Stylized Locomotion Control for Bipedal Humanoid Robots with Adversarial Motion Priors

Michael Piseno[1], Zhongyu Li[2], Alejandro Escontrela[2],
Pieter Abbeel[2], Koushil Sreenath[2], Xue Bin Peng[3], Atil Iscen[4]

***Abstract*— Obtaining whole-body bipedal locomotion controllers that produce motions in the style of a human remains a challenging problem. To produce human-like locomotion, the robot must not only stabilize itself during movement but also execute complex whole-body motions that, while not necessarily aiding stability, enhance the expressiveness and energy efficiency of motion. Reinforcement learning has shown promise in learning locomotion controllers for dynamic motions, however, developing locomotion controllers for these stylized motions and transferring them to real hardware remains an open challenge. In this work, we introduce a framework for learning such a controller by leveraging Adversarial Motion Priors (AMP). Unlike other methods for stylized locomotion, AMP has the benefit of easily allowing for multiple motions, but often suffers from mode collapse to a single motion To address this challenge, we adopt a Multi-Task RL setup, and demonstrate the added benefit of learning multiple skills for policy robustness. We validate the effectiveness of our method through extensive benchmarking on multiple bipedal humanoid models. Moreover, our controller generates energy-efficient motions without explicit optimization for energy consumption, transitions smoothly between different motions, and is deployable on real humanoid robots, demonstrating the first multi-skill locomotion control policy capable of handling diverse stylized humanoid motions.**

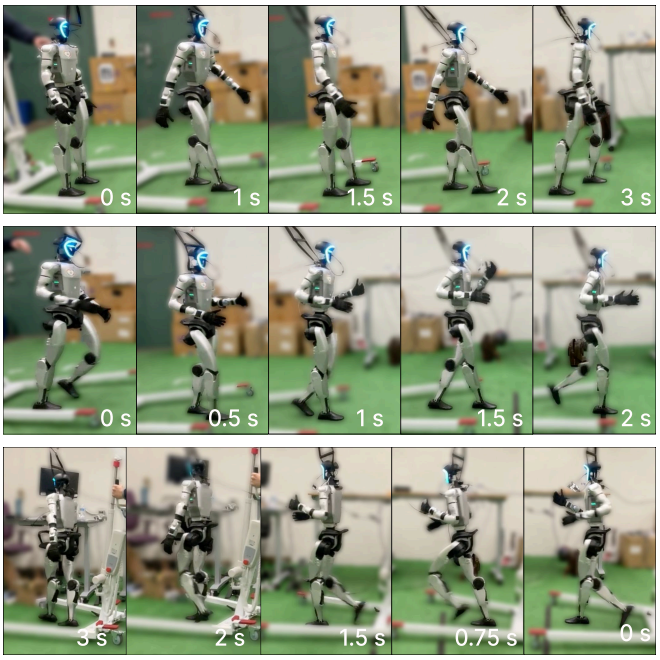


Fig. 1: Real world deployment of the multi-skill policy performing the walking skill (top), jogging skill (middle), and transition from jogging to walking (bottom) on the G1 robot.

I. Introduction

Humanoid robots have garnered increasing interest in recent years, with significant advancements in reinforcement learning (RL) for achieving robust locomotion control [1]–[3]. However, despite the rapid development of humanoid platforms, their walking gaits often appear unnatural. For instance, many robots adopt excessive knee bending to enhance stability [4]–[6], a strategy not commonly observed in human locomotion. Humans, by contrast, move gracefully moving while maintaining energy efficiency by minimizing unnecessary knee flexion, exhibiting diverse stylized gaits such as cat-walking or walking with emotions, and coordinating whole-body movements to convey body language. Besides having known benefits for cost of transform [7], natural motions are desirable for humanoid robots if they are to operate and co-exist in human environments as relatable and interactive companions. Therefore, we aim to learn controllers from human motion data that exhibit stylized motion.

Adversarial Motion Priors (AMP) [8] offers a promising approach for learning stylized locomotion control by training a GAN-style objective to learn a reward function for guiding the policy to generate motion that matches a training dataset distribution, but its effectiveness has largely been limited to simulated characters with an access to full states of the environment. When deployed on a real robot, the resulting stylized locomotion controllers encounter a trade-off between stability and expressiveness [9]. This is because whole-body movement involves real-time control of a high-dimensional nonlinear system, where small perturbations can easily destabilize the entire system. Because of this, previous controllers tend to prioritize stability over expressiveness, often adopting conservative robot behaviors such as excessive knee bending and excluding AMP-style rewards.

To address this challenge we propose to use Multi-Task Reinforcement Learning to improve robustness while adhering to the style of multiple skills. For clarity we will say "Multi-Skill RL", meaning multiple different motions,

[1]Stanford University. mpiseno@stanford.edu
[2]University of California, Berkeley.
{zhongyu_li, escontrela, koushils}@berkeley.edu, pabbeel@cs.berkeley.edu
[3]Simon Fraser University. xbpeng@sfu.ca
[4]Google DeepMind. atil@google.com
View our project site at https://www.michaelpiseno.com/biped\protect_ctrl.github.io/

instead of "Multi-Task RL" so as to not be confused with our concept of a "task" which we define later. Our key insight is that unlike other locomotion controllers that define very specific reward functions that do not offer much flexibility, AMP prescribes softer constraints on the resulting motion, allowing the controller to potentially learn a whole distribution of motions. These additional motions, or "skills", not only benefit exploration during training (indeed they often help each other learn where a single-skill policy would not), but also serve to improve policy robustness. However, increasing motion diversity under these challenging conditions often leads to mode collapse in adversarial learning frameworks (e.g., resulting in deficiencies when using AMP in the benchmark presented in [10]), where the learned policy fails to capture the full spectrum of natural movements and instead converges to an easier-to-learn maneuver. To address this, we also condition the discriminator on a representation of the skill in addition to the policy.

