PPF: Pre-training and Preservative Fine-tuning of Humanoid Locomotion via Model-Assumption-based Regularization

Hyunyoung Jung*,1, Zhaoyuan Gu*,1, Ye Zhao[1], Hae-Won Park[2] and Sehoon Ha[1]

***Abstract*— Humanoid locomotion is a challenging task due to its inherent complexity and high-dimensional dynamics, as well as the need to adapt to diverse and unpredictable environments. In this work, we introduce a novel learning framework for effectively training a humanoid locomotion policy that imitates the behavior of a model-based controller while extending its capabilities to handle more complex locomotion tasks, such as more challenging terrain and higher velocity commands. Our framework consists of three key components: pre-training through imitation of the model-based controller, fine-tuning via reinforcement learning, and model-assumption-based regularization (MAR) during fine-tuning. In particular, MAR aligns the policy with actions from the model-based controller only in states where the model assumption holds to prevent catastrophic forgetting. We evaluate the proposed framework through comprehensive simulation tests and hardware experiments on a full-size humanoid robot, Digit, demonstrating a forward speed of 1.5 m/s and robust locomotion across diverse terrains, including slippery, sloped, uneven, and sandy terrains.**

I. Introduction

Humanoid robots hold unique potential to operate seamlessly in human-centric environments. To realize this, they are expected to function reliably across a wide range of indoor and outdoor settings, which requires advanced locomotion capabilities. However, achieving robust locomotion in unstructured environments remains challenging due to hybrid dynamics involving complex contact and the high dimensionality of bipedal systems. Previous works have approached this problem using model-based methods that compute foot placement using simplified models [1], [2]. To enhance constraint handling and stability, these simplified models are often integrated with optimization-based methods [3], [4], [5], [6] in model predictive control (MPC). However, such approaches with simplified models can be less adaptable in complex environments. An alternative approach uses more accurate full-order models but at the cost of computation speed [7] or accuracy [8]. In addition, the hybrid nature of bipedal locomotion dynamics complicates the optimization formulation, which is often restricted to fixed contact sequences.

More recently, bipedal locomotion has been tackled by learning-based approaches that train control policies through gradient descent on data-driven objective functions [9], [10], [11], [12]. As demonstrated in quadrupedal locomotion [13], [14], [15], [16], [17], these learning-based approaches can enhance robustness and agility. However, they often require extensive reward engineering, and their lack of interpretability and long training times make interactive tuning difficult. To address this, approaches that combine model-based and learning-based approaches have also been widely explored [18], [19], [20]. This hybrid strategy has also been shown to be effective for humanoids, particularly in applications involving contact planning [21], [22]. However, they often require a model-based controller to run in the backend, which can be expensive at runtime. A similar challenge was encountered earlier in the quadrupedal context and was tackled by entirely substituting the model-based controller with a behavior-cloned neural network policy [23], [24]. Especially, Youm et al. [23] demonstrated that a robust control policy can be efficiently trained using a two-stage learning process, where the first stage involves imitation of a model-based controller (MBC), followed by reinforcement learning (RL) to fine-tune performance.

Fig. 1. Our Pre-training and Preservative Fine-tuning (PPF) framework achieves a forward velocity of 1.5 m/s while successfully traversing a whiteboard covered with poppy seeds or olive oil, as well as diverse outdoor terrains, including hills, uneven surfaces, and sand.

Motivated by this two-stage learning process, we seek to train a robust humanoid locomotion policy that achieves higher speed and more robust locomotion while employing symmetric and periodic gaits of MBC. However, unlike quadrupeds, humanoid has more unstable dynamics where its center of mass can easily shift out of its supporting polygon. When RL is applied for fine-tuning, the policy often suffers from catastrophic forgetting [25], overfitting to the task or dynamics it is trained on and deviating significantly from the motions learned during pre-training. This leads to a loss of the originally imitated behavior and degraded performance in the target domain.

To mitigate this problem, we introduce a novel learning framework, PPF: Pre-training and Preservative Fine-tuning of a humanoid control policy via model-assumption-based regularization. Unlike the previous two-stage learning ap-

*These two authors contribute equally to work.

[1]Georgia Institute of Technology, Atlanta, GA, 30308, USA hjung331@gatech.edu, zgu78@gatech.edu, ye.zhao@me.gatech.edu, sehoonha@gatech.edu

[2]Korea Advanced Institute of Science and Technology, Yuseong-gu, Daejeon, 34141, Republic of Korea haewonpark@kaist.ac.kr

This paper has been accepted for publication in IEEE Robotics and Automation Letters.

proach [23], our method introduces an adaptive regularization term based on the model assumption violation in the fine-tuning stage. More specifically, our method regularizes the policy to match the model-based controller's actions if the robot's state aligns with the assumptions of the underlying model. This approach is especially useful when the fine-tuning task extends beyond the capabilities of the model-based controller. For example, the ALIP model [26] assumes a constant body height and zero vertical velocity. When these assumptions are violated, such as during high-speed motion or traversal over rough terrain, the regularization weight is reduced, allowing the policy to improve beyond the limitations of the model.

We demonstrate PPF on a full-sized humanoid robot, Digit, in both simulation and real-world environments. In the simulation, our method can traverse both uneven and sloped terrains while achieving the highest tracking performance compared to the baseline methods. On hardware, our method can achieve diverse locomotion tasks including fast forward walking and robust walking across various indoor and outdoor terrains. Specifically, PPF achieves a forward walking speed of 1.5 m/s while maintaining robustness on a whiteboard covered with poppy seeds or olive oil, as well as on outdoor sloped, uneven, and deformable sandy terrains.

