2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Flip Stunts on Bicycle Robots using Iterative Motion Imitation

Jeonghwan Kim[1,2], Shamel Fahmi[1], Seungeun Rho[1,2], Sehoon Ha[2], and Gabriel Nelson[1]

***Abstract*— This work demonstrates a front-flip on bicycle robots via reinforcement learning, particularly by imitating reference motions that are infeasible and imperfect. To address this, we propose Iterative Motion Imitation (IMI), a method that iteratively imitates trajectories generated by prior policy rollouts. Starting from an initial reference that is kinematically or dynamically infeasible, IMI helps train policies that lead to feasible and agile behaviors. We demonstrate our method on Ultra-Mobility Vehicle (UMV), a bicycle robot that is designed to enable agile behaviors. From a self-colliding table-to-ground flip reference generated by a model-based controller, we are able to train policies that enable ground-to-ground and ground-to-table front-flips. We show that compared to a single-shot motion imitation, IMI results in policies with higher success rates and can transfer robustly to the real world. To our knowledge, this is the first unassisted acrobatic flip behavior on such a platform.**

Fig. 1. UMV executing a front-flip learned via Iterative Motion Imitation.

I. INTRODUCTION

Wheeled and legged robots have recently demonstrated remarkable dynamic capabilities, driven in large part by advances in Reinforcement Learning (RL) and motion imitation [1]–[7]. By tracking demonstrations from motion capture [8], [9], animal locomotion [10], [11], or model-based controllers [7], [12]–[14], robots can learn parkour and agile behaviors. However, if the original motion references are dynamically or kinematically infeasible, the imitation policy may fail to train or lead to unsafe behaviors not suitable for real-world deployment [11]. Blindly tracking these infeasible references may exceed the robot's torque or joint limits or cause self-collisions.

Prior works have explored learning from imperfect demonstrations by reweighting trajectories according to confidence in their optimality [15], filtering out low-quality segments to extrapolate higher-quality behaviors [16], or using adversarial techniques for partial demonstrations [17]. While effective to some extent, these approaches still treat imperfections as liabilities to be corrected offline and as a result, the final policy is still bounded by the static quality of the dataset.

In this work, we introduce Iterative Motion Imitation (IMI), an extension of motion imitation that iteratively transforms infeasible trajectories into feasible and agile behaviors. IMI begins by imitating an initial reference, which may be generated through manual control without safety constraints, trajectory optimization, or hand-crafted drawing, and is often physically infeasible. Once a policy is trained to follow this reference, its rollout is treated as a new reference for the next stage of training. Through this recursive process, imperfections in the reference are progressively removed, which allows the next policies to better satisfy safety-critical constraints. This allows us to use simple reward functions and constraints without relying on extensive reward shaping, and it enables feasible and agile behaviors.

We deploy IMI on Ultra-Mobility Vehicle (UMV) [18], a bicycle robot equipped with a large articulated mass for acrobatic maneuvers as shown in Fig. 1. We focus on a front-flip, as its agility allows us to push the system to its mechanical limits. Beginning with an initial flip-down trajectory with simple imitation rewards, this iterative process allows the policy to generalize to scenarios beyond what the original reference was designed for. We demonstrate this extended agility with ground-to-ground flips and ground-to-table flip-ups. We further confirm the robustness of the learned behaviors through hardware deployment.

To sum up, our contributions are:

- Iterative Motion Imitation (IMI), a method that refines imperfect references into agile policies via iterative imitation, yielding highly agile behaviors from simple tracking rewards.
- Analysis of performance improvement and trajectory for both original flip stunts and adapted flip-up scenarios.
- Hardware demonstration of acrobatic flip stunts on the UMV robot, validating the approach on a platform with unique dynamic stability challenges.

II. RELATED WORK

A. Reinforcement Learning for Agile Locomotion

RL has emerged as a dominant paradigm for synthesizing policies that achieve dynamic locomotion skills beyond the reach of classical control. Recent works have enabled

[1]RAI Institute, Cambridge, MA, USA. [2]Georgia Institute of Technology, Atlanta, GA, USA. This work was done while J. Kim and S. Rho were at the RAI Institute. sfahmi@rai-inst.com, jkim3662@gatech.edu.

quadrupeds to reach record speeds [19]–[21], execute agile parkour [4], [22], [23], and generalize across terrains [24], [25]. These approaches are often "reference-free," relying solely on carefully engineered rewards. While powerful, the reliance on hand-tuned multi-term rewards can be a bottleneck, limiting scalability to diverse and extreme behaviors.

Motion imitation methods alleviate this challenge by leveraging demonstrations such as motion capture [8], [10], video [26], or trajectory optimization [27], [28]. Along with publicly available motion capture datasets [29], [30], motion imitation methods have accelerated the research in humanoid whole-body control [31]–[33]. However, such datasets not only have motion artifacts [34], but the process of retargeting to robot morphologies also leads to artifacts, resulting in physically infeasible references [35], [36].

B. Learning from Imperfect References

In practice, reference trajectories are rarely ideal, often suffering from noise, morphological mismatches, or dynamic infeasibility [11]. Existing methods address this by treating imperfect demonstrations as a static dataset. For instance, trajectories may be weighted or ranked based on scores or demonstration quality [15], [16], [37]. Trajectory optimization can also be used to refine the reference offline [11]. While useful, these techniques cannot refine demonstrations through interaction, and optimization-based methods often require an accurate dynamics model. Moreover, these refinements cannot generate references that exceed the agility of the original demonstrations [38].

An alternative is the adversarial imitation framework [9], which can produce natural behaviors on real robots [10], [39], even from partial demonstrations [17]. Yet, these methods are known to be unstable during training and highly sensitive to hyperparameter tuning [40], [41].

Our work departs from this perspective: rather than coping with reference imperfections, IMI recursively transforms them into feasible and robust guides for policy learning. This is done in the same framework, without re-tuning, and without designing complicated rewards.