A. Contributions

In this work, we aim to ascertain the feasibility of developing a unified locomotion controller that enables humanoid robots to perform diverse stylized locomotion using AMP and Multi-Skill RL. Our key contributions are as follows:

- We develop a learning framework that enables various stylized whole-body locomotion skills on humanoid robots using a single multi-skill policy, deployable in the real world.
- We motivate the use of stylized locomotion via AMP over hand-crafted rewards through an in-depth quantitative analysis of the resulting motions compared to bipedal robots trained with traditionally used hand-designed reward formulations.
- We propose Multi-Skill RL as robustness technique for stylized locomotion controllers.

We validate our approach on four humanoid robot models with diverse bipedal morphologies: Digit and Cassie from Agility Robotics, G1 from Unitree Robotics, and a variant of the Berkeley Humanoid Robot (BHL) [11]. These robots represent a range of digitigrade and plantigrade designs, as well as bipeds with and without an upper body. We use G1 as the testbed for hardware experiments, while all four robots are employed for benchmarking and ablation studies of the proposed algorithm.

B. Related Work

Reward Specification for Locomotion: In locomotion, reward specification is difficult and often involves optimizing many different potentially conflict quantities. To address these issues, researchers have investigated task-specific action spaces [12], [13], complex style reward formulations [1], [14]–[16], and curriculum learning [17], [18]. These approaches achieve state-of-the-art results in locomotion, but defining custom action spaces and hand-designing reward functions requires substantial domain knowledge and a delicate tuning process. Additionally, these approaches are often platform-specific and do not generalize easily across tasks.

Learning Stylized Motion for Humanoids: In the realm of computer graphics, Adversarial Motion Priors (AMP) [8] leverage GAN-style training to learn a "style" reward from a reference motion dataset. The style reward encourages the agent to produce a trajectory distribution that minimizes the Pearson divergence between the reference trajectories and the policy trajectories [19]. However, AMP is typically used in character animation domains where the simulated characters have access to a fully observable state [20]–[22]. Achieving expressive motions on a real robot typically requires very complex reward functions or limits the human-like motion to the upper body only [23], [24]. Recent work uses a discriminator reward for whole body locomotion on a real humanoid robot [9], but is limited in expressiveness and task diversity due to its phase condition in the policy. Concurrent work has succeeded in achieving human-like locomotion on a real robot, but uses reference motion in the policy observation and is therefore also limited in task diversity since the reference motion essentially acts as a phase condition [25]. By contrast, our controller achieves human-like locomotion on a real robot across a diverse range of skills and tasks using a simple reward made possible by motion priors.

II. Preliminaries

A. Terminology

We briefly introduce the terminology of "skill" and "task" in the context of legged locomotion control to align readers with our writing. In this domain, a **skill** refers to different gaits, primarily distinguished by their contact plans—i.e., variations in contact timing and sequence define different skills. For example, walking, jogging, running, and walking with distinct styles, such as catwalking and nominal walking, each have different contact timings, making them distinct skills. A **task**, on the other hand, involves a specific objective to achieve, not the manner in which to achieve it. For instance, a nominal walking skill can be applied to perform different tasks, such as tracking various target velocities. Importantly, a single skill can be used for multiple tasks, and conversely, a given task (e.g., moving at 0.5 m/s) can be realized using different skills. This distinction is exemplified in prior work, such as the **single-skill multi-task** bipedal locomotion policy developed in [3] and the **non-conditioned multi-skill multi-task** quadrupedal policy from [10], where the policy autonomously selects the appropriate skill based on the given task. However, in these approaches, the human operator has no control over which skill the robot performs.

B. Problem Definition

We formulate the problem of control for bipedal locomotion as a Markov Decision Process (MDP). An MDP is a 5-tuple $(\mathcal{S}, \mathcal{A}, R, P, \gamma)$ defined by a state space $\mathcal{S}$ which is the set of possible environment states, action space $\mathcal{A}$ which is the set of possible actions, reward function $R(s_t, a_t, s_{t+1})$ which gives the reward of taking action a_t in state s_t at timestep t and ending up in state s_{t+1}, transition function $P(s_{t+1}|s_t, a_t)$ describing the probability of ending up in

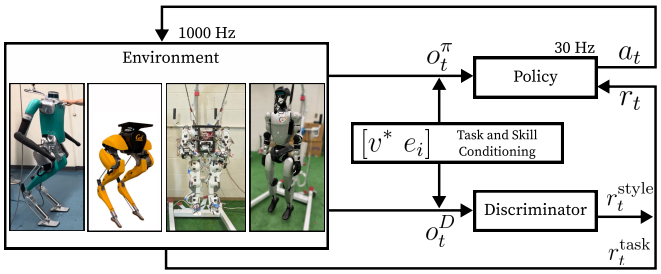


Fig. 2: Overview of the framework. A discriminator takes as input an observation o_t^D and produces the learned style reward r_t^{style}. Additional reward r_t^{task} based on adherence to the task is given by the environment, yielding the total reward r_t used to update the policy. Our robot platforms are also shown in the environment block, from left to right: Digit, Cassie, BHL, G1.

state s_{t+1} when taking action a_t in state s_t, and discount factor γ which serves to discount future rewards.

