Sim-to-Sim Transfer of a Learned Legged Locomotion Policy: Isaac Gym to MuJoCo under Command Switching

Disthorn Suttawet
Institute of Field Robotics
King Mongkut's University of Technology Thonburi – KMUTT
Email: ioonza1d@gmail.com

***Abstract*—Deep reinforcement learning enables agile legged locomotion in simulation, yet policies often degrade when transferred across simulators or to real hardware. Transfer quality is commonly assessed using steady-state tracking metrics, which frequently miss instability that only appears during transient behaviors.**

This paper studies sim-to-sim transfer of a learned quadruped locomotion policy from Isaac Gym (PhysX) to MuJoCo using *command switching* as a systematic stress test for transient dynamics. By transferring identical policy weights without retraining, we isolate discrepancies from actuator stiffness, contact friction, solver dynamics, and observation latency.

Our results show that a 9.4% steady-state tracking gap masks much larger divergences during transient maneuvers: peak torque gaps of 35% and doubled pitch excursions during turning. Foot-level friction and observation delay are the dominant instability sources. Notably, a 20 ms observation delay causes robot falls during turning despite minimal impact on steady-state tracking. We compare three mismatch reduction strategies of ActuatorNet, residual actuator learning, and domain-randomized policy retraining. Finding that residual learning provides fast, reliable improvements while domain randomization yields the best overall robustness with 42–44% roll reduction.

These findings support transient response metrics as a more sensitive transferability indicator than steady-state error, and motivate sim-to-sim validation as a critical step in sim-to-real pipelines.

I. Introduction

Deep reinforcement learning (DRL) can produce remarkably agile locomotion behaviors entirely in simulation [1], [2], [5]. Yet transferring these policies to real hardware or even to a different simulator often reveals fragility that was invisible during training. The gap arises from systematic differences in dynamics modeling, contact solvers, actuator behavior, and timing.

Most prior evaluations rely on steady-state tracking metrics such as average velocity error [3], [4]. These metrics can miss instability that only appears during *transient* behaviors: rapid turns, abrupt stops, and direction reversals. Such transients occur constantly during real deployment, where robots respond to changing commands, terrain, and disturbances.

Sim-to-sim transfer is a practical intermediate validation step. By deploying a trained policy in a second simulator with a different physics engine without retraining, which helps identify failure modes before risking hardware. We use the Unitree Go2 quadruped, training in Isaac Gym (PhysX) and evaluating in MuJoCo without modification to the policy weights.

While we verified that IsaacLab produces equivalent MuJoCo deployment performance, we chose Isaac Gym as the primary training framework because its sim-to-sim pipeline requires no joint order remapping and remains widely adopted in legged locomotion research.

Our contributions are:

- We show that command switching exposes gaps invisible to steady-state evaluation, with transient torque gaps reaching 35% despite only a 9.4% steady-state difference.
- We identify foot friction and observation delay as the dominant instability sources; 20 ms observation delay causes falls while motor delay of the same magnitude is survivable.
- We compare three mismatch reduction approaches and provide a practical deployment checklist for Isaac Gym to MuJoCo transfer.

II. Related Work

A. Sim-to-Real Transfer for Legged Locomotion

Prior work spans several actuator types, each requiring different transfer strategies. Tan *et al.* [3] used analytical actuator models for Minitaur (direct-drive). Hwangbo *et al.* [1] introduced ActuatorNet for ANYmal (series-elastic actuators), showing that systematic excitation data is essential for capturing complex dynamics. Peng *et al.* [4] applied domain randomization to Laikago (quasi-direct-drive), and Kumar *et al.* [2] proposed Rapid Motor Adaptation (RMA) for the Unitree A1.

Our work uses the Go2, whose quasi-direct-drive (QDD) actuators sit between the direct-drive and SEA extremes. This makes it an ideal testbed for comparing all three paradigms.

B. Transient Response Analysis

Lee *et al.* [5] noted that policies robust during steady-state flat terrain can fail during rapid terrain transitions. We extend this observation by formally defining command-switching scenarios as stress tests and using quantitative transient metrics (rise time, settling time, overshoot, peak torque) to compare simulators and mismatch reduction strategies.

III. Problem Setup

A. Robot Platform

The Unitree Go2 is a 12-DOF quadruped (3 joints per leg: hip abduction, hip flexion, knee) with quasi-direct-drive actuators at gear ratio ∼6–10:1. Table I places it in context relative to prior work.

TABLE I
ROBOT PLATFORMS IN PRIOR SIM-TO-REAL WORK AND THEIR ACTUATOR TYPES.

