${}^{\mathrm{b}}T_A$ and ${}^{\mathrm{b}}T_F$ represents the 3D pose of points A and F relative to the base frame. The closed-chain constraint is then enforced as:

$$\tilde{\mathbf{p}}_C \approx \tilde{\mathbf{p}}_D. \tag{5}$$

$\theta_{5,1}$ $\theta_{5,4}$ are the actuated, while $\theta_{5,2}$ and $\theta_{5,3}$ are passive.

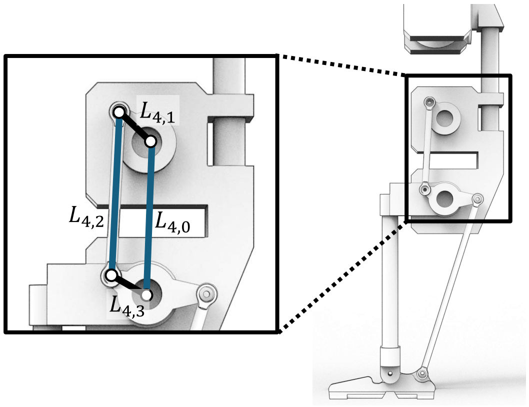


Fig. 5: The BRUCE's 4-bar linkage in close up. $L_{4,1}$ is actuated.

1) Handling Multiple Solutions: The 5-bar linkage can have five possible kinematic solutions given an endpoint position. Hence, starting the linkages with feasible and desirable configurations is necessary. If the singularity of the mechanisms is a concern in the simulation, a sanity check during the simulation is recommended.

C. 4-Bar Linkage Mechanism

The 4-bar linkage is a special case compared to 5-bar mechanisms or differential gears, as it has only one actuator and one output DoF. Thus, it can be viewed similarly to a gear with a nonlinear transmission ratio, which can be approximated using a polynomial function. However, unlike gears, the 4-bar linkage may encounter singularities (e.g., when all the links are collinear), which cannot be accurately captured by polynomial approximation. Therefore, we propose two approaches for modeling the 4-bar linkage.

1) Polynomial Transmission Ratio Approximation: The 4-bar linkage acts as a mechanism that alters the center of rotation, with a transmission ratio determined by the lengths.

$$\rho_4 = \frac{\dot{\theta}_{4,0}}{\dot{\theta}_{4,3}} = \frac{L_{4,2}\sin(\theta_{4,1} - \theta_{4,2})}{L_{4,2}\sin(\theta_{4,1} - \theta_{4,2}) - L_{4,3}\sin(\theta_{4,1} - \theta_{4,3})} \tag{6}$$

Here, ρ_4 denotes the velocity transmission ratio between the input angle θ_0 and the output angle θ_3. The torque transmission ratio is the reciprocal of ρ. In the special case where the 4-bar forms a parallelogram, this ratio becomes constant: $\rho = l_1/l_3$.

To approximate the nonlinear mapping between input and output angles, we use a polynomial constraint of the form:

$$\mathbf{r}_f = y - y_0 - \mathbf{a}^T \boldsymbol{\phi}(x - x_0) = 0 \tag{7}$$

$$\mathbf{a} = \begin{bmatrix} a_0 & \cdots & a_i \end{bmatrix}^T, i \in \mathbb{N} \tag{8}$$

$$\boldsymbol{\phi}(x - x_0) = \begin{bmatrix} (x - x_0)^0 & \cdots & (x - x_0)^i \end{bmatrix}^T \tag{9}$$

The coefficient vector $\mathbf{a}$ can be obtained via least squares regression to fit the observed mapping between $x = \theta_{4,0}$ and $y = \theta_{4,3}$ over a feasible range of motion.

as discussed in Section III-C.2.

2) Closed loop kinematic link constrained representation: While Section. III-C.1 method can capture the position and torque relationship of the 4-bar linkage, it does not model singularities—particularly when all four links become collinear. This limitation can be mitigated if the mechanism physically prevents reaching such configurations via joint limits. Otherwise, the constraints are similar to the 5-bar linkage in Section. III-B is necessary.

D. Backlash Due to Passive Joints

One mechanical drawback of parallel mechanisms is the introduction of unintended compliance, particularly at passive joints that actuated ones indirectly drive. Such backlash can be captured in a simulation when the full kinematics of the parallel mechanism are explicitly modeled.