II. Related Work

A. Model-based Control for Humanoid Locomotion

Humanoid locomotion often relies on optimization algorithms that use mathematical models to capture the robot's essential dynamics. One of the most widely used models for bipedal locomotion is the linear inverted pendulum model (LIPM) [1], which provides analytical solutions for the foot placement to achieve a desired CoM trajectory [2], [6]. However, this simplified model fails to capture detailed joint-level behaviors, often resulting in conservative or infeasible full-body motions. To address these limitations, researchers have proposed more accurate models considering the inertia of the robot, such as single rigid body models [3] and centroidal dynamics [27], [28]. These inertial-informed models enable the online planning of the contact location, force, and centroidal states and have achieved agile locomotion skills such as running [29] and jumping [30]. Recently, with advancements in computational power, it has become feasible to plan with kino-dynamics [7] or even full-body dynamics model [8], [31]. Despite their impressive performance, model-based locomotion methods typically require accurate robot dynamics, well-informed environment setup, and a detailed specification of the gait sequence to obtain this performance, which causes significant manual effort.

B. Learning-based Approach for Humanoid Locomotion

In recent years, there have been significant advancements in legged locomotion due to the emergence of deep reinforcement learning algorithms and the power of massively parallel simulation environments. These learning frameworks initially demonstrated impressive performance in quadrupedal locomotion by optimizing policies based on carefully designed rewards [13], [14], [15], imitating reference motions [17], or incorporating model-based controllers [18], [19], [20]. In particular, incorporating model-based methods enables learning-based methods to acquire periodic and symmetric gaits in a more interpretable way, mitigating the tedious reward shaping. The similar success on quadruped has been replicated on humanoid locomotion via learning-based frameworks. Some studies proposed novel architectures [9], [10] to derive actions from contextual information embedded in observation histories. Others explored the use of demonstration data to streamline the training and mitigate the reward engineering bottleneck by leveraging model-based methods [32], [33] or state-only motion capture data [10], [34].

C. Domain Transfer for Learning-Based Legged Locomotion

While model-based approaches typically design their control frameworks directly in the target domain (i.e., the real world) or within high-fidelity simulators [35], [36], learning-based approaches usually utilize the simulation data to facilitate the data generation process. However, this often struggles with the well-known sim-to-real gap, leading to overfitting to the training simulator. Therefore, transferring a trained policy to different domains remains a key challenge in learning-based methods. In the context of legged locomotion, this problem is handled in two different ways. The first approach is by utilizing target domain data [37], [38], [39], [40], [41], where they directly train the policy in the target domain either through fine-tuning or end-to-end learning. However, this approach requires expensive data collection, especially for humanoids. The other way utilizes the scalability of simulations [9], [11], [42], where they extensively randomize the physical parameters during training to enable zero-shot deployment in the target domain. While these methods perform well in real-world scenarios, questions remain about their sample efficiency and the integration of existing controllers. In this work, we aim to address the domain transfer challenge by learning the periodic and cyclic gaits of the model-based controller without catastrophic forgetting.

III. Background: Model-based Controller

Our framework begins with a given model-based controller (MBC), which is distilled into a learnable neural network and fine-tuned using reinforcement learning (RL). In our implementation, we adopt the humanoid locomotion controller in Shamsah et al [43], which consists of a foot placement controller based on the angular-momentum linear inverted pendulum (ALIP) model [26] and a passivity-based whole-body inverse dynamics controller [44].

Foot placement controller. The work of [43] captures the dynamics of a bipedal robot using the reduced-order ALIP model that consists of a center of mass (CoM) and its connecting massless telescopic legs.

The ALIP model from [26] uses angular momentum about the stance foot as the contact point. Assuming constant CoM height z, the ALIP model has the following dynamics:

$$\dot{\boldsymbol{x}}_c = \frac{\boldsymbol{L}}{mz}, \dot{\boldsymbol{L}} = mg\boldsymbol{x}_c + \boldsymbol{u}_a, \tag{1}$$

where $\boldsymbol{x}_c$ is the horizontal CoM position in a frame attached to the contact point, $\boldsymbol{L}$ is angular momentum about the contact point, and $\boldsymbol{u}_a$ is the ankle torque. m is the mass of the robot, and g is the gravitational acceleration. This model assumes the constant height $\bar{z}$, and zero velocity and zero acceleration in the vertical direction:

$$z = \bar{z},\ \dot{z} = 0,\ \ddot{z} = 0, \tag{2}$$

with $\bar{z} = 1.01$ m for Digit. In this model, the desired foot placement with respect to the CoM position $\boldsymbol{x}_c$ can be computed by the one-step-ahead prediction:

$$\boldsymbol{x}^{\text{des}} = \frac{\boldsymbol{L}\cosh(\omega T)/mz - \boldsymbol{v}_c^{\text{des}}}{\omega \sinh(\omega T)}, \tag{3}$$

where T is the step duration of one walking step, ω is the natural frequency given by $\sqrt{g/\bar{z}}$, and $\boldsymbol{v}_c^{\text{des}}$ is the desired CoM velocity.

Passivity-based controller. Given a desired foot placement, we solve full-body inverse kinematics to generate smooth joint trajectories $\boldsymbol{q}^{\text{des}}, \dot{\boldsymbol{q}}^{\text{des}}, \ddot{\boldsymbol{q}}^{\text{des}}$, where $\boldsymbol{q}$ represents the full joint state of the robot, including both unactuated floating-base joints and actuated motor joints.