Paper	Robot	Actuator	Transfer method
Tan et al. [3]	Minitaur	Direct-drive	Analytical model
Hwangbo et al. [1]	ANYmal	SEA	ActuatorNet
Peng et al. [4]	Laikago	QDD	Domain random.
Kumar et al. [2]	Unitree A1	QDD	RMA
This work	**Go2**	**QDD**	**All three**

B. Policy Architecture and Training

The policy is trained in Isaac Gym with PPO on a 48-dimensional observation vector (Table II) and outputs target joint position offsets executed through a PD controller ($K_p = 20$ N·m/rad, $K_d = 0.5$ N·m·s/rad, action scale 0.25).

TABLE II
OBSERVATION VECTOR STRUCTURE (48 DIMENSIONS TOTAL).

Index	Description	Dim	Scale
0–2	Base linear velocity (body frame)	3	×2.0
3–5	Base angular velocity (body frame)	3	×0.25
6–8	Projected gravity vector	3	×1.0
9–11	Velocity command $[v_x, v_y, \omega_z]$	3	cmd_scale
12–23	Joint position offsets from default	12	×1.0
24–35	Joint velocities	12	×0.05
36–47	Previous action	12	×1.0

C. Simulator Configurations

Table III summarizes the key differences between the two simulators. The most important structural difference is how friction is computed. MuJoCo uses the geometric mean of both contact surfaces:

$$\mu_{\text{eff}} \approx \sqrt{\mu_{\text{floor}} \times \mu_{\text{foot}}} \tag{1}$$

With Isaac Gym's effective $\mu \approx 1.0$ vs. MuJoCo's $\mu_{\text{eff}} \approx 0.63$ (from $\mu_{\text{foot}} = 0.4$), a friction mismatch exists from the start.

D. Deployment Checklist and Interface Alignment

Before measuring physics mismatch, all interface differences must be resolved. Failure to address any of the items in Table IV makes it impossible to tell physics mismatch from implementation bugs.

Items 1–3 fix structural mismatches; items 4–6 fix numerical and control-mode mismatches; items 7–8 are sanity checks before any experiment proceeds. A non-obvious pitfall is item 6: the training config uses `heading_command = True`, meaning `commands[:,2]` (angular velocity) is computed from a heading error, not set directly. Setting it directly gets silently overwritten, causing the robot to rotate at a constant rate even when commanded to go straight. The fix is to set `commands[:,3] = current_heading`.

TABLE III
SIMULATOR CONFIGURATION COMPARISON.

Parameter	Isaac Gym	MuJoCo
Physics engine	PhysX	MuJoCo
Timestep / Policy Hz	200 Hz / 50 Hz	200 Hz / 50 Hz
K_p / K_d	20 / 0.5	20 / 0.5
μ_{floor} / μ_{foot}	1.0 / 1.0	1.0 / 0.4
Quaternion order	(x, y, z, w)	(w, x, y, z)
Torque limits	±30 N·m (implicit)	None

TABLE IV
SIM-TO-SIM DEPLOYMENT CHECKLIST: REQUIRED FIXES FOR ISAAC GYM → MUJOCO TRANSFER (GO2).

#	Fix	Detail
1	Quaternion reorder	$(x, y, z, w) \to (w, x, y, z)$
2	Actuator remapping	MuJoCo `ctrl`: FR, FL, RR, RL order
3	Velocity frame	World → body via `quat_rotate_inverse`
4	Observation scales	lin_vel×2.0, ang_vel×0.25, dof_vel×0.05
5	Torque clipping	Add explicit ±30 N·m clip
6	Heading command	Set `commands[:,3]`, not `[:,2]`, when `heading_command=True`
7	Zero-action stability	Height ≈0.27 m, roll/pitch < 3
8	Observation parity	Gravity diff <0.03, lin_vel diff <0.05

IV. Command Switching as a Stress Test

Table V defines the three command-switching scenarios used throughout this paper. The switch occurs at $t_s = 3.0$ s in every case.

TABLE V
COMMAND SWITCHING SCENARIOS. SWITCH TIME $t_s = 3.0$ S.

ID	Before ($t < 3$ s)	After ($t \geq 3$ s)	Tests
S1 Stop	$v_x = 0.6$, $v_y = 0$, $\omega_z = 0$	$v_x = 0$, $v_y = 0$, $\omega_z = 0$	Braking, pitch
S2 Turn	$v_x = 0.4$, $v_y = 0$, $\omega_z = 0$	$v_x = 0.4$, $v_y = 0$, $\omega_z = 1.0$	Yaw coupling
S3 Lateral	$v_x = 0.3$, $v_y = +0.3$, $\omega_z = 0$	$v_x = 0.3$, $v_y = -0.3$, $\omega_z = 0$	Lateral flip, roll

S2 Turn is the most challenging scenario: it consistently produces the largest sim-to-sim gaps and is used as the primary test case for all ablation experiments.