In MuJoCo [36], all equality constraints are treated as soft and enforced through a virtual spring-damper system modulated by a constraint impedance function:

$$a_{c1} + \mathcal{D}(r) \cdot (b_v v + k_v r) = (1 - \mathcal{D}(r)) \cdot a_{c0} \tag{10}$$

Here, a_{c1} is the constrained acceleration due to the applied constraint force, and a_{c0} is the unconstrained acceleration. The function $\mathcal{D}(r)$ is a smooth, symmetric polynomial sigmoid that modulates the constraint impedance as a function of the constraint violation r. The coefficients b_v and k_v represent damping and stiffness, respectively, and v is the relative velocity at the constraint.

To model backlash, $\mathcal{D}(r)$ is designed to remain low within a deadband region (i.e., $|r| < \epsilon_q$), allowing low impedance motion. As $|r|$ exceeds this threshold, $\mathcal{D}(r)$ increases smoothly toward its maximum, gradually enforcing the constraint. The sigmoid is governed by hyperparameters defining the inflection point and steepness of the sigmoid.

IV. Reinforcement Learning

A. RL Formulation

We model the humanoid locomotion task as a Markov Decision Process [37] defined by the tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ the action space, $P(s_{t+1} \mid s_t, a_t)$ the state transition probability, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ the reward function, and $\gamma \in (0, 1]$ the discount factor.

B. Action Space

In this paper, we explore three different action spaces: positional, residual position command, and residual position command with respect to the previous reference. Note that as discussed in Section III, all of our actuated joints in the simulation are the same as the hardware actuator joints. **Action Space:** The action $a_t \in \mathbb{R}^{16}$ at time t is represented by a vector, which encodes the positional commands for each of the actuator joints of the humanoid robot. These commands are interpreted as offsets relative to a nominal home configuration $q_{\text{nom}} \in \mathbb{R}^{16}$, such that the reference joint angles are given by $q_t = q_{\text{nom}} + a_t$.

1) Observation Space: To capture temporal dependencies and mitigate sensor noise and latency, we maintain a history of H observations. The aggregated vector, $\mathbf{O}_t$, is defined as:

$$\mathbf{O}_t = \begin{bmatrix} \mathbf{o}_t & \mathbf{o}_{t-1} & \cdots & \mathbf{o}_{t-(H-1)} \end{bmatrix}^T, \quad \mathbf{O}_t \in \mathbb{R}^{N_o \cdot H} \tag{11}$$

$$\mathbf{o}_t = \begin{bmatrix} \dot{\psi}_{\text{IMU},t} & \mathbf{g}_{\text{proj},t} & \mathcal{C}_t & \mathbf{q}_t - \mathbf{q}_{\text{nom}} & \mathbf{a}_{t-1} \end{bmatrix} \tag{12}$$

$\mathbf{o}_t$ contains the observation at timestep t, such as sensor readings (11). $\dot{\psi}_{\text{IMU}}$ is the yaw rate measured by IMU, $\mathbf{g}_{\text{proj}}$ is a projected gravity vector representing the robot base frame tilt with respect to the world gravity vector, and $\mathcal{C}_t = (c_x, c_y, c_{\omega_\psi})$ are the user velocity commands in x, y and yaw. $\mathbf{q_t} - \mathbf{q}_{\text{nom}}$ is the difference between the current measured joint angle and the nominal positions, and $\mathbf{a}_{t-1}$ is the action from the previous timestep.

On hardware, the projected gravity vector in (13) is computed from the rotation matrix estimated by the Madgwick filter [38] using IMU gyro and acceleration data. The Madgwick filter provided a more stable estimation for our IMU sensor than the complementary filter.

$$\mathbf{g}_{\text{proj}} = \mathcal{R}_{\text{IMU}}{}^{-1}(0, 0, -1) \tag{13}$$

2) Policy and Objective: We parameterize the agent's policy by θ, with the policy represented as a function: $\pi_\theta : \mathbb{R}^{N_o \cdot H} \to \mathcal{P}(\mathcal{A})$ which maps the aggregated observation $\mathbf{O}_t$ to a probability distribution over actions. The objective is to maximize the expected cumulative reward:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\sum_{t=0}^{T} \gamma^t \, r(s_t, a_t) \right]. \tag{14}$$

3) Training via Proximal Policy Optimization (PPO): We employ PPO to update the policy in a stable manner. Let

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

denote the probability ratio and A_t the advantage estimate at time t. The PPO objective is given by:

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) A_t, \ \text{clip} \left(r_t(\theta), 1 - \epsilon, 1 + \epsilon \right) A_t \right) \right],$$

where ϵ is a hyperparameter that constrains the policy update. **Environment Dynamics and Filtering:** The environment is implemented using the GPU-accelerated physics engine MuJoCo-MJX [36] that simulates the robot dynamics with domain randomization, sensor noise, and latency effects. Specifically, the state transition is given by: A second-order Butterworth lowpass filter and a deadband filter are applied to control signals and observations. These filters ensure that the control commands and sensor readings are smoothed and simulate the control latencies experienced on hardware.