The inverse dynamics controller solves linearized dynamics $M\ddot{\boldsymbol{q}}^{\text{des}} - J^T\boldsymbol{\lambda} - S^T\boldsymbol{\tau} = -C\dot{\boldsymbol{q}} - G$, where M is mass matrix, J is contact jacobian, S is selection matrix, C is Coriolis and centrifugal term, and G is the gravity vector. $\boldsymbol{\tau}$ is the joint torque, and $\boldsymbol{\lambda}$ is the contact force. The inverse dynamics controller separates the linearized dynamics into actuated and unactuated parts to eliminate the contact force and solve for the desired joint torque. A passivity-based feedback controller is applied for stabilization [44] as feedback.

IV. Background: Imitating and Finetuning Model-based Controller

Our framework follows the approach of pretraining the policy using a model-based controller and fine-tuning it using reinforcement learning, as proposed by Youm et al. [23].

Imitation of the model-based controller. In the first stage, the expert model-based controller is distilled to the learnable neural network using behavior cloning with Dataset Aggregation (DAgger) [45]. The DAgger loss is given by the squared Euclidean norm between the actions:

$$L_{DAgger} = \sum_{(\boldsymbol{s},\boldsymbol{a}^{\boldsymbol{E}})\in\mathscr{D}} \left\|\boldsymbol{a}^E - \mu_\theta(\boldsymbol{s})\right\|^2, \tag{4}$$

where $\mathscr{D}$ is the aggregated data buffer, $\boldsymbol{s}$ and $\boldsymbol{a}^E$ denote the sampled robot state and the corresponding action from the expert model-based controller, where E stands for the "expert". μ_θ denotes the neural network policy trained via supervised learning. While deterministic here, it later serves as the mean of the Gaussian policy during RL fine-tuning. This imitation gives us a pre-trained policy π that behaves similarly to the model-based controller but also is learnable.

Finetuning of the pre-trained policy. After pre-training the policy, its performance can be further improved through RL methods, such as Proximal Policy Optimization [46] for the given task reward. The implementation also includes a velocity curriculum in the forward direction similar to [47] and a terrain curriculum over five different terrains with increasing difficulties, including uphill, downhill, flat terrain, and uneven terrains [9], [15]. Lastly, the framework randomizes the dynamic parameters and adds noises to observations for sim-to-real transfer.

V. PPF: Pre-training and Preservative Fine-tuning

In this section, we introduce Pre-training and Preservative Fine-tuning (PPF), a novel framework designed to enhance the performance of a given model-based controller by first distilling its knowledge into a learnable neural network, and then fine-tuning it using RL with model-assumption-based regularization. While our work is based on Imitating and Finetuning Model Predictive Control (IFM) [23], IFM often suffers from catastrophic forgetting, which results in poor performance in the real-world environment (Section V-A). To overcome this issue, we design a regularized loss that further distills knowledge from the expert controller during fine-tuning (Section V-B). In addition, we introduce a model-assumption-based regularization (MAR) to determine the reliability of the expert controller (Section V-C). This allows us to reject unreliable data from MBC and thus improves the performance of the fine-tuned policy. Finally, we will describe a few implementation details, such as reward functions or Lipschitz Continuity Penalty (Section V-D).

A. Motivation: Catastrophic Forgetting of Motion Style

IFM demonstrated robust performance improvements in quadrupedal locomotion. However, it often suffers from catastrophic forgetting during the fine-tuning phase due to unconstrained policy optimization. This forgetting of motion style is even more critical for bipedal locomotion tasks because of their unstable dynamics.

During RL-based fine-tuning, IFM often stabilizes the pelvis lateral movement by initially swinging the leg inward and stepping with narrower foot placement as shown in Fig. 2. However, this overfitted adaptation in a training environment can lead to potential instability, and this problem will be further exacerbated when the policy is deployed on hardware with a significant sim-to-real gap. For instance, frequent foot collisions were observed when deploying the IFM in both the AR-sim and on hardware. While domain randomization may help mitigate this issue, its performance improvement is marginal and requires extensive manual tuning through trial and error.

B. Improved Fine-tuning with Regularization

To address the motion style forgetting, we introduce a regularization term that preserves the expert controller's motion style, which is essential for obtaining high-performance bipedal locomotion policies in the real-world.

In the continual learning community [25], various approaches have been developed to address catastrophic forgetting, where a pre-trained network loses performance on

(a) MBC motion example (b) IFM motion example

Fig. 2. Motion forgetting example. Unlike MBC, IFM is trained to swing its foot inwards initially and step with narrower foot placement, optimizing for lateral tracking accuracy and reduced energy consumption.

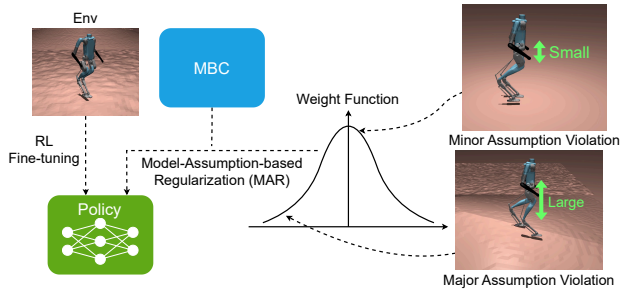

Fig. 3. Overview of Model-Assumption-based Regularization (MAR). Our framework automatically adjusts the supervised loss regularization based on the assumption violation of MBC.

previous tasks during fine-tuning. These approaches include regularizing either the network weights or the function outputs. In our case, we choose to regularize the function outputs using MBC similar to [9]:

$$L_{FullReg}(\theta,\sigma) = L_{PPO}(\theta,\sigma) + w\mathbb{E}_{(\mathbf{s},\mathbf{a}^{\mathbf{E}})\sim\mathscr{D}}\left[\|\mathbf{a}^{E} - \mu_\theta(\mathbf{s})\|_2^2\right], \tag{5}$$

where $\mathbf{a}^E$ is the expert action of the given state $\mathbf{s}$, L_{PPO} is the PPO loss, and w is a weight for the regularization loss. The policy is modeled as Gaussian distribution with the mean $\mu_\theta(\mathbf{s})$ and learnable standard deviation σ.