C. Learning Bicycle Stunts

Bicycle control has long served as a benchmark for both model-based and learning-based robotics. Linear [42] and nonlinear [43] control strategies were studied on basic tasks such as balancing and target reaching. Early works on RL for bicycle control demonstrated that careful reward shaping can solve basic driving tasks [44].

Subsequent research has leveraged RL to push bicycle capabilities toward driving in complex environments. Hierarchical RL has been used to achieve robust path tracking and balancing on rough, unstructured terrain [45]. Others have integrated their RL frameworks combining path planning, trajectory tracking, and balancing to navigate narrow corridors [46]. This trend extends beyond bicycles, as demonstrated by Baltes et al. [47], who trained a humanoid robot to steer and balance on a commercial scooter.

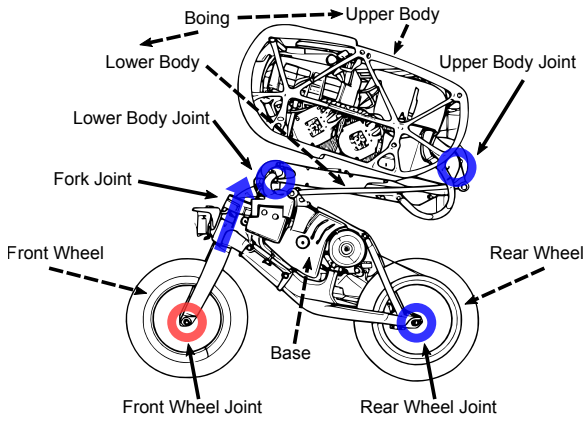

Fig. 2. A 2D overview of the UMV model used in this work.

Beyond driving, dynamic stunts such as jumping have been studied. Learning-based methods have been successfully applied to control ramp jumps by optimizing for flight attitude and landing [48], while model-based approaches have provided complementary insights into the underlying physics using techniques like inverse kinematics and Bayesian optimization [49]. Tan et al. [50] significantly raised the bar of bicycle agility, employing neuro-evolution to successfully learn a diverse repertoire of acrobatic stunts, including bunny hops, wheelies, endos, and pivots.

While these studies showcase a range of dynamic behaviors, most of their results are confined to simulation, lacking validation on physical hardware. Our work builds on this foundation by not only tackling acrobatic flips—a maneuver of greater dynamic complexity than previously demonstrated—but also by successfully deploying the learned policy on a real-world robotic platform.

III. The Ultra-Mobility Vehicle (UMV) Robot

UMV is a custom-built, two-wheeled robot with a bike base comparable in size to a children's bicycle (Fig. 2). A key feature that distinguishes the UMV from a conventional bicycle is a significant articulated mass, referred to as *boing*, mounted atop the bike-base frame.

The model of UMV used in this work consists of five joints and six links. The links are upper-body, lower-body, bike-base, fork, front-wheel, and rear-wheel. The joints are upper-body, lower-body, fork, front-wheel, and rear-wheel. UMV has four actuated joints in total. The upper-body and lower-body joints control the pitch of the upper body, and are the main joints to inject momentum to perform parkour behaviors. The fork joint controls the steering angle (yaw) of the front wheel. The rear-wheel provides driving torque. Consistent with standard bicycle design, the front wheel remains passive and unactuated.

Boing (the upper-body and lower-body links) houses the batteries and joint motors and is connected to the bike-base by two parallel revolute joints (the upper-body and lower-body joints). The axes of these joints are parallel to the wheel axles, allowing the upper-body to pitch forward and backward relative to the bike-base. This mechanism enables

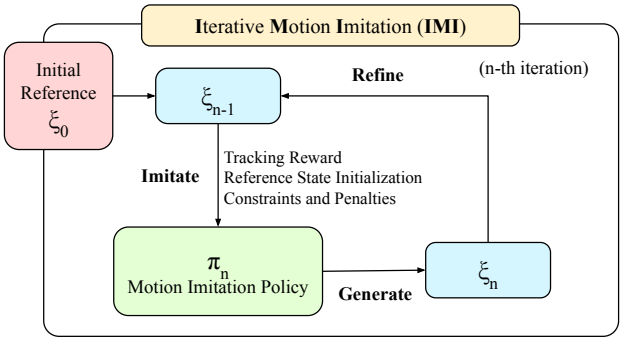

Fig. 3. An overview of IMI. The first iteration starts with an initial reference ξ_0 used to train policy π_1. Then, reference ξ_{n-1} generated by policy π_{n-1} becomes the new more feasible reference used to train a new policy π_n. This loop continues, progressively refining the motion until a high-performance, hardware-deployable policy is achieved.

the robot to dramatically shift its center of mass, a critical capability for generating the angular momentum required for acrobatic maneuvers.

IV. METHOD

A. The Iterative Motion Imitation (IMI)

The goal of IMI, depicted in Fig. 3, is to acquire robust and agile skills from an infeasible trajectory. The framework iteratively refines an initial, imperfect reference until a policy exhibiting the desired feasible behavior is achieved. This process involves three key steps

1) **Imitate:** The policy is trained using constrained RL to track the current reference trajectory.
2) **Generate:** The learned policy is executed in simulation to generate a new, physically plausible trajectory.
3) **Refine:** The generated trajectory is set as the new reference for the next imitation cycle.

The kinematic reference trajectories are used exclusively during training and are progressively improved with each cycle. Experts can also manually trim the trajectory during the refine stage when the desired motion differs from the reference motion, such as adapting a flip-down trajectory to a flat-to-flat flip.

The learned policy is purely reactive, taking only proprioceptive state observations and a phase variable as input to produce joint-level PD targets, which are then tracked by a high-frequency low-level controller.