C. Reward Design

The reward function comprises several components designed to encourage stable, efficient, and task-oriented locomotion. *Tracking rewards* guide the agent to follow commanded linear and angular velocities, while penalizing vertical motion and rotational instability. *Postural rewards*, such as those on torso orientation and angular velocity, promote

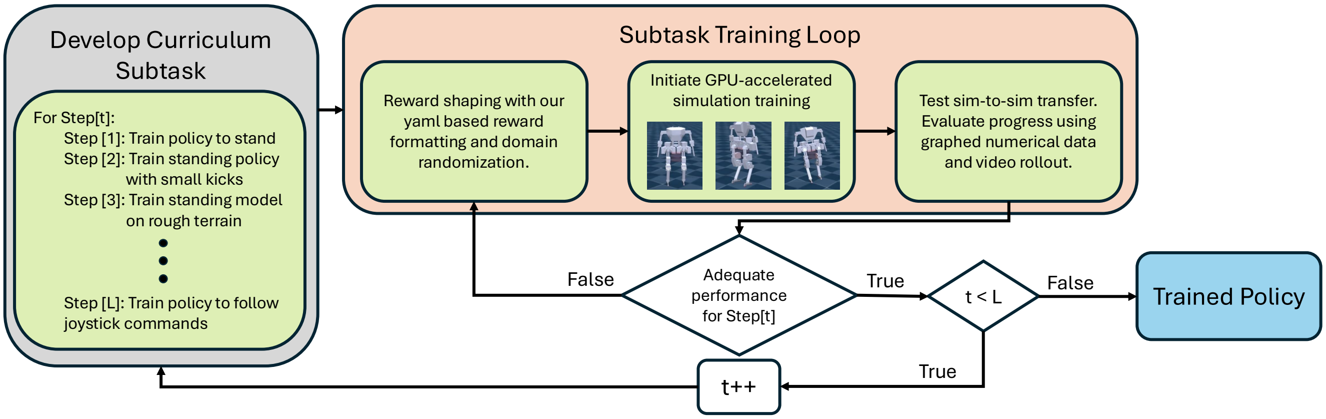

Develop Curriculum Subtask
For Step[t]:
Step [1]: Train policy to stand
Step [2]: Train standing policy with small kicks
Step [3]: Train standing model on rough terrain
Step [L]: Train policy to follow joystick commands
Subtask Training Loop
Reward shaping with our yaml based reward formatting and domain randomization.
Initiate GPU-accelerated simulation training
Test sim-to-sim transfer. Evaluate progress using graphed numerical data and video rollout.
False
Adequate performance for Step[t]
True
t < L
False
Trained Policy
True
t++

Fig. 6: Curriculum reinforcement learning framework overview.

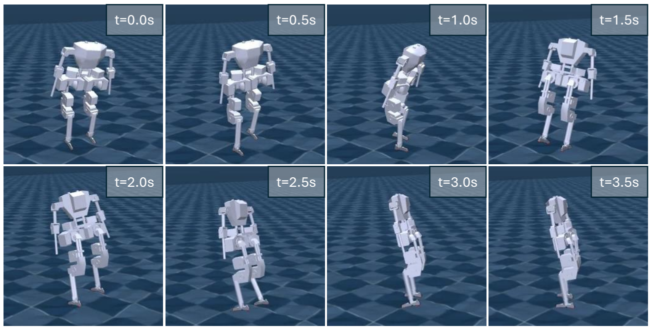


Fig. 7: BRUCE RL Policy in MuJoCo sim-to-sim transfer.