TABLE VI
INTERFACE PARITY CHECKS — ALL MUST PASS BEFORE EXPERIMENTS.

Check	Status	Result
Zero-action stability	✓Pass	Height ≈0.27 m, roll/pitch < 3
Observation parity	✓Pass	Gravity diff <0.03, lin_vel diff <0.05
Joint order	✓Pass	qpos matches; ctrl order remapped

V. EXPERIMENTAL PROTOCOL

All experiments run in closed-loop without retraining. Each episode lasts 6 s from a fixed nominal stance, with a command switch at $t_s = 3.0$ s. The policy runs at 50 Hz via zero-order hold over a 200 Hz simulation.

All 10 episodes per configuration start from identical initial conditions (fixed pose, zero velocity). This is intentional: the goal is to isolate physics-engine differences, not to characterize policy variance. An episode terminates early if height drops below 0.15 m or roll/pitch exceeds 1.0 rad.

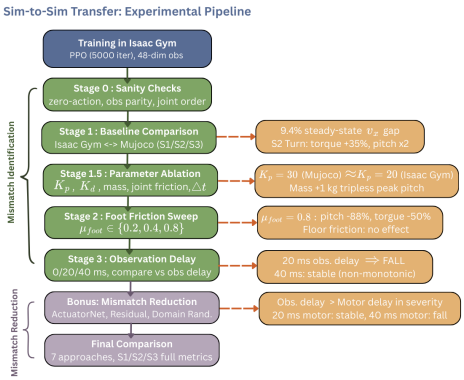

Fig. 1. Experimental sim-to-sim transfer pipeline.

Figure 1 summarizes the complete experimental workflow. We begin with policy training in Isaac Gym, then proceed through interface alignment (Stage 0), baseline comparison (Stage 1), systematic ablation to identify mismatch sources (Stages 1.5–3), and finally evaluate three mismatch reduction strategies. Each stage builds on the previous findings: Stage 1 reveals the gap, Stages 1.5–3 identify the causes, and the bonus section tests solutions.

VI. RESULTS

A. Stage 0: Sanity Checks

Before any measurement, we verify the deployment checklist (Table IV). Table VI confirms all checks passed.

B. Stage 1: Baseline Sim-to-Sim Gap

Table VII shows steady-state tracking under a constant 0.5 m/s forward command. The 9.4% velocity gap might seem acceptable.

TABLE VII
STEADY-STATE PERFORMANCE ($v_x = 0.5$ M/S, 10 S EPISODE).

Metric	Isaac Gym	MuJoCo	Gap
v_x mean (m/s)	0.498	0.451	−9.4%
v_x RMSE (m/s)	0.079	0.066	−16%
ω_z error (rad/s)	0.216	0.007	−97%
Mean torque (N·m)	2.56	2.97	+16%
Peak torque (N·m)	17.29	15.71	−9%

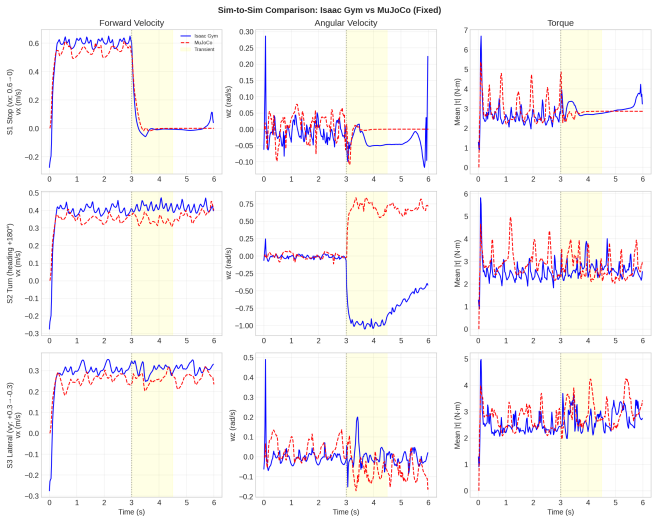

Fig. 2. Sim-to-sim divergence across S1 Stop, S2 Turn, and S3 Lateral: velocity, torque, and body angles.

But Table VIII shows command switching reveals much larger gaps. S2 Turn produces the worst divergence: +3.95 N·m peak torque (+35%) and doubled pitch excursion.