IMI enhances the two primary guidance mechanisms of motion imitation, dense tracking rewards and Reference State Initialization (RSI) [8], by progressively improving the reference trajectory itself. This iterative refinement makes the tracking reward more informative. Initially, an infeasible reference creates a conflicting objective, forcing the policy to deviate from the reference to satisfy physical constraints. As IMI generates more plausible references, the policy can achieve high tracking fidelity while respecting these constraints. Concurrently, RSI becomes more effective. Initializing from a refined, physically achievable reference places the agent in meaningful states that are closer to high-reward regions, accelerating learning and exploration.

B. Initial Reference Trajectory Generation

The initial flip reference can be generated in various ways, but we opt to use a hand-crafted model-based controller without safety constraints in simulation. The robot starts on top of a table and moves forward using a driving controller to build forward linear momentum. As the robot reaches the edge of the table, a whole-body controller is used to track a certain angular momentum that is tuned to successfully flip down the table.

This orchestrated scenario was chosen intentionally. Starting on top of a table, reaching a certain forward momentum, and tracking and tuning a whole-body controller were engineered to provide an extended flight phase, giving the robot ample time and momentum to complete a 360-degree rotation.

While this procedure yields a reference trajectory that achieves a flip, it is not executable without switching mid-air to a separate landing controller. Moreover, this does not meet our desired agility of flipping from flat ground and also violates the desired safety limits. Thus, the initial reference serves only as a rough demonstration, and our method aims to refine this imperfect reference into an end-to-end deployable policy.

C. Problem Formulation

We formulate the motion imitation task as a Constrained Markov Decision Process (CMDP), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \mathcal{C}, \gamma)$. Here, $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(s_{t+1}|s_t, a_t)$ is the state transition probability, $r(s_t, a_t)$ is the reward function, $\mathcal{C} = \{c_1, ..., c_k\}$ is a set of k constraint functions, and $\gamma \in [0, 1)$ is the discount factor. The objective is to find an optimal policy π^* that maximizes the expected discounted sum of future rewards while satisfying a set of constraints. These constraints, $c_i(s_t, a_t) \leq 0$, represent the physical and operational limits of the robot. The full objective is:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\xi\sim\pi}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \tag{1}$$

$$\text{subject to} \quad c_i(s_t, a_t) \leq 0, \quad \forall i \in \{1, ..., k\}, \forall t \tag{2}$$

We handle the constraints by terminating the episode upon any violation as used in [51]. This strategy effectively transforms the CMDP into a standard MDP, which we solve with RL: with a positive per-step reward design, any policy that violates constraints will inherently achieve a lower expected cumulative reward. Each iteration in IMI is solved using Proximal Policy Optimization (PPO) [52].

D. Control, Observations, and Actions

We use IsaacLab as our training environment [53]. The policy operates at 50 Hz, outputting joint PD targets. These targets are tracked by a low-level PD controller running at 200 Hz in simulation, and at 1 kHz on the real robot.

The proprioceptive observations $o_t \in \mathcal{S}$ are defined as

$$o_t = [q, \dot{q}, \omega, g, \theta, a_{t-1}] \in \mathbb{R}^{18} \tag{3}$$

TABLE I

REWARD STRUCTURE WITH TOLERANCE

Component	Formulation	Weight	Tolerance
	Tracking Rewards		
Base Position	$\exp(-\alpha_{\text{base}}\|p_{\text{base}} - p_{\text{base}}^{\text{ref}}\|^2)$	4.0	0.4
Base Orientation	$\exp(-\alpha_{\text{ang}}\|d_{\text{angle}}(q_{\text{base}}, q_{\text{base}}^{\text{ref}})\|^2)$	20.0	0.8
Joint Positions	$\exp(-\alpha_{\text{joint}}\|q_{\text{joint}} - q_{\text{joint}}^{\text{ref}}\|^2)$	1.0	0.1
	Penalty		
Action Smoothness	$\|a_t - a_{t-1}\|^2$	-1e-5 to -1e-3	—
Fork Velocity	$\dot{q}_{\text{fork}}^2$	-0.001	—
Contact Force	$\|F_{\text{contact}}\|^2$	-1e-6	350 N
Body Joint Limits	$\mathbb{I}(q_{\text{joint}} \notin [q_{\min}, q_{\max}])$	-1.0	—
	Post Tracking Terms		
Default Joint Positions	$\exp(-\alpha_{\text{pd}}\|q_{\text{body joint}}\|^2)$	1.0	—
Jitter Penalty	$\sum \dot{q}_{\text{joint}}^2 + \sum \|\dot{q}_{\text{joint}}\|$	-1e-4	—
Velocity Penalty	$\|v_{\text{base}}\|^2$	-3.0	0.5 m/s

where $q \in \mathbb{R}^3$ are the actuated joint positions, $\dot{q} \in \mathbb{R}^4$ are the actuated joint velocities, $\omega \in \mathbb{R}^3$ is the base angular velocity, $g \in \mathbb{R}^3$ is the projected gravity vector in the robot's base frame, and $a_{t-1} \in \mathbb{R}^4$ is the previous action. The observations exclude the rear wheel's position, as it is an unbounded, continuously rotating joint. The phase variable $\theta \in \mathbb{R}^1$ increases linearly from 0 to 1 over the duration of the reference trajectory. The total episode duration is longer than the reference trajectory, and during this extra time, θ is held at 1, requiring the policy to learn a stable balancing behavior after the touchdown.

The policy outputs a four-dimensional action vector $a_t \in \mathbb{R}^4$, corresponding to each joint's PD targets:

$$a_t = [a_{\text{upper-body}}, a_{\text{lower-body}}, a_{\text{fork}}, a_{\text{rear-wheel}}] \tag{4}$$

Each action is scaled by its corresponding action scale. The action of the rear-wheel joint is defined as a velocity setpoint, while the actions of the upper-body, lower-body, and fork joints are defined as position setpoints. The setpoints are tracked by a PD controller with desired joint torques τ

$$\tau = k_p(q_{\text{des}} - q) + k_d(\dot{q}_{\text{des}} - \dot{q}) \tag{5}$$

where q_{des} and $\dot{q}_{\text{des}}$ are the desired position and velocity setpoints for every joint.