TABLE I: Parameter Specifications for Final-Stage Balancing

Domain Randomization	
Terrain	Height map with height variation in [0.0, 0.02] m.
Mass Variation	Up to ±20% of nominal mass.
Actuator Kp Offset	Proportional gains in $[-20, 20]$.
Foot Contact Offset	X-axis: ±0.006 m; Y/Z-axes: ±0.003 m.
COM Position Offset	±0.015 m translation in xyz.
Disturbances	
Observation Noise	±0.03 IMU and joint observations.
Kick Interval	Every 50 steps.
Kick Velocity	[0.2, 0.45] m/s.
Small Kick Interval	Every 10 steps.
Small Kick Velocity	[0.05, 0.1] m/s.
IMU Displacement	±0.006 m translational offset.
IMU Tilt	±0.06 rad angular offset.
Latency	
Observation Latency	$\mathcal{N}(0, (10\,\text{ms})^2)$, where 10 ms = 0.5 × 20 ms (environment step time).
Action Latency	Uniform$\{0, 1, 2\}$ steps (i.e., 0–40 ms, rounded to nearest 20 ms step).

TABLE II: Reward Term Formulations for Final-Stage Locomotion

Tracking and Motion	
Tracking Linear Velocity	$r = \exp\left(-\frac{\lVert \mathbf{v}_{\text{cmd}} - \mathbf{v}_{\text{local}} \rVert^2}{2\sigma^2}\right)$
Tracking Angular Velocity	$r = \exp\left(-\frac{(w_{\text{cmd}} - w_{\text{base}})^2}{2\sigma^2}\right)$
Angular Velocity XY Penalty	$r = -\lVert \boldsymbol{\omega}_{xy} \rVert^2$
Orientation Penalty	$r = -\lVert \text{rot\_up}_{xy} \rVert^2$
Control and Smoothness	
Torque Penalty	$r = -\left(\lVert \boldsymbol{\tau} \rVert_2 + \lVert \boldsymbol{\tau} \rVert_1\right)$
Action Rate Penalty	$r = -\lVert \mathbf{a}_t - \mathbf{a}_{t-1} \rVert^2$
Gait and Foot Contact	
Feet Air Time Reward	$r = \mathbf{1}_{\lVert \mathbf{c}_{\text{cmd}} \rVert > \epsilon} \cdot \sum (t_{\text{air}} - t_{\text{thresh}}) \cdot \mathbf{1}_{\text{contact}}$
Foot Slip Penalty	$r = -\left(\lVert \mathbf{v}_{\text{foot}} \rVert_2 + \lVert \boldsymbol{\omega}_{\text{foot}} \rVert_2\right) \cdot \mathbf{1}_{\text{contact}}$
Feet Phase Reward	$r = \mathbf{1}_{\lVert \mathbf{c}_{\text{cmd}} \rVert > \epsilon} \cdot \exp\left(-\frac{\lVert \mathbf{z}_{\text{foot}} - r_z \rVert^2}{2\sigma^2}\right)$
Posture and Termination	
Standstill Penalty	$r = \mathbf{1}_{\lVert \mathbf{c}_{\text{cmd}} \rVert < \epsilon} \cdot \lVert \mathbf{q}_{\text{joint}} - \mathbf{q}_{\text{default}} \rVert_1$
Early Termination Penalty	$r = \mathbf{1}_{\text{done} \wedge (t < t_{\max})} \cdot (-1.0)$

balance and uprightness. *Control penalties* discourage excessive torque use, abrupt action changes, and high-magnitude actions to ensure smooth and energy-efficient behavior. *Gait-related terms*, including feet air time, foot slip penalties, and a feet phase reward, encourage natural stepping patterns, reliable ground contact, and correct timing of foot placement. Additional terms like *stand_still* and *termination* ensure the robot maintains a nominal pose when idle and penalizes premature terminations, respectively. Table II lists each reward term.

D. Training Scheme

Our training pipeline follows a multi-stage curriculum that progressively increases task complexity, enhancing both policy robustness and efficiency. Fig. 6 outlines the overall process, and Table I summarizes the parameter values. All the stages include domain randomization. Trained and sim-to-sim transferred policy is visualized in Fig. 7.

The first stage trains the policy to maintain an upright posture. Rewards penalize deviations in the center of mass and body sway. In the second stage, small kicks are introduced to develop balance and recovery. The reward function penalizes imbalances while encouraging rapid stabilization. The third stage introduces terrain irregularities, making it challenging to walk on uneven surfaces and randomizing the contacts. The reward structure includes penalties for slipping or falling. In the final phase, the policy performs dynamic walking based on random commands. Rewards from previous stages are combined with task-oriented components.