In the real world, the exact state of the environment is not accessible. Instead, we make observations of the environment to gain limited information about the true state. We introduce the notion of an observation space $\mathcal{O} \subseteq \mathcal{S}$ and an observation o_t comprised of measurable quantities of the environment. The problem formulation now takes the form of a Partially Observable Markov Decision Process (POMDP) to highlight the fact that we do not have full observability of the state. Details of our observation and action spaces are given in section III-A.

To simulate an agent that learns to act optimally within the POMDP, we define a policy $\pi : \mathcal{O} \to \mathcal{A}$ which takes observation $o_t \in \mathcal{O}$ as input and outputs actions $a_t \in \mathcal{A}$ at each timestep t. The policy is implemented using a neural network. We learn the parameters of the policy network using model-free reinforcement learning (RL). Model-free RL is a method of solving MDPs that maximizes the expected sum of future discounted rewards. Concretely, for a policy π, the objective is to maximize:

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T-1}\gamma^t r_t\right], \tag{1}$$

where $\tau = \{(o_t, a_t, r_t)\}_{t=0}^{T-1}$ is a trajectory produced by the policy when acting in the environment. We use Proximal Policy Optimization (PPO) [26] as our RL algorithm. To handle Multi-Skill RL, we consider a set of skills Ω and tasks which we append to the observation at each timestep. Concretely, $e_i \sim \Omega$ and $v^* \sim$ will be our skill one-hot encoder and task specification (in this case target velocity), but these could be any skill and task formulation in general. We model tasks as "what" to do and skills as "how" to do it.

C. Adversarial Motion Priors

Style Reward: In Adversarial Motion Priors (AMP) [8], a style reward function is learned that encourages the distribution of policy-generated trajectories to closely match the distribution of reference trajectories. AMP assumes access to a dataset of reference motions $d^{\mathcal{M}}$ and learns a style reward function concurrently with the policy using a GAN-based approach. More specifically, the policy acts as a generator while a discriminator learns to distinguish between the reference data $d^{\mathcal{M}}$ and the policy-generated data d^{π}. In doing so, a style reward can be calculated which is larger when d^{π} is close to $d^{\mathcal{M}}$ and lower when d^{π} is far from $d^{\mathcal{M}}$.

The discriminator is implemented as a neural network D_ψ as in [8]. D_ψ takes input o_t^D, which need not be the same as the policy input o_t^π, and outputs a scalar value $d_t \in [-1, 1]$ representing whether o_t^D came from the reference data or the policy. A value closer to -1 means the discriminator believes o_t^D came from the policy, while a value closer to +1 means the discriminator believes o_t^D came from the reference data. Using d_t, we can then compute the style reward as follows:

$$r_t^{\text{style}} = \max\{1 - 0.25 \cdot (d_t - 1)^2, 0\}. \tag{2}$$

Task Reward: The AMP framework optionally allows for additional hand-crafted reward terms, which we call task rewards. Because of the strong prior given by the style reward, these hand-crafted task rewards can be much simpler than would otherwise be required for achieving expressive locomotion. In general, this reward could encompass any quantity that should be explicitly optimized, such as minimizing energy output or tracking a heading angle. In our experiments, we use a simple task reward based on tracking a target velocity. Details of our reward functions are given in section III-B.

TABLE I: Parameters for Domain Randomization

Parameter	Distribution and Range	Unit
Modeling Error	**Uniform**	
Floor Friction Coefficient	$[0.4, 1.2]$	
Link Mass	$[0.8, 1.2] \times$ default	kg
Pelvis Mass	$[-1, 1] +$ default	kg
Friction Loss	$[0.8, 1.2] \times$ default	
Armature	$[0.8, 1.2] \times$ default	
Random Perturbations	$[-0.5, 0.5]$	$\mathrm{m\,s^{-1}}$
Action Delay	$[0, 50]$	Hz
Sensor Noise	**Uniform**	
Motor Position	$[-0.05, 0.05]$	rad
Motor Velocity	$[-0.5, 0.5]$	$\mathrm{rad\,s^{-1}}$
Projected Gravity Noise	$[-0.05, 0.05]$	m
Base Angular Velocity	$[-0.2, 0.2]$	$\mathrm{rad\,s^{-1}}$

III. Method

A. Observation Space and Action Space

Observation Spaces: The policy input o_t^π is comprised of the rotation in quaternion form θ_t, joint positions q_t, joint velocities $\dot{q}_t$, and previous action a_{t-1}. This information from the previous 10 timesteps, including the current timestep t, is concatenated along with a target xy velocity v^* and one-hot skill encoding e_i for skill i. The value function input o_t^V is constructed in a similar manner, but with additional privileged information, namely, the base linear velocity v_t and angular velocity ω_t. The discriminator input o_t^D contains

the joint positions, joint velocities, feet and hand positions in the robot's local frame p_t^{fh}, and e_i, and uses a 3-timestep history. The target velocity is always in the robot's local frame. Our control frequency is $\lambda_{\text{env}} = 30\,\text{Hz}$.

$$o_t^{\pi} = \left[\{\theta_{t'}, q_{t'}, \dot{q}_{t'}, a_{t'-1}\}_{t'=t-9}^{t}, v^*, e_i\right] \tag{3}$$

$$o_t^{V} = \left[\{v_{t'}, \omega_{t'}, \theta_{t'}, q_{t'}, \dot{q}_{t'}, a_{t'-1}\}_{t'=t-9}^{t}, v^*, e_i\right] \tag{4}$$

$$o_t^{D} = \left[\{q_{t'}, \dot{q}_{t'}, p_{t'}^{fh}\}_{t'=t-2}^{t}, e_i\right] \tag{5}$$

Action Space: The action a_t represents the desired joint position offsets from a nominal joint position. Once the desired joint positions are calculated, we apply a low-pass filter to attenuate high-frequency noise in the action. We find that the low-pass filter results in a significant learning speed-up. After the desired joint positions are filtered, the motor torques are computed using PD control.