E. Rewards

The reward is a weighted sum of tracking and penalty terms, detailed in Table I. The tracking rewards encourage the policy to follow the reference motion, and penalties discourage behaviors that are unsuitable for hardware execution. We do not track the fork and rear-wheel joint positions to grant the policy more freedom to discover strategies for balancing and momentum generation. We clip the tracking errors to certain tolerances so that the maximum reward can be achieved without the need to track the exact reference. After successfully tracking the reference (i.e., when the phase variable $\theta = 1$), the *post-tracking terms* are triggered to promote a still, balanced state by rewarding a default joint posture while penalizing base velocity and joint jitter.

F. Constraints and Terminations

To ensure the safe deployment of learned policies on hardware, we enforce critical physical limits as constraints. Similar to [51], we implement these constraints through termination conditions. Unlike approaches that use soft terminations to allow a policy to learn recovery behaviors, we employ hard terminations. This is because a constraint violation in our task represents an irrecoverable failure state. For instance, exceeding the current limit triggers a protective hardware shutdown, while an excessive touchdown velocity is an instantaneous event that can cause catastrophic hardware fracture. Based on that, an episode is terminated if any of the following conditions are met:

Touchdown Velocity: The landing vertical velocity of the wheels exceeds a certain threshold.

Mechanical Power: The total mechanical power of the joints exceeds a certain threshold $\sum \tau\dot{q}$. This serves as a proxy for the peak current limit.

Joint Position: The bounds of the joint position limits are reached. This prevents self-collisions and singularities.

Motor Torques: The motor torques exceed their torque limit.

Wheel Velocity: The wheel velocity exceeds a safety limit.

Early Termination: The robot's bike-base position or orientation deviates too much from the reference trajectory.

Ground Collision: The upper-body, lower-body, or bike-base bodies collide with the ground.

To encourage early exploration during training, we introduce these hard constraints via a curriculum. The training process begins with relaxed constraint boundaries that are gradually tightened as the policy improves. This approach allows the agent to first learn the fundamental task before refining its behavior to operate within the strict physical limits, thereby preventing the learning process from stagnating.

Constraints can also be gradually introduced during different IMI iterations. For instance, we only enforce the joint position limits at the first iteration, leaving other constraints to be tightened in future iterations. For further iterations, the joint position limits were enforced from the beginning of training, and the rest of the terminations are applied as a curriculum.

G. Sim2Real and Policy Training

Reference State Initialization. We use RSI to expose the policy to critical states early in training [8]. 50% of the episodes begin from an initial state and the remaining 50% are initialized at random states sampled from the reference trajectory. We also ignore the last 10% of the data during touchdown.

Domain Randomization. To bridge the sim-to-real gap, we apply domain randomization to various physical and system parameters during all stages of IMI. Specifically, we randomize physical properties such as mass, friction, and motor strength, as well as control-related parameters including actuator gains and actuation delay. We also introduce observation noise to joint positions, velocities, angular velocity, and projected gravity, and further apply external disturbances to the body velocity.

Furthermore, to handle varied contact scenarios such as premature ground impact, we add terrain randomization. In 50% of training episodes, we introduce a step obstacle with a

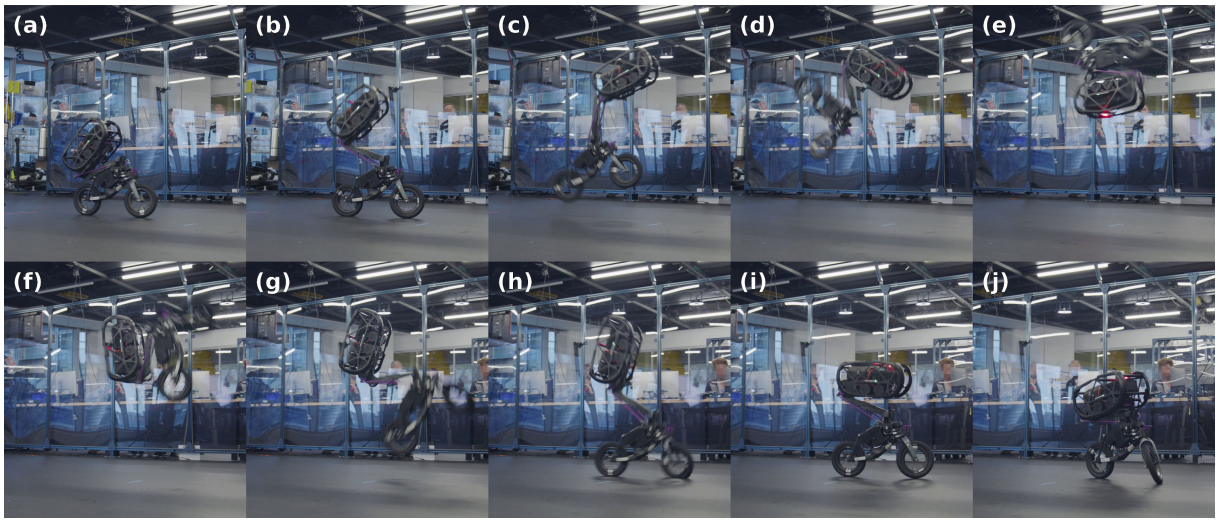

Fig. 4. Overlayed screenshots of UMV performing a front flip.