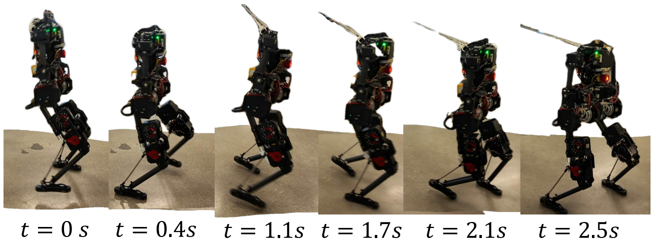


Fig. 8: BRUCE RL policy locomotion on slippery smooth concrete surface. A safety leash is loosely attached at the top.

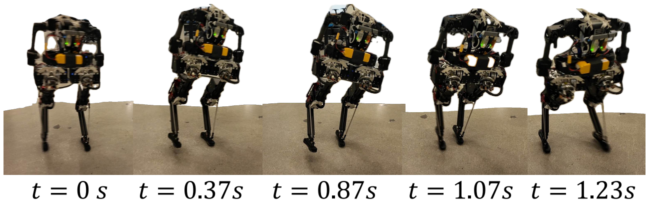


Fig. 9: BRUCE RL policy standstill perturbation rejection in sideways. BRUCE (facing backward) was able to balance on only its right leg to stabilize itself, and then took one step back. After $t = 0$ to $t = 1.07$ s the left leg was in the air. A safety leash is attached on top at loose.

The trained policy is evaluated to verify that the objective is achieved and that the policy is robust to various disturbances and domain randomization. This evaluation is done in both MJX and MuJoCo physics backends, validating the robustness through sim-to-sim transfer.

V. Results and Hardware Experiments

A. Simulation Efficiency for Parallel Mechanisms

A primary concern with incorporating closed-chain mechanisms into GPU-accelerated simulators is potential computational overhead. To quantify this, we benchmarked several BRUCE simulation variants, including a simplified serial chain model and models that incrementally incorporated the 4-bar, 5-bar, and differential closed-chain constraints. Each model executed 400 time steps using MJX with 8,192 parallel environments, on an Intel i9-13900K CPU and Nvidia RTX4090 GPU.

As summarized in Table III, the inclusion of all three parallel mechanism constraints introduced only a 3.4 % increase in per-step simulation time relative to the unconstrained model. Differential constraints contributed the most significant overhead, consistent with Section III-A due to two additional equality constraints, while the 4-bar constraint's impact was negligible. The Just-In-Time (JIT) compilation time increased 10.5 % with all parallel mechanisms constraints, reflecting MJX GPU initialization. For typical reinforcement learning workloads, this one-time cost is amortized and insignificant compared to the overall training duration.

B. Sim-to-Real Transfer: Policy Validation on Hardware

To validate our approach, we deployed policies trained with high-fidelity closed-chain constraints directly on a BRUCE hardware unit, without additional fine-tuning.

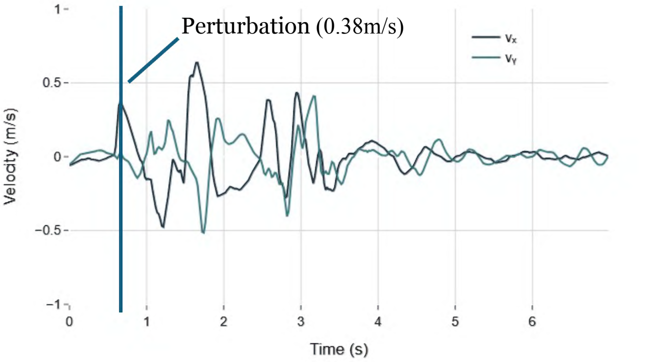


Fig. 10: Perturbation disturbance rejection from standstill. The graph shows the linear velocity odometry over time. The robot was pulled forward (x-axis direction) at 0.38 m/s at $t = 0.66$ s. The robot took steps to stabilize the torso orientation.

TABLE III: Relative simulation overhead for each parallel mechanism on BRUCE. Measurements from MJX with 8,192 parallel environments over 400 time steps each.

	Simplified	4-Bar	5-Bar	Differential	All
Time/step	0.52 [μs]	**0.0%**	**+1.2%**	**+2.3%**	**+3.4%**
Steps/sec	1,907,040	**0.0%**	**-1.2%**	**-2.3%**	**-3.4%**
JIT time	16.07 [s]	**0.0%**	**+4.3%**	**+6.9%**	**+10.5%**

1) Experimental Setup: BRUCE's onboard computer (Khadas Edge2, 8-core 2.25 GHz Cortex-A76) executes all inference and control. The RL policy runs at 50 Hz, whereas the convex MPC baseline [2] is at 500 Hz. Control input in both MPC and RL cases consists of desired body linear and angular yaw velocities. We evaluate performance on a variety of real-world surfaces: foam, synthetic grass, and smooth concrete, such as in Fig. 8.