B. Reward

Following section II-C, we have two distinct sources of reward. The first is a style reward, computed as in Eq. 2. The second is a task reward, intended to track a target velocity, and is formulated as

$$r_t^{\text{vel}} = \max\left\{1 - \|v^* - v_t\|_2, 0\right\}, \tag{6}$$

where v_t is the linear velocity of the robot's base at timestep t and v^* is the target velocity. This reward formulation encourages the policy to attempt movement where otherwise it might opt to stand still to minimize the risk of falling over.

For most motions, Eq. 6 is a sufficient task reward, however it is sometimes possible for the policy to exploit subtleties in the reference motion or robot morphology that lead to unnatural-looking motions while still achieving a reasonable style reward. For example, when learning a jogging motion for the Cassie robot, it instead learned an unnatural-looking hopping motion. Consequently, it is sometimes necessary to add additional terms to the task reward to discourage such behaviors. However this reward tuning process is generally much simpler thanks to the strong prior offered by the style reward.

For the Digit and G1 robots, we simply use the task reward $r_t^{\text{task}} = r_t^{\text{vel}}$. For the Cassie robot, we additionally include a penalty that gives zero task reward if both feet are off the ground to discourage the aforementioned hopping behavior:

$$r_t^{\text{task}} = r_t^{\text{vel}} \cdot \mathbb{1}\{\text{At least one foot on ground}\}. \tag{7}$$

For BHL, the robot sometimes stalled at the beginning of episodes. To prevent this, we include the foot velocities in the robots local frame to the discriminator observation.

With task and style reward functions defined, we can compute the total reward r_t at each timestep as

$$r_t = w^{\text{style}} \cdot r_t^{\text{style}} + w^{\text{task}} \cdot r_t^{\text{task}}, \tag{8}$$

where w^{style} and w^{task} are weights on the task and style reward functions, respectively. We use $w^{\text{style}} = 2$ and $w^{\text{task}} = 1$ for all experiments. An illustration of our complete framework is given in Fig. 2.

C. Simulation Environment

For our simulator, we use Mujoco [27] a general purpose physics engine commonly used in robotics research. We choose Mujoco for its ability to model under-actuated joints and broad modeling capabilities. It is worth noting, however, that recent work [28] uses a novel method of modeling Digit's under-actuated joints in Isaac Gym. In order to facilitate fast iteration on sim-to-real experiments, we also use MJX for the G1 robot only. For domain randomization, we randomize the dynamics of the environment during training. We adapt the domain randomization practices from Li et al. [29], which addresses two main sources of error in the sim to real gap: 1) modeling error of the robot and environment and 2) sensor noise. The parameters for the domain randomization procedure and their ranges are given in Table I. We only perform domain randomization on the G1 training.

IV. Experimental Setup

A. Baselines

We will now comprehensively outline each of our baselines and ablations discussed in section V. We compare our method to two established baselines for stylized locomotion in simulation and several ablations to highlight the important components of our method in both sim and real.

Ours: Our multi-skill policy for natural locomotion detailed in section III.

Motion Tracking Baseline (MT): A dense motion tracking reward that matches a target state frame-by-frame and has been shown to work on biped robots [29]. We follow the reward formulation of [29] and do extensive tuning on the weights to fit our robots:

$$r_t^{\text{MT}} = w^T[r_t^{q_l}, r_t^{q_u}, r_t^{\dot{q}}, r_t^{v}, r_t^{\theta}, r_t^{\omega}, r_t^{u}, r_t^{f}]$$

where $r_t^{\dot{q}} = \exp(-\rho_i \|\dot{q} - \dot{q}^*\|_2^2)$, with $\dot{q}^*$ being the reference joint velocities, and other terms are similarly defined. The ρ_i and w terms are scaling factors and reward weights that we tune for each robot. The reward terms compute errors in the lower body joint positions, upper body joint positions, global position, target velocity, global rotation, angular velocity, torque, and foot contact forces, respectively. We include the one-hot skill encoding in the policy observation as well as target joint positions for the next timestep. This baseline is discussed in our sim experiments in section V.

Energy Consumption Baseline (EC): The reward formulation from Fu et al. [30] that shows natural locomotion as an emergent property. Although this method was shown to work on quadrupeds, we investigate its application to bipedal robots. This baseline is discussed in section V-A.

Single Skill Ablation (SS-Walk): Our method without a multi-skill policy. There is no skill conditioning in the policy or discriminator. This ablation is discussed in our real-world experiments in section V-D.

No Skill-Conditioned Discriminator (NSCD): Our method without conditioning the discriminator on the skill representation during training. This ablation is done in simulation and is discussed in section V-E.