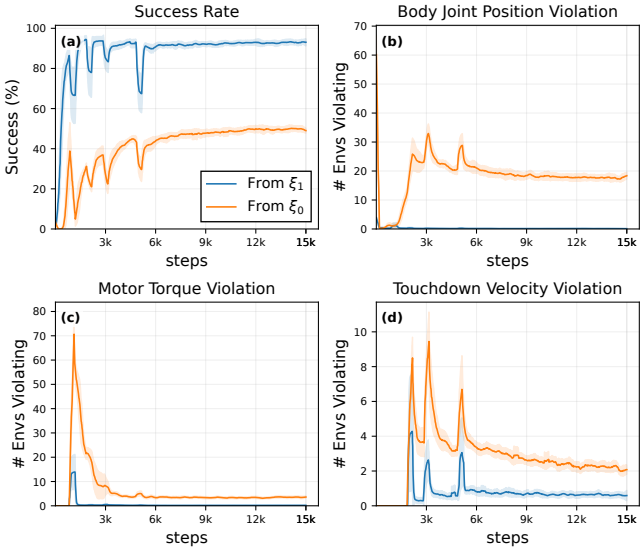

Fig. 5. Comparison of training from scratch (ξ_0) versus using refined reference (ξ_1). Each run uses 3 seeds. The solid line is the mean and the shaded area is the standard deviation.

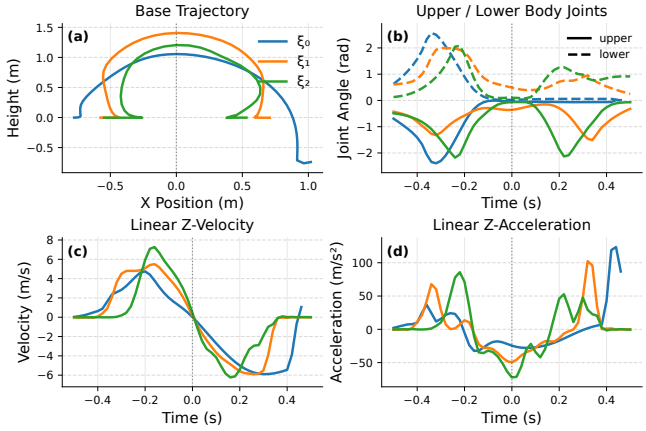

Fig. 6. The evolution of trajectories ξ over two iterations. The time is aligned at the peak height ($t = 0$) and the interval is ± 0.5 s around that peak. (a) bike-base X-Z position, (b) lower-body and upper-body joint angles, (c) bike-base vertical velocity, and (d) bike-base vertical acceleration.

height uniformly sampled from the range [0 m, 0.2 m]. Collectively, these randomizations encourage the policy to learn behaviors that are robust to modeling errors and unmodeled dynamics.

Network Architecture. The actor and critic networks are implemented as Multi-Layer Perceptrons (MLPs) with ELU activation functions and three hidden layers of size [512, 256, 128] and [512, 512, 256], respectively. In addition to the actor's observations, the critic observes the bike-base's linear velocity and global position as privileged information during training. The policies are trained for 15,000 iterations.

V. Results

We evaluate IMI through simulations and experiments on UMV, and show that it can transform an initially infeasible reference trajectory into a robust policy that transfers to the real world as shown in Fig. 4. We first detail the importance of iterative imitation on the sim-to-real transfer of our front-flip policy. Then, we conduct an ablation study to analyze the effect of iterative refinement, showing its ability to adapt to a harder maneuver such as a front-flip onto a table.

A. Refinement of Flip Trajectories

We train IMI with two iterations. The first iteration trains a policy π_1 with the initial trajectory generated by the model-based controller (i.e., ξ_0). The second iteration trains a policy π_2 with the trajectory generated from the first iteration (i.e., ξ_1). Using the policy from the second iteration π_2, we generate a third trajectory for comparison (i.e., ξ_2).

Since the initial reference ξ_0 starts with the robot on a table, we translated the base pose so that the robot is initialized on the ground. For all references in the iteration, we trim the trajectory from the moment the robot acquires forward velocity until ground touchdown.

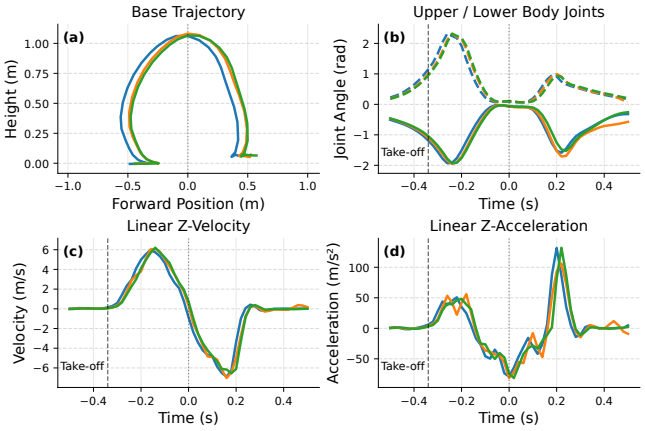

Fig. 7. Three hardware flip experiments of UMV. The time is aligned at the peak height ($t = 0$) and the interval is ± 0.5 s around that peak. (a) bike-base X-Z position, (b) lower-body and upper-body joint angles, (c) bike-base vertical velocity, and (d) bike-base vertical acceleration. The vertical dashed line indicates the time at take-off.

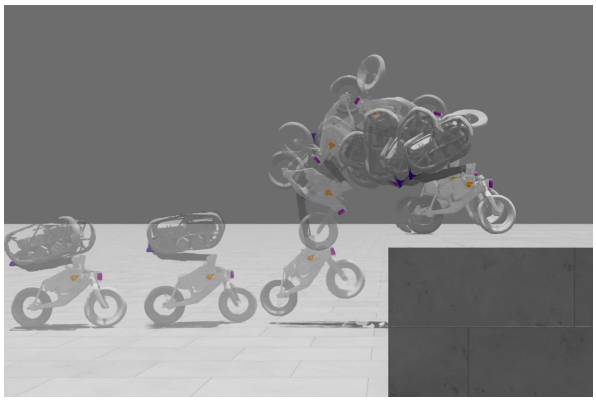

Fig. 8. Overlay of UMV flipping up a 70 cm box.