2) Locomotion Robustness and Adaptability: The RL policy demonstrates an adaptive walking style, standing still when no velocity commands are provided and dynamically transitioning to a natural gait once a disturbance is felt or commanded velocities are received.

a) Standstill stability: BRUCE RL policy has demonstrated an agile response to various perturbations. Fig. 9 shows the RL policy reacting to a sideways disturbance with only the right leg and the left leg was in the air for 1.07 s, which is more natural, human-like sideways balancing behavior. Fig. 10 graphs the perturbation rejection from standstill, which is more challenging since the robot has to quickly react to the disturbance and take a step if necessary. At $t = 0.66$ s, perturbation of 0.38 m/s in the $+\dot{x}$ forward direction, which is significant given the scale of the robot. The RL policy tried to pull back, and then took two steps forward as seen in the linear velocity odometry in Fig. 10. The robot torso reached steady state after 5 s from the push. This showcases RL policy adaptiveness and stability.

b) Walking: In Fig. 8, the RL policy was set to step at 1.9 Hz, and phase tracking accuracy was 1.91 Hz over 10 s of straight walk. However, the RL policy was able to adjust the gait frequency and sequence naturally in response to external

TABLE IV: In-place stepping success across surfaces

Method	Synthetic Grass	Foam	Smooth Concrete
RL Policy	✓	✓	✓
MPC Baseline	✓		

disturbances. In contrast, the MPC controller struggles to respond to disturbances from a standstill, requiring constant stepping. Although the gait phase time is adaptable in MPC, the sequence is fixed.

3) Surface Generalization: Table IV summarizes the performance of RL and MPC controllers for in-place stepping across different surfaces. The RL policy consistently succeeded on all tested surfaces, including highly resistive synthetic grass, high-friction foam, and smooth, low-friction concrete. In contrast, the MPC controller could sustain stepping on synthetic grass for 15 minutes but failed to operate reliably on foam and concrete despite tuning efforts. For our RL policy, it failed due to excessive bouncing of the foot when stepping, which is more commonly observed on hard concrete. In the future, such contact impedance should be better modeled and domain-randomized in the training.

4) Locomotion Speed: On hardware, the RL policy achieved a peak forward walking speed of 0.18 m/s on a flat, smooth concrete floor. While the maximum speed is 28 % lower than the reported MPC maxima, 0.25 m/s [39], the RL policy can operate on a broader range of surfaces.

5) Policy Inference and Computation Cost: The policy runs at 50 Hz, but the RL policy network executes at 3.2 ms per inference. The MPC pipeline computes faster but must run at a higher frequency of 500 Hz and perform online quadratic programming, and requires separate kinematics computations. Our policy architecture, enabled by the closed-chain simulation, eliminates the need for extra forward and inverse kinematics or state estimation on hardware, reducing CPU consumption and system complexity.

6) Ablation Study: Effect of Parallel Mechanism Fidelity on Sim-to-Real Gap: We compared sim-to-real performance with and without explicit simulation of passive joint compliance and backlash. Policies trained without such a parallel mechanism's backlash modeling were brittle in terms of hardware compliance. They tended to fail during backward tipping events, as BRUCE's heel is short and passive compliance limits force exertion at the heel. Including this compliance and backlash in the simulation led to qualitatively improved disturbance response strategies, with earlier footlift and stepping actions.

VI. Discussion and Limitation

This section discusses the observed benefits of our RL training framework and its inherent limitations.

A. Benefits in RL Training

1) System identification: The motor system identification is a crucial part of zero-shot RL policy deployment. However, system identification on parallel mechanisms is challenging since the passive joints often lack encoders and sensors. With Jacobian stiffness modeling [40], etc., can provide estimation, it becomes a challenge to integrate into the simulation. Simulating the full parallel mechanisms simplified the system identification process, as the same motors drive all lower body joints.

2) Torque and Velocity: Incorporating parallel mechanisms directly into the simulation is a critical step toward achieving a truly end-to-end RL policy. Aligning the simulation's action space with the hardware's actuation space enables a more seamless policy transfer. This closed-chain simulation allows for non-positional commands, such as torque or velocity, without requiring explicit inverse dynamics or Jacobian computations.