B. Motion Dataset

We aim to learn a single policy that is capable of mimicking multiple locomotion gaits from human reference motion. We take three distinct locomotion gaits from the SFU dataset [31] as our reference motions. Each reference motion is the instantiation of one skill. We retarget these motions to the joint space of each robot using Pybullet's [32] inverse kinematics solver. Specifically, the "0018_Catwalk001", "0007_Walking001", and "0005_Jogging001" motions for cat-walking, walking, and jogging gaits. Importantly, for the MT baseline we also clip reference motions to make them approximately periodic based on Eq. 10. For our method we do no such clipping, as we do not require periodic motions.

C. Training Details

Learning Algorithm Details: The policy, value function, and discriminator are all implemented as neural networks with hidden layer sizes $(1024, 1024, 512)$. We run PPO for 30K iterations using a buffer size of 4096 (approximately 120M samples) for an initial training phase without domain randomization. The policy and value networks use an adaptive learning rate initialized at 1e-5 and update based on the KL divergence, while the discriminator uses a fixed learning rate of 1e-3. To prevent mode collapse to a single behavior, we find several exploration strategies helpful. We use learnable action noise on the policy output with initial standard deviation 0.5 and initialize from a state in the reference data with probability 0.1. For the G1 only, we train in MJX [33] with 4096 parallel environments and a horizon of 32 steps for a total buffer size of 131072. For this reason, the G1 results should not be compared too strictly to the results of other robots. We deviate from the training setup of the other robots for our G1 experiments due to the added challenge of sim-to-real to iterate more quickly.

Metrics: We compare the poses error in joint space, energy efficiency, and RMSE of the target velocity. The pose error e^{pose} is computed as the ℓ_2 norm of the distance in joint space between the robot's current pose and the reference data and is meant to quantify adherence to the intended style. Because the policy motion can be out of phase with the reference motion, we compare the current joint positions q_t to a one-second sliding window $\mathbf{W}_t$ of the reference data q_t^* centered at t.

$$e^{\text{pose}}(q, q^*) = \frac{1}{T}\sum_{t=0}^{T-1} \min_{t' \in \mathbf{W}_t} \|q_{t'}^* - q_t\|_2 . \tag{9}$$

To approximate energy E, we compute the instantaneous power P_t at each timestep using the motor torques u_t and joint velocities $\dot{q}_t$, then numerically integrate to obtain energy.

$$P_t = |u_t|^T |\dot{q}_t| \tag{10}$$

$$E = \frac{1}{T}\sum_{t=0}^{T-1} \frac{1}{\lambda_{\text{env}}} \sum_{t' \in \mathbf{W}_t} P_{t'}. \tag{11}$$

To compute target velocity error, the target x (forward) velocity $v_x \in [0.5, 1.5]\ \mathrm{m\,s^{-1}}$ is sampled uniformly and the y velocity is 0 for all motions. Importantly, for MT training we also clip the reference motions to make them cyclic based on Eq. 10. For our method we do no such clipping, as we do not require cyclic motions.

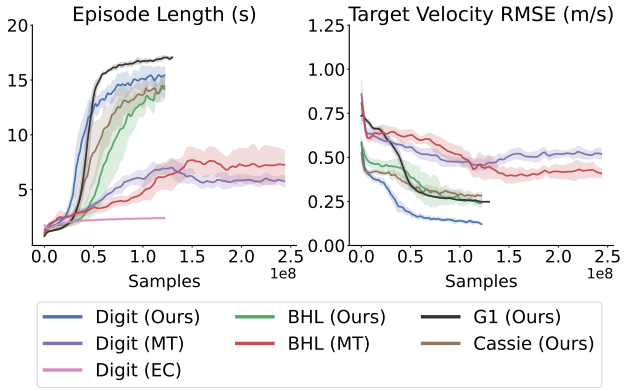


Fig. 3: Episode length and target velocity error for each method. The maximum episode length in the environment is limited to 20 seconds. Target x (forward) velocity is sampled uniformly in $[0.5, 1.5]$ and target y velocity is always 0.

V. Results

In this section we highlight the advantages of our method over the baselines in terms of learning efficiency, energy efficiency, and versatility of the resulting policy. We also discuss key design decisions in our method through ablation studies. We select the Digit and BHL platforms for our comparison to cover both digitigrade and plantigrade morphologies. Additional results for our method are provided for the G1 and Cassie platforms, while hardware experiments are conducted on G1. Due to the training differences mentioned in section IV-C, we do not compare G1 performance to that of other robots, rather we include the G1 results for completeness. For all statistics, we average across 5 seeds.

A. Learning Performance

Our method produces more robust policies that are better able to track target velocities. We compare the episode length over the course of training in Fig. 3 (left). Measuring episode length allows for a more fine-grained analysis than success rate alone at a particular episode length threshold. The maximum episode length is 600 steps (20 seconds). It is possible for the policy to learn standing still, artificially increasing the episode length, however low target velocity error shown in Fig. 3 (right) assuages these concerns.