A notable observation from S2 Turn: Isaac Gym accumulated -117.3 (CCW) while MuJoCo accumulated $+118.0$ (CW) during the same command. The magnitudes match but the signs are opposite, indicating a yaw sign convention difference between PhysX and MuJoCo, not a policy failure.

Figure 2 visualizes these gaps across the full 6-second episode for all three scenarios. The yellow-highlighted transient windows show when divergences are largest. In S1 Stop (top row), both simulators decelerate successfully but MuJoCo exhibits higher torque oscillations during braking. In S2 Turn (middle row), the opposite-sign yaw behavior is clearly visible: Isaac Gym's ω_z (blue) goes negative while MuJoCo's (red) goes positive in the middle-center angular velocity plot. S3 Lateral (bottom row) shows both simulators reversing lateral direction, but with different overshoot and torque profiles during the transition.

C. Stage 1.5: Parameter Ablation

To find the cause of each gap, we vary one parameter at a time in MuJoCo. Table IX summarizes all ablation results.

1) Actuator Stiffness (K_p): MuJoCo needs about 50% higher stiffness to achieve the same response as Isaac Gym,

TABLE VIII
TRANSIENT RESPONSE UNDER COMMAND SWITCHING. POST-SWITCH WINDOW: $[t_s,\ t_s + 1.5]$ S. Δ = MUJOCO − ISAAC GYM.

Scenario	Peak Torque (N·m)			Max Pitch (°)			Fall?	
	Isaac Gym	MuJoCo	Δ	Isaac Gym	MuJoCo	Δ	Isaac Gym	MuJoCo
S1 Stop	15.26	16.29	+1.03	3.8	4.8	+1.0	No	No
S2 Turn	11.37	**15.32**	**+3.95**	3.0	**5.0**	**+2.0**	No	No
S3 Lateral	12.75	13.77	+1.02	3.8	5.2	+1.4	No	No

TABLE IX
PARAMETER ABLATION SUMMARY. EACH ROW VARIES ONE PARAMETER WHILE HOLDING ALL OTHERS AT THEIR BASELINE VALUE.

Parameter	Tracking	Stability	Key finding
K_p stiffness	High	Medium	K_p=30 in MuJoCo matches Isaac Gym K_p=20; PhysX behaves softer
K_d damping	Low	Low	3× variation changes overshoot by <0.1 rad/s; K_p dominates
Base mass	Medium	High	+1 kg nearly triples pitch (5.0°→15.5°); removing −1 kg minimal
Joint friction	Low	Medium	Removing friction raises peak roll 82%; no PhysX equivalent
Foot friction	Medium	High	μ=0.8 cuts pitch 88%, torque 50%; floor friction has zero effect
Floor friction	None	None	No change across 0.5–1.5; bottleneck is foot friction
Timestep	Low	Low	<1 cm/s change at fixed 50 Hz policy rate
Observation delay	Medium	**Critical**	20 ms (1 step) causes fall; 40 ms survives (non-monotonic)
Motor command delay	Low	High	20 ms survivable; 40 ms causes fall; less critical than obs. delay

TABLE X
K_p SWEEP: FORWARD VELOCITY TRACKING IN MUJOCO ($v_x^{\text{CMD}} = 0.5$ M/S).

K_p (N·m/rad)	v_x mean (m/s)	v_x error (m/s)
10	0.101	0.399
20 (baseline)	0.451	0.049
30	**0.466**	**0.034**
40	0.407	0.093

because PhysX behaves "softer" than MuJoCo's Newton-based contact solver.

In contrast, varying K_d by 3× (0.3 to 1.0) changes wz overshoot by less than 0.1 rad/s with no falls, confirming that K_p, not K_d, is the dominant PD gain mismatch.

TABLE XI
MASS PERTURBATION EFFECTS ON S2 TURN TRANSIENT METRICS.

Mass (kg)	Δm	v_x	Max pitch	Peak torque
5.921	−1 kg	0.363 m/s	3.8°	13.63 N·m
6.921	baseline	**0.353**	**5.0°**	**15.32 N·m**
7.921	+1 kg	0.358	**15.5°**	**23.11 N·m**
8.921	+2 kg	0.237	11.2°	21.56 N·m

2) Mass Perturbation: The effect is asymmetric: adding +1 kg nearly triples peak pitch and raises peak torque by 51%, while removing the same amount has minimal impact. This asymmetry is why the DR policy randomizes mass as $[-1,\ +2]$ kg rather than symmetrically.

D. Stage 2: Foot Friction Sweep

Floor friction swept from 0.5 to 1.5 produced *zero* measurable change. This is because MuJoCo computes effective friction as the geometric mean (Eq. 1), so foot friction ($\mu_{\text{foot}} = 0.4$) is the actual bottleneck.