However, this straightforward regularization can result in a suboptimal policy because the regularization term limits the policy improvement that could be achieved by RL. In the extreme case of $w \to \infty$, the learning framework simply replicates the expert's behaviors without any improvements. Ideally, the weights should be dynamically adjusted, increasing the importance of the regularization term when the expert shows stable performance and vice versa.

C. Model-Assumption-based Regularization

To this end, we introduce model-assumption-based regularization (MAR). This approach adjusts the weight of the regularization term by measuring how much the current state violates the assumption of the model-based controller as shown in Fig. 3. Because our model follows the ALIP model assumption [26] (Eq. (18)) and incorporates the base height as feedback to the model-based controller, we use the vertical velocity of the base as a criterion to determine whether the current state violates the model assumption. Then the loss function given by Eq. (5) is modified as follows:

$$L_{PPF}(\theta,\sigma) = L_{PPO}(\theta,\sigma) + \mathbb{E}_{(\mathbf{s},\mathbf{a}^{\mathbf{E}})\sim\mathscr{D}}\left[w(\mathbf{s})\|\mathbf{a}^{E} - \mu_\theta(\mathbf{s})\|_2^2\right], \tag{6}$$

where $w : \mathbf{s} \to [0,\infty)$ is the weighting function given as follows with the smoothing parameter δ and coefficient w_0:

$$w(\mathbf{s}) = w_0 e^{\frac{-\dot{z}(\mathbf{s})^2}{\delta}}, \tag{7}$$

TABLE I

REWARD

Term	Equation	Weight
Lin Vel Track	$\exp(-\|\mathbf{v}_{xy} - \mathbf{v}_{xy}^{cmd}\|_2^2/\delta_{xy})$	1.2
Ang Vel Track	$\exp(-(\omega_z - \omega_z^{cmd})^2/\delta_\omega)$	1.1
Torque Penalty	$\|\tau\|_2^2$	-4×10^{-6}
Base Motion	$\|\omega_{xy}\|_2^2$	-0.6

In our case, $w_0 = 5$ and $\delta = 0.0159$ to preserve the motion in reliable states. This particular choice of the weighting function is based on the observation that the model assumption can be violated even when the controller performs successfully. By smoothly transitioning the weight to zero as violations increase, it dynamically adjusts the importance of expert knowledge for each sample by evaluating it against the given assumption. As a result, when a state deviates from the desirable state due to various factors, such as terrain changes or high command velocities, our framework automatically adjusts the weight of the regularization term for the corresponding states and achieves better performance. Moreover, our approach provides a more direct and intuitive adjustment of the loss function compared to the indirect tuning of reward functions [9], [22].

D. Implementation Details

Observation and action space. The observation space includes root linear and angular velocities, projected gravity, user command, lower-body joint positions and velocities, gait phase and domain, previous action, and the history of lower-body joint positions and velocities. The action space consists of target joint positions and velocities for the lower body.

Reward functions. We design a simple reward function that aims to track the target linear and angular velocities. It also regularizes excessive movements by penalizing torques and base motion. The details are listed in Table I.

Training Time and Data Size The model is trained using a GeForce RTX 4090 GPU and an Intel i9-14900K CPU. We use MuJoCo [48] for the simulator. The DAgger stage runs for 380 iterations, corresponding to 9.12 million samples, and takes approximately 5 hours and 26 minutes. The fine-tuning stage completes 10,000 iterations, generating approximately 400 million samples over 31 hours and 33 minutes.

Lipschitz Continuity Penalty. To suppress the vibration of motors, we introduce a Lipschitz continuity penalty as introduced in Chen et al. [49]. Unlike [49], we directly penalize the norm of the change in network μ_θ, which is deployed in the testing time:

$$\begin{aligned} \min_{\theta,\sigma} \quad & L_{PPF}(\theta,\sigma) \\ \text{s.t.} \quad & \max_{\mathbf{s}} \|\nabla_{\mathbf{s}}\mu_\theta(\mathbf{s})\|^2 \le K^2, \end{aligned} \tag{8}$$

where K is the Lipschitz constant. Following the simplification process in [49], it is formulated as follows:

$$\min_{\theta,\sigma} \quad L_{PPF}(\theta,\sigma) + \alpha\mathbb{E}\left[\|\nabla_{\mathbf{s}}\mu_\theta(\mathbf{s})\|^2\right] \tag{9}$$

where α is a tunable variable set to $\alpha = 0.0001$ in our case.

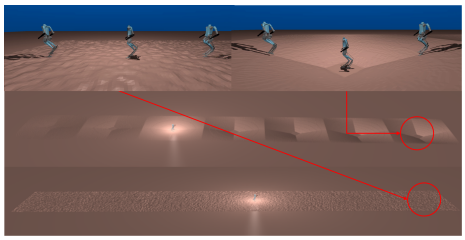

Fig. 4. Testing terrains for the robustness tests in MuJoCo.

VI. Experimental Results

We design simulation tests and hardware experiments to investigate the following questions: (1) Can PPF learn an effective policy in the training environments? (2) Can PPF show robust performance in sim-to-sim and sim-to-real transfer scenarios compared to the baseline methods? (3) Can MAR dynamically adjust the sample weights based on the model-based assumption violation?