Even when using joint position as the control input, as done in this work, the reward function includes an energy minimization term in Section IV-C. This term becomes inaccurate if the mechanism model is simplified or a kinematic joint space is used. For example, in general gear ratio configurations of differential pulleys, minimizing output torques does not guarantee a reduction in actuator torques.

B. Nonlinear Actuation Joint Limit

A key limitation in simulating closed-chain kinematics is the nontrivial and configuration-dependent nature of joint limits in actuated coordinates. This issue can be mitigated in simulation by setting passive joint limits or adding self-collision constraints to enforce physical realism. In hardware, however, the lack of sensing at passive joints makes it nontrivial to enforce these limits without forward kinematics. For actuators driving parallel mechanisms, the action scale may require additional consideration when the position is a policy action. In our setup, the RL policy learned kinematic joint limits implicitly through training, and we have not encountered any joint limit violations during hardware deployment.

C. Parallel Mechanisms with Compliant Members

While the current simulation models only include backlash effects, compliant parallel mechanisms—such as those with embedded springs—are common in legged systems [7]. The equality constraint formulation in (10) can be tuned to approximate spring-damper behavior for small displacements. However, modeling large-deformation compliant elements or flexure members would require additional extensions that are not addressed in this work.

VII. Conclusion

This work presents an end-to-end curriculum reinforcement learning framework for humanoid robots equipped with parallel mechanisms, demonstrated on BRUCE, a kid-sized humanoid with differential pulleys, 5-bar, and 4-bar linkages. Incorporating full closed-chain kinematic constraints in MJX simulation enables learning directly in the hardware actuator space, preserving the mechanical intelligence of parallel actuation. Our experiments show that accurate modeling improves control fidelity and seamless sim-to-real transfer. Comparisons with model predictive control (MPC) further validate the effectiveness of our RL policy in real-world deployment. This study highlights the importance of embracing mechanical structure in learning-based control and opens the door for broader integration of parallel mechanisms in legged robot RL training pipelines.

References

[1] M. Chignoli, D. Kim, E. Stanger-Jones, and S. Kim, "The mit humanoid robot: Design, motion planning, and control for acrobatic behaviors," in *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2021, pp. 1–8.

[2] Y. Liu, J. Shen, J. Zhang, X. Zhang, T. Zhu, and D. Hong, "Design and control of a miniature bipedal robot with proprioceptive actuation for dynamic behaviors," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 8547–8553.

[3] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Real-world humanoid locomotion with reinforcement learning," *Science Robotics*, vol. 9, no. 89, p. eadi9579, 2024.

[4] T. Haarnoja, B. Moran, G. Lever, S. H. Huang, D. Tirumala, J. Humplik, M. Wulfmeier, S. Tunyasuvunakool, N. Y. Siegel, R. Hafner *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *Science Robotics*, vol. 9, no. 89, p. 8022, 2024.

[5] L. de Matteis, V. Batto, J. Carpentier, and N. Mansard, "Optimal control of walkers with parallel actuation," *arXiv preprint arXiv:2504.00642*, 2025.

[6] G. Kenneally, A. De, and D. E. Koditschek, "Design principles for a family of direct-drive legged robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 900–907, 2016.

[7] H. Shin, T. Ishikawa, T. Kamioka, K. Hosoda, and T. Yoshiike, "Mechanistic properties of five-bar parallel mechanism for leg structure based on spring loaded inverted pendulum," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2019, pp. 320–327.

[8] M. Chignoli, J.-J. Slotine, P. M. Wensing, and S. Kim, "Urdf+: An enhanced urdf for robots with kinematic loops," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, 2024, pp. 197–204.

[9] Y. Tanaka, Y. Shirai, X. Lin, A. Schperberg, H. Kato, A. Swerdlow, N. Kumagai, and D. Hong, "Scaler: A tough versatile quadruped free-climber robot," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 5632–5639.

[10] D. J. Blackman, J. V. Nicholson, C. Ordonez, B. D. Miller, and J. E. Clark, "Gait development on minitaur, a direct drive quadrupedal robot," in *Unmanned Systems Technology XVIII*, vol. 9837. SPIE, 2016, pp. 141–155.

[11] G. Kenneally and D. E. Koditschek, "Leg design for energy management in an electromechanical robot," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 5712–5718.