For both Digit and BHL, our method achieves significantly higher episode length than MT, indicating that our controller produces more stable motions, while EC falls almost immediately. It is worth noting however, that EC was originally trained for 1.5 billion samples [30], while we train all our policies for 120 million samples. While it is feasible that EC could learn with many more samples, we show this highlight the advantage of our method here in terms of sample efficiency. Both our method and EC benefit from

TABLE II: Pose Error (e^{pose}), Energy Utilization (E), and Target Velocity Error for Different Robots and Motions

Method	Robot	Pose Error			Energy Utilization			Target Velocity RMSE		
		Cat-walk	Walk	Jog	Cat-walk	Walk	Jog	Cat-walk	Walk	Jog
MT	Digit	0.61 ± 0.05	0.56 ± 0.02	0.64 ± 0.08	616 ± 96	478 ± 51	683 ± 101	0.26 ± 0.07	0.24 ± 0.04	0.24 ± 0.04
Ours	Digit	**0.42 ± 0.08**	**0.37 ± 0.04**	**0.33 ± 0.01**	**493 ± 138**	**402 ± 70**	**476 ± 79**	**0.13 ± 0.03**	**0.15 ± 0.01**	**0.13 ± 0.04**
MT	BHL	0.70 ± 0.08	0.65 ± 0.07	0.70 ± 0.03	76 ± 3.7	78 ± 16	106 ± 14	0.31 ± 0.13	0.27 ± 0.08	0.26 ± 0.06
Ours	BHL	**0.50 ± 0.04**	**0.59 ± 0.09**	**0.52 ± 0.04**	**43 ± 6.3**	**47 ± 6.2**	**63 ± 12**	**0.26 ± 0.12**	**0.25 ± 0.07**	**0.19 ± 0.04**
Ours	G1	0.82 ± 0.08	0.87 ± 0.11	0.88 ± 0.2	138 ± 13	139 ± 13	129 ± 9.3	0.24 ± 0.043	0.25 ± 0.05	0.25 ± 0.04
	Cassie	0.19 ± 0.02	0.14 ± 0.01	0.21 ± 0.01	121 ± 17	81 ± 11	118 ± 7.3	0.21 ± 0.07	0.32 ± 0.10	0.25 ± 0.04

simple reward functions, yet ours learns with a fraction of the samples. Because EC fails to learn in the desired number of samples, the rest of our results focus on comparing our method to MT.

In MT and our method, Digit achieves higher episode length than BHL while also following the target velocity more faithfully. This trend suggests that digitigrade morphologies may be more amenable to learned natural locomotion gaits, particularly at the relatively high speeds present in our experiments at which digitigrade posture has known benefits in terms of energy efficiency [7].

B. Energy Utilization

Motion priors trained to mimic human motion yield energy-efficient locomotion gaits. In addition to providing a robust controller for locomotion, our method adheres to natural-looking gaits and does so in an energy-efficient manner. In Table II, we compare the pose error, energy efficiency, and target velocity error of our method to MT for Digit and BHL, and additionally provide results for Cassie and G1. We show the mean and standard deviation across 5 seeds. Importantly, we train MT for twice as long as our method to give it an opportunity to reach convergence, and evaluate the best checkpoint in terms of episode return for each seed. Note that Table II suggests lower target velocity errors than in Fig. 3 because the table computes statistics on successful episodes only.

Our method achieves lower pose error compared to MT for most motions, demonstrating our method's effectiveness at producing more natural-looking gaits. We bold statistics that achieve better mean performance, but note that on 5 seeds some results may not be statistically significant. Interestingly, our method also consumes significantly less energy than MT for BHL (and also potentially for Digit), despite MT explicitly optimizing for torques and contact forces whereas our method contains no such reward terms. This demonstrates the inherent efficiency present in human motion and the ability of learned motion priors to encode energy-efficient motion.

C. Velocity-Conditioning and Smooth Motion Transition

Motion priors allow for smooth motion transitions between different target velocities and different skills. Transitions between velocities: It is desirable to be able to move at different velocities while either keeping the motion the same or changing motions after reaching a certain velocity, since different motions are more energy efficient at different speeds [30]. However, using an energy consumption reward with a naive conditioning on the target velocity was shown in prior work to result in mode collapse [30]. This was resolved using a more complex approach that involved policy distillation with several single-task policies each trained with a constant target velocity. Alternatively, with MT it is perhaps not surprising that tracking different target velocities is difficult, since this causes the gait to go out of sync with the reference motion. These limitations are not present under our framework, as motion priors objective imposes less restrictive constraints than that of MT, which relies on frame-by frame matching, or EC, which imposes energy constraints. This allows more freedom for the robot to deviate from the training distribution to match target velocities, indicated by significantly lower target velocity error compared to MT for all Digit and BHL motions in Table II.

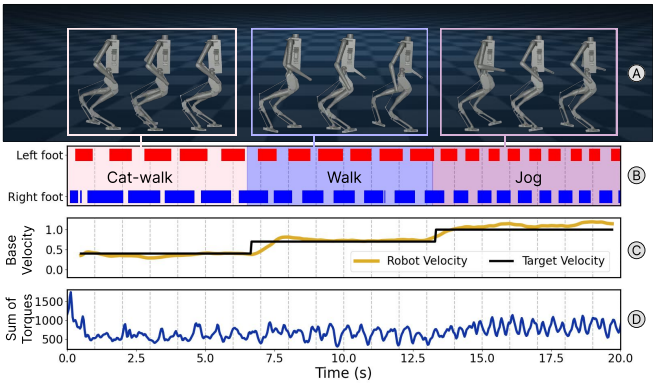
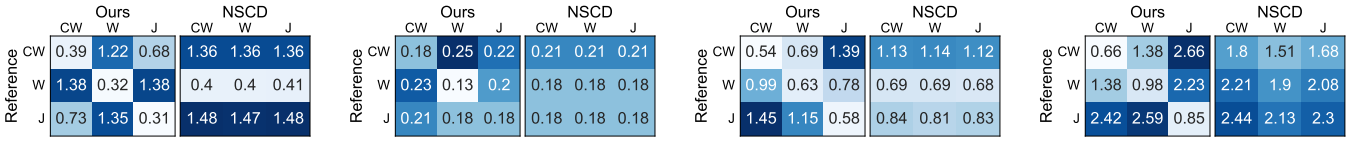


Fig. 4: **A:** Key frames of Digit showing each motion. Cat-walk demonstrates stationary hands at the hip and short steps, walk exhibits swinging arms and longer strides, and jog demonstrates the arms lifted and tucked in. **B:** Gait patterns showing transitions between different motions. **C:** Robot base velocity showing correct tracking of the target velocity. **D:** Sum of the absolute values of motor torques showing no sudden deviations, indicating smooth motion transitions.