A. Experimental Details

1) Baselines: We consider the following baselines for our simulation tests and hardware experiments. All neural network-based controllers use a multi-layer perceptron with hidden layers of sizes 512, 256, and 64.

- **MBC:** As a model-based control (MBC) baseline, we adopt the passivity-based whole-body controller proposed in [43]. In MuJoCo [48], this controller is adjusted to run at 200 Hz with action space conversion applied, as described in [23].
- **IFM:** IFM [23] learns a policy by imitating the given MBC and fine-tuning it. We employ the same reward configuration and Lipschitz penalty.
- **FullReg:** FullReg fully regularizes the learning with the MBC-labeled actions (Eq. 5) without MAR.
- **PureRL:** PureRL is trained with reinforcement learning without any pre-training or regularization. Its reward configuration is similar to [9]. However, we leverage a swing foot trajectory from an ALIP foot placement controller, add rewards for foot air time and second-order action smoothing, and exclude the selected joint position penalty and target joint position smoothing.

2) Humanoid: As a robotic platform, we employ the Digit provided by Agility Robotics. This has the same hardware configuration as described in [9].

3) Simulation: We used MuJoCo [48] for training and testing (Sec. VI-B), and the company-provided simulator (AR-Sim) for testing (Sec. VI-C). MuJoCo is a high-performance physics engine designed to simulate complex dynamic systems. AR-Sim offers dynamics that closely mirror the dynamics of the physical robot. Once the training is done in MuJoCo, we deploy policies to either AR-Sim or hardware without additional fine-tuning.

B. Training Environment Experiments

We evaluate the robustness of the controllers in MuJoCo on sloped and uneven terrains (Fig. 4). The robot is commanded to move forward at 0.6 m/s while adjusting its orientation to maintain a forward direction. As the robot moves forward, it traverses a sequence of terrains with increasing difficulty. The test is conducted with five different random seeds, and each controller runs until the robot either falls, reaches the time limit of 130 seconds, or reaches the end of the last level of the terrain (goal location).

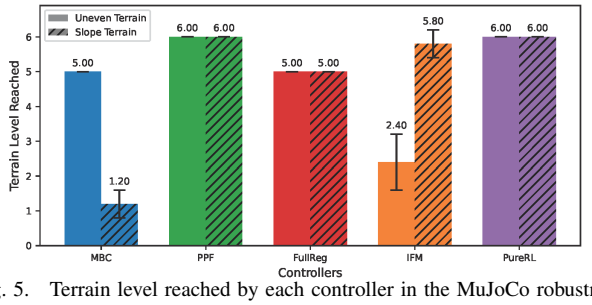

Fig. 5. Terrain level reached by each controller in the MuJoCo robustness test. PPF successfully traverses all terrains within the time limit. IFM stumbles over its own feet on uneven terrain. FullReg fails to complete the final terrain due to poor tracking performance.

TABLE II
AR-Sim Robustness Performance

Method	10° Uphill		12° Uphill		14° Uphill	
	Succ. (%)	Track Err. (↓) (%)	Succ. (%)	Track Err. (↓) (%)	Succ. (%)	Track Err. (↓) (%)
MBC	0	34.5	0	33.2	0	34.4
PPF	**100**	**8.4**	**100**	**9.1**	**100**	**18.9**
FullReg	**100**	12.1	**100**	13.1	**100**	28.7
IFM	**100**	15.5	**100**	24.9	**100**	30.7
PureRL	85	10.9	60	15.4	0	54.1

We measure the terrain level reached by each controller shown in Fig. 5. Overall, PPF (ours) and PureRL show the best performance among all baselines. While IFM reaches near the final level on sloped terrain, it often fails on uneven terrain due to unexpected foot contacts. These contacts cause severe lateral and angular disturbances, eventually leading the robot to trip over its own foot and fall. In contrast, PPF robustly handles such disturbances as a result of its pre-trained motion and consistently reaches the final terrain level in both terrains. FullReg also demonstrates this level of robustness compared to IFM, but its tracking performance degrades under model-assumption violations, causing it to time out before reaching the goal. A more detailed discussion of this is provided in Section VI-E. PureRL also reaches the goal within the time limit on both terrains. However, the policy is overfitted to the simulation environment, resulting in degraded motion quality, which may lead to poor performance in both sim-to-sim and sim-to-real transfer. We discuss this further in Sections VI-C and VI-D.

C. Sim-to-sim Transfer Experiments

In this subsection, we evaluate sim-to-sim transfer performance through experiments in AR-Sim.

1) Robustness Test: We first evaluate robustness by commanding the robot to traverse uphill platforms with three different slopes and measure the success rate and linear velocity tracking error with respect to the commanded velocity, as shown in Table II. A trajectory is considered successful if the robot traverses the 5-meter uphill section without falling.

Overall, PPF, IFM, and FullReg demonstrate strong robustness, while MBC and PureRL exhibit significantly degraded

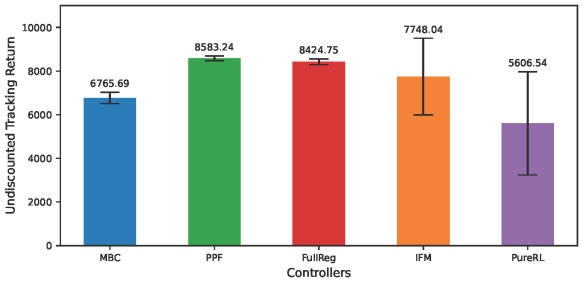

Fig. 6. Mean undiscounted return of tracking rewards in the AR-Sim random velocity tracking test. The command range is wider than that used during training. Policies learned with regularization, PPF and FullReg, show superior performance compared to those without MBC regularization.

performance. MBC consistently performs the worst, whereas controllers that imitate it (PPF, FullReg, and IFM) show substantially improved robustness. This improvement can be attributed to training the policies on diverse terrains with reinforcement learning objectives, which improves their ability to handle challenging environments. Although PureRL successfully navigates the final slope terrain in the MuJoCo test, it achieves much lower success rates in AR-Sim. This highlights the importance of incorporating the periodic gait style of MBC, which facilitates transfer to different simulators and potentially to real-world environments.