Fig. 6 illustrates the evolution of these trajectories ξ. The initial trajectory ξ_0 exhibits undesired behavior after the peak height: as shown in Fig. 6(b), the robot reaches its joint limits, leading to a self-collision. The landing is also unregulated, with the horizontal impact velocity and acceleration (Fig. 6(c,d)) being the largest among the three iterations.

In contrast, ξ_2 demonstrates improved landing characteristics over ξ_1. Its peak horizontal velocity and acceleration at touchdown are reduced (Fig. 6(c,d)), leading to a smoother and less damaging impact.

B. Hardware Experiments

We deployed policy π_2 on two different UMVs. Figure 4 shows the hardware experiment from one robot, and Fig. 7 shows the outcome of three different runs on another.

The robot successfully performed multiple front flips on multiple different hardware, demonstrating the policy's robustness and the effectiveness of our methodology. From Fig. 7(c,d), we observe that this agile maneuver is completed within 0.8 s of flight time, achieving a full 360-degree rotation while providing sufficient time to prepare for a controlled smooth landing with low vertical velocity and acceleration. From Fig. 7(b), we observe that the robot extends its joints before take-off, tucks its boing to gain angular momentum and untucks again for smooth landing, and then transitions to the default joint positions, all without reaching any joint limits.

C. Ablation: Iterative vs. Single-Shot Imitation

Here, we compare two settings: IMI, which iteratively refines motion imitation sequences, against a single-iteration motion imitation baseline. To do so, we train two different policies with identical settings except that the first policy is trained with the initial reference ξ_0 while the second policy is trained with the reference from the previous IMI iteration ξ_1. In other words, we compare training from a refined reference trajectory ξ_1 against a baseline trained using an initial reference trajectory ξ_0.

Figure 5 summarizes the comparison between the two settings, averaged over three runs with different seeds. Fig. 5(a) shows the success rate, where success is defined as episodes that end in time-outs. Fig. 5(b,c) reports the number of environments (out of 4096) terminated at each step due to body joint limit and motor torque violations, respectively. Fig. 5(d) shows terminations caused by touchdown velocity limit. Sudden drops in the success rate are moments when the termination curriculum was enforced.

Based on Fig. 5, with IMI, the training converges substantially faster compared to the training from the initial reference. Furthermore, IMI achieved a success rate of 93 %, whereas training from the initial reference reaches only 49 %. The most prominent cause of failure was due to joint limits (Fig. 5(b)). For a successful flip maneuver, the robot needs to tuck its boing as close to its limit as possible to achieve maximum angular velocity. Then, it needs to quickly untuck to prepare for a smooth landing. Doing this complicated maneuver in a limited flight time is prone to joint position and motor torque violations without rich reference guidance. The refined trajectory ξ_1 already anticipates the need to open its body in preparation for a safe landing compared to ξ_0, where smooth landing was not taken into consideration. Another prominent cause of failure was due to touch down velocity (Fig. 5(c)), which occurs due to limited time for extending posture for smooth landing.

D. Task Adaptation: Flip Up a Box

To showcase IMI's effectiveness beyond safety-focused refinement, we task the robot to do a harder maneuver of flipping up a box. Starting from the refined reference of ξ_1, we train a separate policy π_2' with a modified terrain. This training includes a 70 cm (as shown in Fig. 8) or a 26 cm box in front of the robot. As shown in Fig. 8, the robot was able to successfully perform a flip-up stunt over a 70 cm box using IMI. Figure 9 (a,b) shows how the robot deviates from its reference depending on the task difficulty.

For quantitative analysis, we perform a third IMI iteration, where the policy is trained to execute a flip-up by imitating

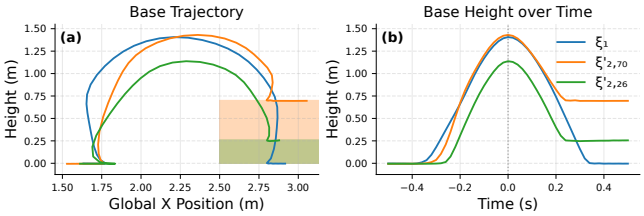

Fig. 9. Flipping up 70 cm ($\xi'_{2,70}$) and 26 cm ($\xi'_{2,26}$) boxes, each learned by imitating the reference trajectory ξ_1. (a) bike-base trajectory in the global frame, with the shaded region indicating the table position. (b) bike-base vertical trajectory aligned at peak height.

TABLE II

FLIP-UP PERFORMANCE FROM ξ_0, ξ_1, AND ξ_2.

	Success Rate		Average Return	
ξ	70cm	26cm	70cm	26cm
ξ_2	**95.5%**	**97.2%**	**81.8**	**94.1**
ξ_1	91.9%	92.4%	80.5	93.0
ξ_0	0.0%	85.1%	33.6	57.0

a trajectory generated from the second-iteration of a ground-to-ground flip ξ_2. As shown in the first row of Table II, this three-iteration process enhances flip-up performance in terms of both success rate measured from the start and average return. Furthermore, the 70 cm flip-up stunt could not be achieved by directly imitating the initial reference ξ_0 (i.e., non-iterative motion imitation), which lacks sufficient angular momentum and height clearance.

VI. LIMITATIONS AND FUTURE WORK

While successfully demonstrating robust flipping maneuvers, we acknowledge several limitations of IMI. The iterative refinement process relies on human judgment to decide when to initiate additional iterations. The operator qualitatively assesses the simulated trajectory's performance and feasibility, and if the outcome is unsatisfactory, a new iteration is triggered. Furthermore, the learned policy is specialized to a single skill. While robust within that skill, the resulting controller cannot perform other acrobatic maneuvers without training a separate policy, requiring the re-iteration of the process to guide to a different direction.