Transition between motions: Our method is able to transition smoothly between different motions and velocities, despite target velocity changes within an episode not occurring during training. Fig. 4 shows an example of a Digit policy that transitions from cat-walking at $0.4\,\mathrm{m\,s^{-1}}$ to walking at $0.7\,\mathrm{m\,s^{-1}}$ to jogging at $1\,\mathrm{m\,s^{-1}}$. Our method accurately tracks the changing target velocity and can be seen visually to smoothly transition between motions. Moreover, the torque profile of the motors does not demonstrate large



(a) Digit (b) Cassie (c) BHL (d) G1

Fig. 5: Pose errors of policy motions trained in our method (left of each sub-figure) and NSCD (right of each sub-figure) compared against each reference motion for Digit (a), Cassie (b), BHL (c), and G1 (d). CW = Cat-walk, W = Walk, J = Jog.

	Sim	Sim-DR	Real
SS-Walk	0.98 ± 0.01	0.87 ± 0.04	0.8
Ours	1.0 ± 0.0	0.96 ± 0.01	1.0

TABLE III: Robustness experiments in simulation (Sim), simulation with domain randomization (Sim-DR), and real.

deviations, providing further validation of smooth motion transition.

D. Real-World Experiments

In order to facilitate sim-to-real transfer of the policy, we made the following adjustments to the training procedure: We use additional task reward terms of the form $\exp(-\rho \|\cdot\|^2)$ that reward low angular velocity, torque, joint position deviation from a nominal pose (i.e. $\|q - q_{\text{nominal}}\|$), and action rate (i.e. $\|a_t - a_{t-1}\|$). We also replaced the waist and wrist motion data with small Gaussian noise, effectively biasing these joints toward 0 without trivializing the discriminator's objective. Simply setting these values to 0 would allow the discriminator to easily distinguish policy versus reference data, resulting in extremely low style reward.

We deploy our natural locomotion controller to the real world using the G1 robot. Fig. 1) shows our result in the real world. The robot was able to perform the walking and jogging skills. The jogging skill did not end up learning to keep one foot in the air like a human might do. We hypothesize that the robot learned this behavior due it it being more stable, and that explicit foot penalties in the task reward could correct this. The robot is also able to smoothly transition between different skills by simply changing the skill encoding during execution. This would be challenging for MT because the motion would be out of sync with the reference.

Multi-Skill RL as a policy robustness technique: We validate our hypothesis that Multi-Skill RL can be used to improve robustness policy robustness in both sim and real. We compare the success rate of our strongest single-skill policy (SS-Walk) to that of a multi-skill policy (Ours) on five trials of each. We test in simulation without domain randomization (Sim), simulation with domain randomization sampled from the ranges in Table I (Sim-DR), and in real. A trial is deemed successful if the policy lasts 20 seconds in simulation and 5 seconds in real (due to space limitations). We report the average success rate of 100 trials in simulation and 5 in real to prevent excess wear and tear on the robot. For simulation, we average the results of policies trained on 5 random seeds. For real, we take the best policy across these seeds. Table III We find that Multi-Skill RL offers a notable robustness improvement over a single walking motion. We hypothesize that this is because learning Multi-Skill policies inherently involves more exploration during training.

E. Task-Conditioned Discriminator Prevents Mode Collapse

Conditioning the discriminator on the skill encoding is essential for reliably learning a multi-skill policy. In simulation, we ablate conditioning the discriminator on the skill encoding (but still conditioning the policy) and report results across 5 random seeds. Fig. 5 shows the mean pose errors of our method and our method with no skill-conditioning on the discriminator (NSCD) compared against the reference motion. When using a skill-conditioned discriminator, the pose error for the policy-generated motion is lowest with respect to the correct reference motion. In the NSCD case, the pose error is lowest for all policy-generated motions when compared against one of the reference motions, indicating mode collapse to a single motion. Visually, we confirmed that the policy only learned to imitate a single motion no matter which skill encoding was provided to the policy. For Cassie, the difference in pose error between correct and incorrect motions is not as large due to the lack of an upper body.

We believe mode collapse occurs because the discriminator does not necessarily learn a multi-modal distribution of motions. In the NSCD case, learning only a single motion does not harm the discriminator's performance. In contrast, by conditioning on the skill encoding, the discriminator is able to match skill encodings in the policy-generated data with those in the reference data. The effect is that learning a single motion is now harmful to the discriminator's performance. The policy is then forced to learn a multi-modal distribution of motions to maximize the style reward. We therefore posit that a skill-conditioned discriminator is necessary for learning multi-skill policies with our framework.

VI. Conclusion and Future Work

This work represents, to our knowledge, the first use of Multi-Skill stylized locomotion on a real humanoid robot. We provide detailed benchmarking and analysis on several bipedal robot platforms and provide evidence that Multi-Skill RL is useful for improving policy robustness. In contrast to prior work based on complex motion-tracking rewards, our method offers a more simple reward function, is more energy efficient even without explicitly optimizing for energy, and can smoothly transition between motions and speeds. An

interesting direction for future work could involve investigating which types of skills are more useful than others for improving robustness and understanding how the benefits scale with larger motion datasets.