TABLE XII
FOOT FRICTION SWEEP ON S2 TURN ($\mu_{\text{FLOOR}} = 1.0$, CONSTANT).

μ_{foot}	v_x	ω_z overshoot	Max pitch	Peak torque
0.2	0.294 m/s	+1.87 rad/s	25.1°	27.6 N·m
0.4 (base)	**0.353**	−0.16	5.0°	15.3 N·m
0.8	0.353	−0.25	**3.0°**	**13.7 N·m**

Increasing μ_{foot} from 0.2 to 0.8 reduces peak pitch by 88% and peak torque by 50%. At $\mu_{\text{foot}} = 0.2$, the system enters a slip-prone regime where lateral forces cannot be transmitted to the ground. This confirms that the friction mismatch between Isaac Gym ($\mu \approx 1.0$) and MuJoCo ($\mu_{\text{eff}} \approx 0.63$) contributes significantly to the observed sim-to-sim gap.

E. Stage 3: Observation Delay

Just 20 ms of observation delay (one policy step) causes a fall during turning, with roll reaching 167.8°. Delayed state feedback causes the policy to over-correct, creating an unstable oscillation. Counterintuitively, 40 ms delay does not cause a fall, suggesting that specific delay durations interact with gait timing in unpredictable ways is a form of sensitivity that would not be predicted by steady-state analysis alone.

Since real robots typically have 10–50 ms sensing latency, this result has direct implications for sim-to-real transfer.

TABLE XIII
OBSERVATION DELAY EFFECTS ON S2 TURN TRANSIENT METRICS.

Metric	0 ms	20 ms	40 ms
v_x mean (m/s)	0.353	0.330	0.322
ω_z overshoot	−0.16	+1.35	+0.97
Max roll (°)	5.1	**167.8**	18.8
Max pitch (°)	5.0	**35.1**	28.4
Peak torque (N·m)	15.3	**27.6**	26.2
Fall?	No	**YES**	No

F. Stage 3.5: Motor Command Delay

Motor command delay of 20 ms is survivable while the same observation delay causes an immediate fall. This asymmetry makes physical sense: observation delay forces the policy to act on stale state information, compounding errors through over-correction. Motor command delay still lets the policy observe correct state and compute appropriate actions; only execution is delayed.

The practical implication: prioritize low-latency state estimation over actuator response time when deploying to real hardware.

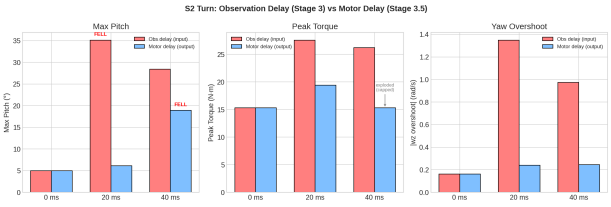

Fig. 3. Observation delay vs. motor command delay comparison. 20 ms observation delay causes a fall; 20 ms motor delay is stable.

TABLE XIV
MOTOR COMMAND DELAY EFFECTS ON S2 TURN. COMPARE WITH TABLE XIII FOR THE INPUT/OUTPUT ASYMMETRY.

Metric	0 ms	20 ms	40 ms
ω_z overshoot (rad/s)	−0.16	+0.24	+0.25
Max pitch (°)	5.0	6.1	18.9
Max roll (°)	5.1	5.3	**73.5**
Peak torque (N·m)	15.32	19.44	exploded
Fall?	No	No	**Yes**

G. Summary of Mismatch Sources

Three tiers emerge from Table XVI. *Critical* sources cause falls under conditions relevant to real deployment. *High-impact* sources produce large but non-fatal changes. And three *negligible* sources (floor friction, K_d, timestep) can be removed from calibration checklists entirely.

VII. MISMATCH REDUCTION STRATEGIES

We compare three complementary strategies, each targeting a different layer of the mismatch problem.

A. ActuatorNet: Direct Torque Replacement

We replace the PD controller with a learned torque predictor following Hwangbo *et al.* [1]. Three versions were developed, each with different training data. Table XV summarizes the progression.

The key lesson is that *data diversity matters more than test-set* R^2.

V1 achieved 99.21% accuracy but failed at deployment because normal walking data does not cover the transient regimes needed for command switching.

V2 expanded the dataset to 750,000 samples using policy-driven excitation (random perturbations, high-speed commands, sudden switches). This improved steady-state performance and stabilized constant turning, but V2 became *unstable* during S2 Turn command switching, with peak pitch reaching 17.8° and torque spiking to 28.5 N·m. The problem: even policy-driven data is biased toward the locomotion task and misses frequency-domain coverage at the actuator level.