Among the methods achieving a 100% success rate, PPF outperforms the others in tracking accuracy, exhibiting the smallest increase in tracking error as the slope angle increases. In contrast, FullReg shows worse tracking performance, suggesting that full regularization with MBC degrades tracking accuracy in the presence of model assumption errors. We further analyze the effect of MAR in Section VI-E in the presence of model-assumption violation.

2) Random Velocity Tracking Test: We further evaluate the tracking performance of each controller by providing four random velocity commands on flat terrain. To assess generalization in the command space, the commands are sampled from a range that includes previously unseen lateral and angular velocities. We measure the mean undiscounted return of linear and angular velocity tracking rewards over 100 trajectories, as shown in Fig. 6.

PPF outperforms all baselines, achieving a slightly higher return than FullReg. In contrast, IFM shows worse tracking performance than the other regularization-based methods. This is mainly due to the robot frequently tripping over its own foot and falling when given sudden changes in lateral or angular commands.

Despite its robust performance in the training environment, PureRL demonstrates the worst performance among the baselines, performing even worse than MBC. This result aligns with the observations from the previous experiments, supporting the claim that motion distillation from MBC, whether through pre-training alone (IFM) or in combination with regularization (PPF and FullReg), enhances the controller's robustness to domain shifts.

D. Hardware Experiments

We deploy the controllers on hardware across six different scenarios, both indoor and outdoor, as illustrated in Fig. I.

TABLE III
HARDWARE EXPERIMENTS

Method	Indoor Experiment		Outdoor Experiment		
	Max. Forward Vel. (m/s)	Step & Slip Succ. (%)	Slope Succ.	Uneven Succ.	Sand Succ.
MBC	0.5 m/s	0%	✗	✗	N/A
PPF	**1.5 m/s**	**100%**	✓	✓	✓
FullReg	1.1 m/s	0%	✓	✓	✓
IFM	>1.0 m/s (unsafe)	40%	N/A	N/A	N/A
PureRL	N/A	0%	N/A	N/A	N/A

The results are summarized in Table III.

1) Indoor Experiments: In indoor experiments, we measure the maximum forward velocity on flat terrain and the success rate of traversing the step-and-slip platform. This platform consists of a whiteboard covered with poppy seeds or olive oil to induce slipping, and the robot is commanded to traverse it at 0.3 m/s. The success rate is measured over five trials.

PPF achieves the best performance among all baselines, reaching a maximum velocity of 1.5 m/s while maintaining a 100% success rate in the step-and-slip test. In contrast, MBC and FullReg reach maximum velocities of only 0.5 m/s and 1.1 m/s, respectively. Both controllers fail the step-and-slip test in all trials, slipping in the middle of the whiteboard. Despite being trained with the same RL objective as PPF, FullReg shows limited performance and exhibits a failure pattern similar to MBC.

One possible reason for PPF's performance gain is its ability to handle model assumption violations at higher velocities, whereas MBC requires further tuning of its parameters [26]. Additionally, by adjusting reliance on MBC action regularization in certain states and placing greater emphasis on the RL objective, PPF better adapts to prevent slips.

IFM struggles with stability during the maximum forward velocity test, frequently tripping over its own foot. For safety, we halted the experiment at 1.0 m/s. PureRL performs the worst in hardware tests. Its maximum velocity could not be measured due to poor tracking performance, and it fails the step-and-slip test as its foot becomes stuck on the whiteboard surface. PureRL struggles to transfer to new domains beyond its training domain unless the motion is carefully designed through extensive reward engineering or it is trained with additional components for sim-to-real, as demonstrated in [9].

2) Outdoor Experiments: We deploy the robot in real-world outdoor scenarios, including sloped, uneven, and sandy terrains. IFM and PureRL are excluded due to unsafe behavior observed even in the controlled indoor testbed.

Both PPF and FullReg successfully traverse all terrain types, including deformable surfaces such as sand, which were not included during training. In contrast, MBC shows limited robustness, struggling even on mildly challenging terrain. Please refer to the supplemental video for details.

E. Effect of MAR in Training and Testing

Although FullReg demonstrates robust performance in both simulation and hardware, its capabilities are often

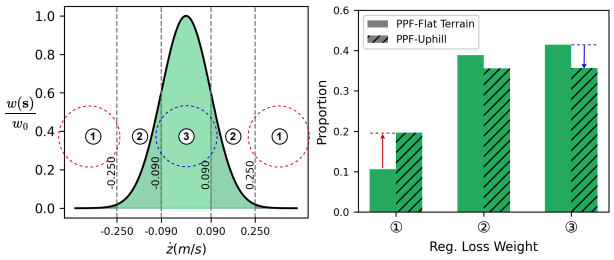

Fig. 7. **Left:** an adaptive weight distribution against model assumption violation. **Right:** a histogram of regularization loss weight during PPF training on both uphill and flat terrain. We observe more unreliable samples with low regularization weight (region 1) and fewer reliable samples with high weight (region 3) in the uphill case, reflecting how PPF filters out unreliable regularization in violation-prone regions.