REFERENCES

[1] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, jan 2022. [Online]. Available: https://doi.org/10.1126%2Fscirobotics.abk2822

[2] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," 2021.

[3] 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*, p. 02783649241285161, 2024.

[4] X. Gu, Y.-J. Wang, X. Zhu, C. Shi, Y. Guo, Y. Liu, and J. Chen, "Advancing humanoid locomotion: Mastering challenging terrains with denoising world model learning," *arXiv preprint arXiv:2408.14472*, 2024.

[5] T. He, W. Xiao, T. Lin, Z. Luo, Z. Xu, Z. Jiang, J. Kautz, C. Liu, G. Shi, X. Wang *et al.*, "Hover: Versatile neural whole-body controller for humanoid robots," *arXiv preprint arXiv:2410.21229*, 2024.

[6] J. Long, J. Ren, M. Shi, Z. Wang, T. Huang, P. Luo, and J. Pang, "Learning humanoid locomotion with perceptive internal model," *arXiv preprint arXiv:2411.14386*, 2024.

[7] A. M. Pagano, A. M. Carnahan, C. T. Robbins, M. A. Owen, T. Batson, N. Wagner, A. Cutting, N. Nicassio-Hiskey, A. Hash, and T. M. Williams, "Energetic costs of locomotion in bears: is plantigrade locomotion energetically economical?" *Journal of Experimental Biology*, vol. 221, no. 12, p. jeb175372, 06 2018. [Online]. Available: https://doi.org/10.1242/jeb.175372

[8] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "AMP," *ACM Transactions on Graphics*, vol. 40, no. 4, pp. 1–20, jul 2021. [Online]. Available: https://doi.org/10.1145%2F3450626.3459670

[9] Q. Zhang, P. Cui, D. Yan, J. Sun, Y. Duan, G. Han, W. Zhao, W. Zhang, Y. Guo, A. Zhang, and R. Xu, "Whole-body humanoid robot locomotion with human reference," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 11 225–11 231.

[10] X. Huang, Y. Chi, R. Wang, Z. Li, X. B. Peng, S. Shao, B. Nikolic, and K. Sreenath, "Diffuseloco: Real-time legged locomotion control with diffusion from offline datasets," *arXiv preprint arXiv:2404.19264*, 2024.

[11] Q. Liao, B. Zhang, X. Huang, X. Huang, Z. Li, and K. Sreenath, "Berkeley humanoid: A research platform for learning-based control," 2024.

[12] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Proceedings of The 2nd Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Billard, A. Dragan, J. Peters, and J. Morimoto, Eds., vol. 87. PMLR, 29–31 Oct 2018, pp. 916–926. [Online]. Available: https://proceedings.mlr.press/v87/iscen18a.html

[13] T.-Y. Yang, T. Zhang, L. Luu, S. Ha, J. Tan, and W. Yu, "Safe reinforcement learning for legged locomotion," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 2454–2461.

[14] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," 2021.

[15] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, oct 2020. [Online]. Available: https://doi.org/10.1126%2Fscirobotics.abc5986

[16] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7309–7315.

[17] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, "Allsteps: curriculum-driven learning of stepping stone skills," in *Computer Graphics Forum*, vol. 39, no. 8. Wiley Online Library, 2020, pp. 213–224.

[18] W. Yu, G. Turk, and C. K. Liu, "Learning symmetric and low-energy locomotion," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 1–12, jul 2018. [Online]. Available: https://doi.org/10.1145%2F3197517.3201397

[19] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[20] 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.

[21] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, and M. Inaba, "Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2024, p. 13107–13114. [Online]. Available: http://dx.doi.org/10.1109/ICRA57147.2024.10610449

[22] J. Zhang, Z. Li, M. Lu, and C. Gan, "Conditional adversarial motion priors by a novel retargeting method for versatile humanoid robot control," *International Journal of Adaptive Control and Signal Processing*, vol. n/a, no. n/a, 2024. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/acs.3915

[23] R. Grandia, E. Knoop, M. A. Hopkins, G. Wiedebach, J. Bishop, S. Pickles, D. Müller, and M. Bächer, "Design and control of a bipedal robotic character," in *Proceedings of Robotics: Science and Systems (RSS)*. Disney Research Switzerland and Walt Disney Imagineering R&D USA, 2024.

[24] X. Cheng, Y. Ji, J. Chen, R. Yang, G. Yang, and X. Wang, "Expressive whole-body control for humanoid robots," *arXiv preprint arXiv:2402.16796*, 2024.

[25] Q. Zhang, C. Weng, G. Li, F. He, and Y. Cai, "Hilo: Learning whole-body human-like locomotion with motion tracking controller," 2025. [Online]. Available: https://arxiv.org/abs/2502.03122

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[27] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[28] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Real-world humanoid locomotion with reinforcement learning," *arXiv:2303.03381*, 2023.

[29] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2811–2817.

[30] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Conference on Robot Learning (CoRL)*, 2021.

[31] G. J. Ying and K. Yin, "Sfu motion capture database." [Online]. Available: http://mocap.cs.sfu.ca/

[32] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.

[33] DeepMind, "MuJoCo XLA (MJX)." [Online]. Available: https://mujoco.readthedocs.io/en/stable/mjx.html