These limitations suggest promising directions for future work. To reduce reliance on human assessment, one could design automated stopping criteria that detect when the performance improvements plateau (e.g., landing stability or energy efficiency). Moreover, an ambitious extension would be to replace the discrete iteration loop with a continuous refinement process, potentially leveraging Reinforcement Learning from Human Feedback (RLHF) [54] to incorporate the operator's preferences in a more structured and scalable manner. To generalize beyond a single skill, we plan to extend IMI to learn command-conditioned policies that learn from a library of refined skills, as well as investigate ways to automatically generate initial reference trajectories from high-level task objectives.

VII. CONCLUSION

In this paper, we introduced Iterative Motion Imitation (IMI), a reinforcement learning framework that learns agile and physically robust robotic behaviors by iteratively refining an imperfect initial reference. We demonstrated that by iteratively imitating trajectories generated by its own predecessor policies, IMI effectively transforms a dynamically infeasible motion into a high-performance policy that respects hardware limits. Our key insight is that this recursive process naturally amplifies the effectiveness of standard motion imitation techniques such as Reference State Initialization (RSI) and reward shaping, enabling robust learning without complex reward engineering.

We validated our approach on the UMV, a bicycle robot. The IMI-trained policy successfully executed unassisted front flips on hardware, marking the first demonstration of such acrobatic stunts on this platform.

VIII. ACKNOWLEDGEMENT

We thank Ardalan Tajbakhsh for providing the initial trajectories from the model-based controller and the UMV team for their hardware support.

REFERENCES

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

[2] J. Lee, M. Bjelonic, A. Reske, L. Wellhausen, T. Miki, and M. Hutter, "Learning robust autonomous navigation and locomotion for wheeled-legged robots," *Science Robotics*, vol. 9, no. 89, p. eadi9641, 2024.

[3] E. Vollenweider, M. Bjelonic, V. Klemm, N. Rudin, J. Lee, and M. Hutter, "Advanced skills through multiple adversarial motion priors in reinforcement learning," *arXiv preprint arXiv:2203.14912*, 2022.

[4] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 11 443–11 450.

[5] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," *arXiv preprint arXiv:2309.05665*, 2023.

[6] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," *Science Robotics*, vol. 9, no. 88, p. eadi7566, 2024.

[7] A. Miller, S. Fahmi, M. Chignoli, and S. Kim, "Reinforcement learning for legged robots: Motion imitation from model-based optimal control," *arXiv preprint arXiv:2305.10989*, 2023.

[8] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[9] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (ToG)*, vol. 40, no. 4, pp. 1–20, 2021.

[10] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems (RSS)*, 2020.

[11] T. Yoon, D. Kang, S. Kim, J. Cheng, M. Ahn, S. Coros, and S. Choi, "Spatio-temporal motion retargeting for quadruped robots," *IEEE Transactions on Robotics*, 2025.

[12] D. Youm, H. Jung, H. Kim, J. Hwangbo, H.-W. Park, and S. Ha, "Imitating and finetuning model predictive control for robust and symmetric quadrupedal locomotion," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7799–7806, 2023.

[13] D. Kang, J. Cheng, M. Zamora, F. Zargarbashi, and S. Coros, "Rl+ model-based control: Using on-demand optimal control to learn versatile legged locomotion," *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6619–6626, 2023.

[14] F. Jenelten, J. He, F. Farshidian, and M. Hutter, "Dtc: Deep tracking control," *Science Robotics*, vol. 9, no. 86, p. eadh5401, 2024.

[15] Y.-H. Wu, N. Charoenphakdee, H. Bao, V. Tangkaratt, and M. Sugiyama, "Imitation learning from imperfect demonstration," in *International Conference on Machine Learning (ICML)*, 2019, pp. 6818–6827.

[16] D. S. Brown, W. Goo, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International Conference on Machine Learning (ICML)*, 2019, pp. 783–792.

[17] C. Li, M. Vlastelica, S. Blaes, J. Frey, F. Grimminger, and G. Martius, "Learning agile skills via adversarial imitation of rough partial demonstrations," in *Conference on Robot Learning*. PMLR, 2023, pp. 342–352.

[18] B. Bokser, D. Gonzalez, S. Singh, A. Preston, A. Bahner, A. Wollschläger, A. Ilvonen, A. Eckert-Erdheim, A. Khadke, B. Hammoud *et al.*, "System design of the ultra mobility vehicle: A driving, balancing, and jumping bicycle robot," *arXiv preprint arXiv:2602.22118*, 2026.

[19] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control," *The International Journal of Robotics Research*, vol. 44, no. 5, pp. 840–888, 2025.

[20] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.

[21] A. Miller, F. Yu, M. Brauckmann, and F. Farshidian, "High-performance reinforcement learning on spot: Optimizing simulation parameters with distributional measures," *arXiv preprint arXiv:2504.17857*, 2025.

[22] H. Kim, H. Oh, J. Park, Y. Kim, D. Youm, M. Jung, M. Lee, and J. Hwangbo, "High-speed control and navigation for quadrupedal robots on complex and discrete terrain," *Science Robotics*, vol. 10, no. 102, p. eads6192, 2025.

[23] S. Rho, K. Garg, M. Byrd, and S. Ha, "Unsupervised skill discovery as exploration for learning agile locomotion," in *Conference on Robot Learning*. PMLR, 2025, pp. 2678–2694.

[24] 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.

[25] J. He, C. Zhang, F. Jenelten, R. Grandia, M. Bächer, and M. Hutter, "Attention-based map encoding for learning generalized legged locomotion," *Science Robotics*, vol. 10, no. 105, p. eadv3604, 2025.

[26] X. B. Peng, A. Kanazawa, J. Malik, P. Abbeel, and S. Levine, "Sfv: Reinforcement learning of physical skills from videos," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 6, pp. 1–14, 2018.