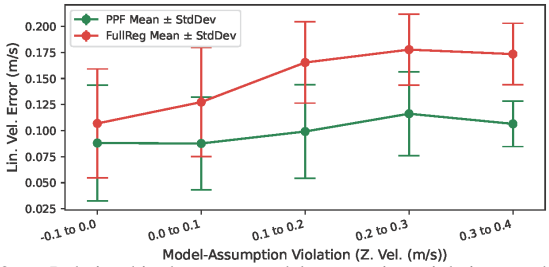

Fig. 8. Relationship between model-assumption violations and linear velocity tracking errors during a 14-degree uphill test. FullReg exhibits higher tracking errors than PPF when model-assumption violations are more pronounced.

limited compared to PPF when model-assumption violations occur, including the maximum velocity it can reach and tracking ability. We analyze how MAR takes effect when there are model-assumption violations in training and testing.

1) Analysis of MAR at Training: During training, MAR dynamically reduces the weight of the regularization loss as the model-assumption violation increases, enabling higher performance while preserving the pre-trained motion. In Fig. 7, we plot the normalized histogram of regularization loss weights across three levels of model-assumption violation, evaluated on both flat and uphill terrains. This illustrates how the weighting adapts dynamically to different scenarios.

Overall, MAR successfully detects model-assumption violations and leverages them to adjust the regularization weights dynamically. On flat terrain, violations are infrequent except under high-velocity commands. In this case, a large proportion of MBC actions remain reliable (region 3 in Fig. 7), resulting in a higher regularization weight. In contrast, more violations are observed on sloped terrain (region 1 in Fig. 7), leading to lower weights. This enables PPF to downweight suboptimal regularization, thereby improving performance in both simulated and real-world environments.

2) Analysis of MAR at Testing: We examine the relationship between model-assumption violations and linear velocity errors for PPF and FullReg to evaluate the effectiveness of the MAR during testing. Fig. 8 shows the relationship between model-assumption violation (Z-velocity) and linear velocity error in the 14-degree uphill test.

During testing, the model trained with MAR demonstrates consistent performance even in the presence of model-assumption violations. For FullReg, linear velocity tracking error increases as violations become more frequent.

Interestingly, MAR improves not only performance on challenging terrains but also on simpler ones. While PPF performs better on uneven terrain and slopes in MuJoCo simulations, it also outperforms other methods in the Maximum Forward Velocity test on flat terrain and the Step-and-Slip test on hardware, as shown in Table III. This improvement may result from increased model-assumption violations caused by high-speed or unstable locomotion even in simple scenarios, or it may reflect PPF's ability to achieve more general performance by ignoring irrelevant samples during training.

VII. CONCLUSIONS

We propose a novel learning framework, PPF, that combines model-based control and learning-based approaches to effectively train a robust humanoid locomotion control policy. PPF integrates several components, such as pre-training through imitation of a model-based controller, fine-tuning via reinforcement learning, and model assumption-based regularization (MAR), to enhance the robustness and adaptability of humanoid robots across diverse and challenging tasks. Our approach addresses key limitations of prior methods, such as the limited performance of model-based controllers and the catastrophic forgetting in IFM [23].

Through extensive simulation tests and hardware experiments on the Digit humanoid robot, we demonstrate that PPF outperforms baseline methods in terrain robustness, velocity tracking, and sim-to-real transfer. PPF achieves a maximum forward velocity of 1.5 m/s and can navigate complex terrains, including slippery surfaces, slopes, uneven ground, and deformable sand, with zero-shot deployment. The incorporation of MAR enables the policy to adapt to scenarios where model assumptions are violated, resulting in superior performance compared to fully regularized methods.

One future extension of this work is to apply MAR to a broader range of model-based assumptions. For example, MAR can utilize estimated foot slipperiness through stance foot velocity, as most high fidelity models assume zero contact velocity. Furthermore, applying MAR to more sophisticated model-based controllers could provide insights into whether incorporating stronger prior knowledge during pre-training enhances training efficiency and final performance. Another possible extension is to adapt the framework for more dynamic and high-impact motions, such as running or jumping. This could involve refining the model assumptions or regularization techniques to better handle faster dynamics and increased instability, broadening the framework's applicability to agile robotic behaviors.

REFERENCES

[1] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2001, pp. 239–246.

[2] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2006, pp. 200–207.

[3] J. Li, J. Ma, O. Kolt, M. Shah, and Q. Nguyen, "Dynamic loco-manipulation on hector: Humanoid for enhanced control and open-source research," arXiv preprint arXiv:2312.11868, 2023.

[4] G. Gibson, O. Dosunmu-Ogunbi, Y. Gong, and J. Grizzle, "Terrain-adaptive, alip-based bipedal locomotion controller via model predictive control and virtual constraints," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 6724–6731.

[5] N. Scianca, D. De Simone, L. Lanari, and G. Oriolo, "Mpc for humanoid gait generation: Stability and feasibility," *IEEE Trans. Robot*, vol. 36, no. 4, pp. 1171–1188, 2020.

[6] Z. Gu, R. Guo, W. Yates, Y. Chen, Y. Zhao, and Y. Zhao, "Walking-by-logic: Signal temporal logic-guided model predictive control for bipedal locomotion resilient to external perturbations," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024, pp. 1121–1127.

[7] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 295–302.

[8] C. Khazoom, S. Hong, M. Chignoli, E. Stanger-Jones, and S. Kim, "Tailoring solution accuracy for fast whole-body model predictive control of legged robots," *arXiv preprint arXiv:2407.10789*, 2024.