[12] M. P. Austin, J. M. Brown, C. A. Young, and J. E. Clark, "Leg design to enable dynamic running and climbing on bobcat," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2018, pp. 3799–3806.

[13] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Spröwitz, A. Abate, and J. Hurst, "Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1497–1521, 2016.

[14] K. G. Gim, J. Kim, and K. Yamane, "Design and fabrication of a bipedal robot using serial-parallel hybrid leg mechanism," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 5095–5100.

[15] A. Roig, S. K. Kothakota, N. Miguel, P. Fernbach, E. M. Hoffman, and L. Marchionni, "On the hardware design and control architecture of the humanoid robot kangaroo," in *6th workshop on legged robots during the international conference on robotics and automation*, 2022.

[16] Y. Tanaka, A. Schperberg, A. Zhu, and D. Hong, "Scaler-b: A multimodal versatile robot for simultaneous locomotion and grasping," in *IEEE International Conference on Robotics and Automation 40*.

[17] K. Kawaharazuka, S. Yoshimura, T. Suzuki, K. Okada, and M. Inaba, "Design optimization of wire arrangement with variable relay points in numerical simulation for tendon-driven robots," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1388–1395, 2023.

[18] T. Makabe, K. Okada, and M. Inaba, "Design of morphable statenet based on pseudo-generalization of standing up motions for humanoid with variable body structure," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 15 336–15 342.

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

[20] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[21] H. Tan, "Reinforcement learning with deep deterministic policy gradient," in *2021 International conference on artificial intelligence, big data and algorithms (CAIBDA)*. IEEE, 2021, pp. 82–85.

[22] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, p. eabc5986, 2020. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abc5986

[23] R. P. Singh, M. Morisawa, M. Benallegue, Z. Xie, and F. Kanehiro, "Robust humanoid walking on compliant and uneven terrain with deep reinforcement learning," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, 2024, pp. 497–504.

[24] S. Kaleem and I. Torshin, "Reinforcement learning in robotics: Challenges and opportunities," 11 2023.

[25] S. Cabi, S. G. Colmenarejo, A. Novikov, K. Konyushkova, S. Reed, R. Jeong, K. Zolna, Y. Aytar, D. Budden, M. Vecerik, O. Sushkov, D. Barker, J. Scholz, M. Denil, N. de Freitas, and Z. Wang, "Scaling data-driven robotics with reward sketching and batch reinforcement learning," 2020. [Online]. Available: https://arxiv.org/abs/1909.12200

[26] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning." arXiv, 2022.

[27] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei, "Transic: Sim-to-real policy transfer by learning from online correction," 2024. [Online]. Available: https://arxiv.org/abs/2405.10315

[28] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," *CoRR*, vol. abs/2009.13303, 2020.

[29] J. Josifovski, S. Auddy, M. Malmir, J. Piater, A. Knoll, and N. Navarro-Guerrero, "Continual domain randomization," 2024. [Online]. Available: https://arxiv.org/abs/2403.12193

[30] T. Huang, N. Sontakke, K. N. Kumar, I. Essa, S. Nikolaidis, D. W. Hong, and S. Ha, "Bayrntune: Adaptive bayesian domain randomization via strategic fine-tuning," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024, pp. 670–676.

[31] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Learning humanoid locomotion with transformers," *arXiv:2303.03381*, 2023.

[32] E. Valassakis, Z. Ding, and E. Johns, "Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics," in *International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[33] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," 2020. [Online]. Available: https://arxiv.org/abs/2003.04960

[34] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," 2018. [Online]. Available: https://arxiv.org/abs/1707.05300

[35] A. Zhu, Y. Tanaka, and D. Hong, "Aura: Agentic upskilling via reinforced abstractions," *arXiv preprint arXiv:2506.02507*, 2025.

[36] 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.

[37] A. Wachi and Y. Sui, "Safe reinforcement learning in constrained markov decision processes," in *International Conference on Machine Learning*. PMLR, 2020, pp. 9797–9806.

[38] S. O. Madgwick *et al.*, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," 2010.

[39] J. Shen, J. Zhang, Y. Liu, and D. Hong, "Implementation of a robust dynamic walking controller on a miniature bipedal robot with proprioceptive actuation," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, 2022, pp. 39–46.

[40] Y. Tanaka, Y. Shirai, A. Schperberg, X. Lin, and D. Hong, "Scaler: Versatile multi-limbed robot for free-climbing in extreme terrains," *IEEE Transactions on Robotics, (TRO)*, 2025.