V3 uses Hwangbo-style systematic excitation: sinusoidal sweeps (0.5–10 Hz), chirp signals, torque saturation probing, and multi-sine compositions (420,000 samples total)—all collected *independently of the locomotion policy*. This ensures frequency-domain coverage that task-driven data cannot provide, resolving the S2 Turn instability with an 84% pitch reduction (17.8°→2.8°) and 53% torque reduction.

TABLE XV
ACTUATORNET VERSION COMPARISON ON S2 TURN COMMAND SWITCHING.

Ver.	Data type	R^2	S2 stable?	S2 pitch
V1	Normal walking (360k)	99.21%	–	–
V2	Policy excitation (750k)	94.55%	No	17.8°
V3	Hwangbo systematic (420k)	99.80%	**Yes**	**2.8°**

B. Residual Learning

Rather than replacing PD control, we learn an additive correction:

$$\tau = \tau_{\text{PD}} + \Delta\tau_{\text{learned}} \tag{2}$$

where $\Delta\tau_{\text{learned}}$ is trained on residuals $\Delta\tau = \tau_{\text{Isaac}} - \tau_{\text{PD}}$ collected from Isaac Gym rollouts. Table XVII shows the results.

Investigation reveals the residual captures two distinct effects:

(1) Implicit torque clipping: Isaac Gym clips torques at ±30 N·m, a constraint absent in default MuJoCo. However, simply adding explicit clipping to the PD controller is *insufficient*, the robot still falls during constant turning. This proves that clipping alone does not explain the full mismatch.

(2) Velocity-dependent compensation: At joint velocities above 2 rad/s, the residual network predicts positive torque corrections of +0.3 to +1.5 N·m (correlation $r = -0.49$

TABLE XVI
MISMATCH SOURCE SEVERITY SUMMARY, ORDERED BY IMPACT.

Source	Tracking	Stability	Key evidence
Critical			
Observation delay	Medium	**Critical**	20 ms causes fall; non-monotonic: 40 ms survivable
High impact			
Foot friction	Medium	High	μ=0.2→0.8 cuts pitch 88%, torque 50%; floor friction: zero effect
Mass perturbation	Medium	High	+1 kg nearly triples pitch (5.0°→15.5°); asymmetric
Motor command delay	Low	High	20 ms survivable; 40 ms causes fall
Torque clipping	Medium	High	Isaac Gym clips at ±30 N·m; clipping alone not enough
Vel.-dependent dynamics	Medium	High	Residual r=−0.49 vs. joint vel.; +0.3 to +1.5 N·m at $\|\dot{q}\|$>2 rad/s
Moderate			
Actuator stiffness (K_p)	High	Medium	K_p=30 in MuJoCo matches K_p=20 in Isaac Gym
Joint friction	Low	Medium	Removing friction raises peak roll 82%; no PhysX equivalent
Negligible – safe to skip in calibration			
Floor friction	None	None	No measurable change across 0.5–1.5
Damping (K_d)	Low	Low	<0.1 rad/s across 3× sweep
Timestep	Low	Low	<1 cm/s at fixed 50 Hz policy rate

TABLE XVII
PD BASELINE VS. PD + RESIDUAL LEARNING PERFORMANCE.

Metric	PD	PD+Resid.	Change
Steady-state ($v_x = 0.5$ m/s)			
v_x error (m/s)	0.049	0.043	−12%
Peak torque	15.71	15.42	−2%
Constant turn ($v_x = 0.4$, $\omega_z = 1.0$)			
v_x error (m/s)	0.093	0.052	−44%
ω_z error	0.395	0.338	−14%
Status	Fallen	Stable	Fixed
S2 Turn command switch			
Peak torque (N·m)	15.22	14.59	−4%
Max pitch (°)	4.7	4.6	−2%
Max roll (°)	6.0	5.4	−10%

with joint velocity). This suggests Isaac Gym applies implicit velocity-dependent friction compensation not replicated by MuJoCo's standard PD model. During rapid turning, joint velocities increase significantly, and without this compensation MuJoCo's actuator model under-compensates for dynamic friction.

This structured velocity dependence explains why residual learning stabilizes constant turning while simple clipping does not: the learned correction captures both saturation *and* the missing velocity-dependent dynamics.

C. Domain Randomization

We retrain the policy with domain randomization following the Phase 1 methodology of Kumar *et al.* [2]. Randomization covers friction ($\mu \in [0.3, 1.5]$), payload ($\Delta m \in [-1, +2]$ kg), and PD gains ($\times[0.8, 1.2]$). An asymmetric actor-critic provides the critic with 58-dimensional privileged observations during training; the actor receives the standard 48-dimensional vector.