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

[10] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Robust and versatile bipedal jumping control through reinforcement learning," in *Proc. Robot.: Sci. Syst.*, 2023.

[11] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 7309–7315.

[12] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning," in *Proc. Robot.: Sci. Syst.*, 2021.

[13] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Sci. Robot.*, vol. 7, no. 62, p. eabk2822, 2022.

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

[15] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. Conf. Robot Learn. (CoRL)*, 2022, pp. 91–100.

[16] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid Motor Adaptation for Legged Robots," in *Proc. Robot.: Sci. Syst.*, 2021.

[17] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning Agile Robotic Locomotion Skills by Imitating Animals," in *Proc. Robot.: Sci. Syst.*, 2020.

[18] Z. Xie, X. Da, B. Babich, A. Garg, and M. v. de Panne, "Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model," 2021, arXiv:2104.09771.

[19] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Proc. Conf. Robot Learn. (CoRL)*, 2021, pp. 773–783.

[20] X. Da, Z. Xie, D. Hoeller, B. Boots, A. Anandkumar, Y. Zhu, B. Babich, and A. Garg, "Learning a contact-adaptive controller for robust, efficient legged locomotion," 2020, arXiv:2009.10019.

[21] S. Yu, N. Perera, D. Marew, and D. Kim, "Learning generic and dynamic locomotion of humanoids across discrete terrains," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2024, pp. 1048–1055.

[22] H. J. Lee, S. Hong, and S. Kim, "Integrating model-based footstep planning with model-free reinforcement learning for dynamic legged locomotion," *arXiv preprint arXiv:2408.02662*, 2024.

[23] 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 Robot. Autom. Lett.*, vol. 8, no. 11, pp. 7799–7806, 2023.

[24] A. Miller, S. Fahmi, M. Chignoli, and S. Kim, "Reinforcement learning for legged robots: Motion imitation from model-based optimal control," 2023. [Online]. Available: https://arxiv.org/abs/2305.10989

[25] L. Wang, X. Zhang, H. Su, and J. Zhu, "A comprehensive survey of continual learning: Theory, method and application," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 46, no. 8, pp. 5362–5383, 2024.

[26] Y. Gong and J. Grizzle, "Zero dynamics, pendulum models, and angular momentum in feedback control of bipedal locomotion," arXiv preprint arXiv:2105.08170, 2022.

[27] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous robots*, vol. 35, pp. 161–176, 2013.

[28] G. Romualdi, S. Dafarra, G. L'Erario, I. Sorrentino, S. Traversaro, and D. Pucci, "Online non-linear centroidal mpc for humanoid robot locomotion with step adjustment," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 10 412–10 419.

[29] P. Wensing, "Optimization and control of dynamic humanoid running and jumping," Ph.D. dissertation, The Ohio State University, 2014.

[30] J. Li, O. Kolt, and Q. Nguyen, "Continuous dynamic bipedal jumping via real-time variable-model optimization," arXiv preprint arXiv:2404.11807, 2024.

[31] E. Dantec, M. Naveau, P. Fernbach, N. Villa, G. Saurel, O. Stasse, M. Taix, and N. Mansard, "Whole-body model predictive control for biped locomotion on a torque-controlled humanoid robot," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2022, pp. 638–644.

[32] 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.

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

[34] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi, "Learning human-to-humanoid real-time whole-body teleoperation," *arXiv preprint arXiv:2403.04436*, 2024.

[35] Y. Gong and J. Grizzle, "One-step ahead prediction of angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-inspired controller," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 2832–2838.

[36] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *Proc. Amer. Control Conf. (ACC)*, 2019, pp. 4559–4566.

[37] I. Kostrikov, L. M. Smith, and S. Levine, "Demonstrating A Walk in the Park: Learning to Walk in 20 Minutes With Model-Free Reinforcement Learning," in *Proc. Robot.: Sci. Syst.*, 2023.

[38] L. M. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and Adapting Agile Locomotion Skills by Transferring Experience," in *Proc. Robot.: Sci. Syst.*, 2023.

[39] L. Smith, J. C. Kew, X. Bin Peng, S. Ha, J. Tan, and S. Levine, "Legged robots that keep on learning: Fine-tuning locomotion policies in the real world," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 1593–1599.

[40] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," in *Proc. Conf. Robot Learn. (CoRL)*, 2020.

[41] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Proc. Robot.: Sci. Syst.*, 2019.

[42] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," in *Proc. Robot.: Sci. Syst.* Virtual Conference: Robotics: Science and Systems Foundation, July 2021. [Online]. Available: https://roboticsconference.org/program/papers/220/

[43] A. Shamsah, Z. Gu, J. Warnke, S. Hutchinson, and Y. Zhao, "Integrated task and motion planning for safe legged navigation in partially observable environments," *IEEE Trans. Robot*, vol. 39, no. 6, pp. 4913–4934, 2023.

[44] H. Sadeghian, C. Ott, G. Garofalo, and G. Cheng, "Passivity-based control of underactuated biped robots within hybrid zero dynamics approach," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 4096–4101.

[45] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. Int. Conf. Artif. Intell. Stat. (AISTATS)*, 2011, pp. 627–635.

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

[47] G. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," in *Robotics: Science and Systems*, 2022.

[48] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.

[49] Z. Chen, X. He, Y.-J. Wang, Q. Liao, Y. Ze, Z. Li, S. S. Sastry, J. Wu, K. Sreenath, S. Gupta, and X. B. Peng, "Learning smooth humanoid locomotion through lipschitz-constrained policies," *arxiv preprint arXiv:2410.11825*, 2024.