[27] Y. Fuchioka, Z. Xie, and M. Van de Panne, "Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," *arXiv preprint arXiv:2210.01247*, 2022.

[28] F. Liu, Z. Gu, Y. Cai, Z. Zhou, H. Jung, J. Jang, S. Zhao, S. Ha, Y. Chen, D. Xu *et al.*, "Opt2skill: Imitating dynamically-feasible whole-body trajectories for versatile humanoid loco-manipulation," *arXiv preprint arXiv:2409.20514*, 2024.

[29] N. Mahmood, N. Ghorbani, N. F. Troje, G. Pons-Moll, and M. J. Black, "Amass: Archive of motion capture as surface shapes," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 5442–5451.

[30] F. G. Harvey, M. Yurick, D. Nowrouzezahrai, and C. Pal, "Robust motion in-betweening," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 60–1, 2020.

[31] T. He, J. Gao, W. Xiao, Y. Zhang, Z. Wang, J. Wang, Z. Luo, G. He, N. Sobanbab, C. Pan *et al.*, "Asap: Aligning simulation and real-world physics for learning agile humanoid whole-body skills," *arXiv preprint arXiv:2502.01143*, 2025.

[32] Z. Chen, M. Ji, X. Cheng, X. Peng, X. B. Peng, and X. Wang, "Gmt: General motion tracking for humanoid whole-body control," *arXiv preprint arXiv:2506.14770*, 2025.

[33] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi, "Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning," *arXiv preprint arXiv:2406.08858*, 2024.

[34] Z. Luo, J. Cao, A. W. Winkler, K. Kitani, and W. Xu, "Perpetual humanoid control for real-time simulated avatars," in *International Conference on Computer Vision (ICCV)*, 2023.

[35] Y. Ze, Z. Chen, J. P. Araújo, Z. ang Cao, X. B. Peng, J. Wu, and C. K. Liu, "Twist: Teleoperated whole-body imitation system," *arXiv preprint arXiv:2505.02833*, 2025.

[36] W. Xie, J. Han, J. Zheng, H. Li, X. Liu, J. Shi, W. Zhang, C. Bai, and X. Li, "Kungfubot: Physics-based humanoid whole-body control for learning highly-dynamic skills," *arXiv preprint arXiv:2506.12851*, 2025.

[37] Z. Cao, E. Biyik, Y. Wu, and D. Sadigh, "Learning from imperfect demonstrations from agents with varying dynamics," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 10 736–10 743.

[38] N. R. Arachchige, Z. Chen, W. Jung, W. C. Shin, R. Bansal, P. Barroso, Y. H. He, Y. C. Lin, B. Joffe, S. Kousik, and D. Xu, "Sail: Faster-than-demonstration execution of imitation learning policies," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2025. [Online]. Available: https://arxiv.org/abs/2506.11948

[39] J. Wu, G. Xin, C. Qi, and Y. Xue, "Learning robust and agile legged locomotion using adversarial motion priors," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 4975–4982, 2023.

[40] X. B. Peng, A. Kanazawa, S. Toyer, P. Abbeel, and S. Levine, "Variational discriminator bottleneck: Improving imitation learning, inverse rl, and gans by constraining information flow," *arXiv preprint arXiv:1810.00821*, 2018.

[41] C. Li, M. Hutter, and A. Krause, "Feature-based vs. gan-based learning from demonstrations: When and why," *arXiv preprint arXiv:2507.05906*, 2025.

[42] J. Xiong, R. Yu, and C. Liu, "Steering control and stability analysis for an autonomous bicycle: part ii experiments and a modified linear control law," *Nonlinear Dynamics*, vol. 112, no. 5, pp. 3107–3132, 2024.

[43] L. Cui, S. Wang, J. Lai, X. Chen, S. Yang, Z. Zhang, and Z.-P. Jiang, "Nonlinear balance control of an unmanned bicycle: Design and experiments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7279–7284.

[44] J. Randløv and P. Alstrøm, "Learning to drive a bicycle using reinforcement learning and shaping." in *ICML*, vol. 98, 1998, pp. 463–471.

[45] X. Zhu, X. Zheng, Y. Deng, Z. Chen, B. Liang, and Y. Liu, "Deep reinforcement learning-based control of bicycle robots on rough terrain," in *2023 9th International Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2023, pp. 103–108.

[46] Q. Zheng, Y. Tian, Y. Deng, X. Zhu, Z. Chen, and B. Liang, "Reinforcement learning-based control of single-track two-wheeled robots in narrow terrain," in *Actuators*, vol. 12, no. 3. MDPI, 2023, p. 109.

[47] J. Baltes, G. Christmann, and S. Saeedvand, "A deep reinforcement learning algorithm to control a two-wheeled scooter with a humanoid robot," *Engineering Applications of Artificial Intelligence*, vol. 126, p. 106941, 2023.

[48] Q. Zheng, D. Wang, Z. Chen, Y. Sun, and B. Liang, "Continuous reinforcement learning based ramp jump control for single-track two-wheeled robots," *Transactions of the Institute of Measurement and Control*, vol. 44, no. 4, pp. 892–904, 2022.

[49] B. Wang, F. Jing, Y. Deng, Z. Chen, and B. Liang, "Bayesian optimization-based ideal landing planning for ramp jump of single-track two-wheeled robots," in *2024 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2024, pp. 396–401.

[50] J. Tan, Y. Gu, C. K. Liu, and G. Turk, "Learning bicycle stunts," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, pp. 1–12, 2014.

[51] E. Chane-Sane, P.-A. Leziart, T. Flayols, O. Stasse, P. Souères, and N. Mansard, "Cat: Constraints as terminations for legged locomotion reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

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

[53] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023, DOI:10.1109/LRA.2023.3270034.

[54] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.