We also attempted the full two-phase RMA methodology. Phase 1 training converged (reward = 24.9) but the environment encoder collapsed to near-zero variance ($\sigma = 0.13$), indicating the policy learned to ignore the encoding. The Phase 2 adaptation module achieved $R^2 = 0.997$ on held-out data but failed during deployment. This is a known challenge when the base policy is already sufficient without privileged information; we leave it for future work.

TABLE XVIII
ORIGINAL POLICY VS. DOMAIN-RANDOMIZED (DR) POLICY IN MUJOCO.

Metric	Original	DR Policy	Change
Steady-state ($v_x = 0.5$ m/s)			
v_x error (m/s)	0.049	0.072	+47%
Peak torque (N·m)	15.71	13.01	−17%
Constant turn ($v_x = 0.4$, $\omega_z = 1.0$)			
Status	Fallen	Stable	Fixed
v_x error (m/s)	—	0.033	—
S2 Turn switch			
Peak torque (N·m)	15.22	13.23	−13%
Max pitch (°)	4.7	3.9	−17%
Max roll (°)	6.0	**3.5**	−42%

The DR policy achieves the best stability across all scenarios, including −44% roll reduction on S3 Lateral (4.1°→2.3°). The cost is slightly lower steady-state tracking precision (+47% error), which is expected: domain randomization produces more conservative policies that sacrifice peak tracking for robustness.

D. Comprehensive Comparison

Table XX consolidates all strategies. The progression from PD-only to DR policy tells a clear story: each approach solves a different layer of the problem.

E. Transient Response Comparison

Table XIX provides full transient metrics across all scenarios for the four primary controllers. The complementary strengths are clear: DR consistently achieves the lowest torques and body excursions, while residual learning achieves the fastest response times.

Table XXI summarizes the winner per scenario and provides practical recommendations.

Recommendations: Use DR-trained for best overall robustness. Use PD+Residual when fast tracking response is more important than peak excursion. Use ActuatorNet V3 when systematic excitation data is available and pitch control is the priority.

Figure 4 Visualizes full transient plots show divergence across S1 Stop, S2 Turn, and S3 Lateral of the four primary controllers

VIII. Discussion

A. Transient Metrics as Better Transferability Indicators

The central finding is that transient response metrics are much more sensitive to sim-to-sim mismatch than steady-state tracking. The 9.4% steady-state gap looks harmless, yet the same policy shows 35% torque spikes and doubled pitch during turning. The 20 ms delay experiment makes this stark: steady-state tracking degrades by only 6%, but the robot falls. Sim-to-sim and sim-to-real validation protocols should routinely include transient stress tests.

B. Data Quality for Learned Actuator Models

The ActuatorNet progression (V1→V2→V3) shows that high test-set accuracy is not enough. V1's 99.21% R^2 was built on a narrow operating regime. V3's systematic excitation ensures the model sees the full actuator dynamic range, including the transient saturation events that V2 missed. This reinforces Hwangbo *et al.*'s original recommendation: collect excitation data independently from the locomotion policy.

C. Complementarity of Strategies

The three strategies tackle different layers. Residual learning corrects actuator-level artifacts (clipping, velocity-dependent friction) without touching the policy. ActuatorNet provides a more complete actuator model but needs careful data collection. Domain randomization makes the policy inherently robust by exposing it to diverse dynamics during training, at the cost of some tracking precision. For real-world deployment, combining actuator-level correction with domain randomization would likely provide the best of both.

D. Limitations

This study is limited to two simulators with known, rigid physics engines; it does not model sensor noise, communication delays beyond the tested latencies, or terrain variability. The non-monotonic delay sensitivity (20 ms causes falls, 40 ms does not) was observed under fixed initial conditions and may not generalize across gait phases. Future work should test stochastic delay distributions and varied terrain profiles.

IX. Conclusion

This paper presents a systematic study of sim-to-sim transfer using command switching as a transient stress test. Three takeaways stand out.

Transient metrics expose what steady-state hides. The same 9.4% steady-state gap that looks manageable becomes a 35% torque spike and doubled pitch excursion during turning. Command switching is a more honest test of transfer quality.

Foot friction and observation delay dominate instability. Raising foot friction from 0.2 to 0.8 reduces peak pitch 88%. A 20 ms observation delay causes falls; the same motor command delay does not. Floor friction, K_d, and timestep have negligible effect and can be removed from calibration checklists.

The three reduction strategies are complementary. Residual learning provides reliable, fast improvements. Domain randomization gives the best overall robustness. ActuatorNet requires systematic excitation data but delivers exceptional pitch control. None of the three is universally best; choose based on what the deployment scenario demands.

Together, the command-switching protocol and the deployment checklist (Table IV) provide a practical toolkit for validating sim-to-sim and sim-to-real transfer before any hardware experiment.

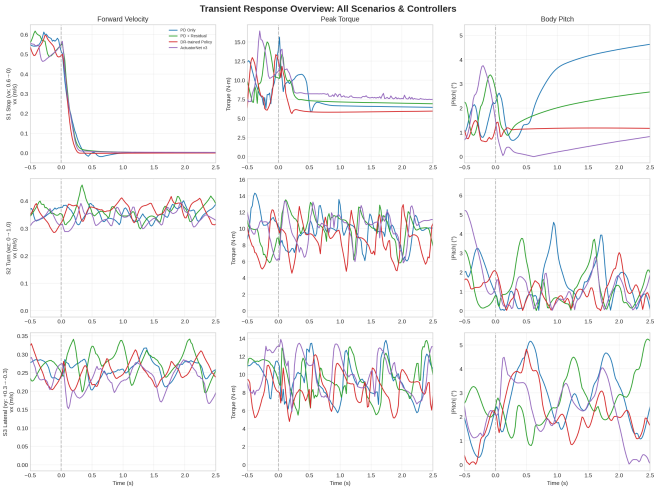

Fig. 4. Transient response across all three scenarios and all four primary controllers.

Acknowledgment

The author thanks the Institute of Field Robotics (FIBO) at KMUTT and the Sim2Real Internship program at VISTEC for the research framework and computational resources.

TABLE XIX
DETAILED TRANSIENT METRICS ACROSS ALL SCENARIOS. BEST VALUE PER METRIC IS **BOLDED**.

Scenario	Metric	PD Only	PD+Resid.	DR	ActNet V3
S1 Stop	Rise time 10–90% (ms)	300	200	**140**	180
	Settling time 5% (ms)	380	240	**160**	240
	v_x overshoot	0.5%	0.9%	−0.4%	−0.8%
	Peak torque (N·m)	15.93	16.16	**13.85**	14.03
	Peak pitch (°)	4.7	4.8	5.2	**1.5**
	Peak roll (°)	3.1	**2.4**	2.8	2.5
S2 Turn	v_x rise time (ms)	1880	**300**	880	20
	ω_z overshoot (%)	−17.1	**−10.1**	−23.3	+26.3
	Peak torque (N·m)	15.22	14.59	**13.23**	13.34
	Peak pitch (°)	4.7	4.6	3.9	**2.8**
	Peak roll (°)	6.0	5.4	**3.5**	5.1
S3 Lateral	v_x rise time (ms)	1280	**40**	1120	520
	Peak torque (N·m)	14.02	13.95	**12.18**	13.88
	Peak pitch (°)	5.3	5.2	4.8	**4.5**
	Peak roll (°)	4.1	5.5	**2.3**	6.3

TABLE XX
ALL MISMATCH REDUCTION APPROACHES ON S2 TURN. BEST VALUES BOLDED.

Approach	Baseline err.	Const. turn	S2 stable?	S2 pitch	S2 roll	Complexity
PD only	0.049 m/s	Fallen	Stable	4.7°	6.0°	Low
PD + clipping	—	Fallen	—	—	—	Low
ActuatorNet V1	0.277 m/s	—	—	—	—	Medium
ActuatorNet V2	0.062 m/s	Stable	Unstable	17.8°	15.3°	Medium
ActuatorNet V3	∼0.06	Stable	Stable	**2.8**°	5.1°	Medium
PD + Residual	**0.043**	Stable	Stable	4.6°	5.4°	Medium
DR policy	0.072	Stable	Stable	3.9°	**3.5**°	High

TABLE XXI
TRANSIENT RESPONSE SUMMARY AND RECOMMENDATIONS.

Controller	S1 Stop	S2 Turn	S3 Lateral
PD Only	Baseline	High overshoot	Baseline
PD+Residual	Fast settling	Best overshoot	Fastest rise
DR-trained	Best overall	Best stability	Best stability
ActuatorNet V3	Best pitch	Good pitch	Good pitch

REFERENCES

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

[2] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," in *Proc. Conf. Robot Learning (CoRL)*, 2021, pp. 132–145.

[3] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Proc. Robotics: Science and Systems (RSS)*, 2018.

[4] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 3803–3810.

[5] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, Art. no. eabc5986, 2020.

SUPPLEMENTARY MATERIAL

Source code and experimental artifacts are available at: https://github.com/i-oon/Sim-to-Sim_Policy_Transfer_for_Learned-Legged-Locomotion