

# Adversarial Motion Priors Make Good Substitutes for Complex Reward Functions

Alejandro Escontrela<sup>γ,σ</sup>, Xue Bin Peng<sup>γ</sup>, Wenhao Yu<sup>σ</sup>, Tingnan Zhang<sup>σ</sup>

Atil Iscen<sup>σ</sup>, Ken Goldberg<sup>γ</sup>, Pieter Abbeel<sup>γ</sup>

$\gamma$ : UC Berkeley,  $\sigma$ : Google Brain

$\gamma$ : {escontrela, xbpeng, goldberg, pabbeel}@berkeley.edu

$\sigma$ : {magicmelon, tingnanzhang, atil}@google.com

**Abstract**— Training a high-dimensional simulated agent with an under-specified reward function often leads the agent to learn physically infeasible strategies that are ineffective when deployed in the real world. To mitigate these unnatural behaviors, reinforcement learning practitioners often utilize complex reward functions that encourage physically plausible behaviors. However, a tedious labor-intensive tuning process is often required to create hand-designed rewards which might not easily generalize across platforms and tasks. We propose substituting complex reward functions with “style rewards” learned from a dataset of motion capture demonstrations. A learned style reward can be combined with an arbitrary task reward to train policies that perform tasks using naturalistic strategies. These natural strategies can also facilitate transfer to the real world. We build upon Adversarial Motion Priors – an approach from the computer graphics domain that encodes a style reward from a dataset of reference motions – to demonstrate that an adversarial approach to training policies can produce behaviors that transfer to a real quadrupedal robot without requiring complex reward functions. We also demonstrate that an effective style reward can be learned from a few seconds of motion capture data gathered from a German Shepherd and leads to energy-efficient locomotion strategies with natural gait transitions.

## I. INTRODUCTION

Developing controllers for high-dimensional continuous control systems such as legged robots has long been an area of study. Early work in this field focused on developing approximate dynamics models of a system and then using trajectory optimization algorithms to solve for the actions that lead an agent to achieving a desired goal [1]–[4]. However, the resulting controllers tend to be highly specialized for a particular task, limiting their ability to generalize across more diverse tasks or environments. More recently, there has been a surge in algorithms that use reinforcement learning (RL) to learn locomotion behaviors [5]–[9]. This approach proved highly effective in simulation [10], but this success did not translate to the real world due to challenges associated with overcoming the simulation to reality gap.

One of the main challenges inhibiting RL approaches from being more effective in the real world is related to the aggressive and overly-energetic behaviors that are often learned by RL agents trained using under-specified reward functions. As an example, a legged RL agent trained with a reward that encourages forward velocity will often learn

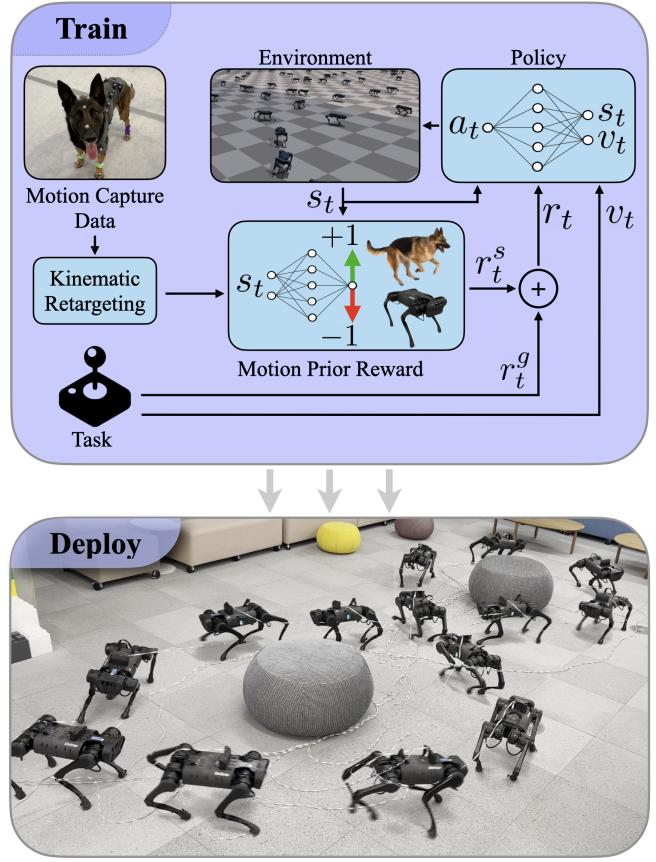


Fig. 1. Training with Adversarial Motion Priors encourages the policy to produce behaviors which capture the essence of the motion capture dataset while satisfying the auxiliary task objective. Only a small amount of motion capture data is required to train the learning system (4.5 seconds in our experiments).

a control policy that exploits flailing of the limbs or high-impulse contacts, and other inaccurate simulator dynamics, to achieve forward movement. Such behaviors are unlikely to be effective when transferred to a real robot due to actuator limits and potential damage to the robot. To overcome the issues posed by reward under-specification, researchers have investigated task-specific action spaces [12], [13], complex style reward formulations [6]–[8], [14], and curriculum learning [15], [16]. These approaches achieve state-of-the-art

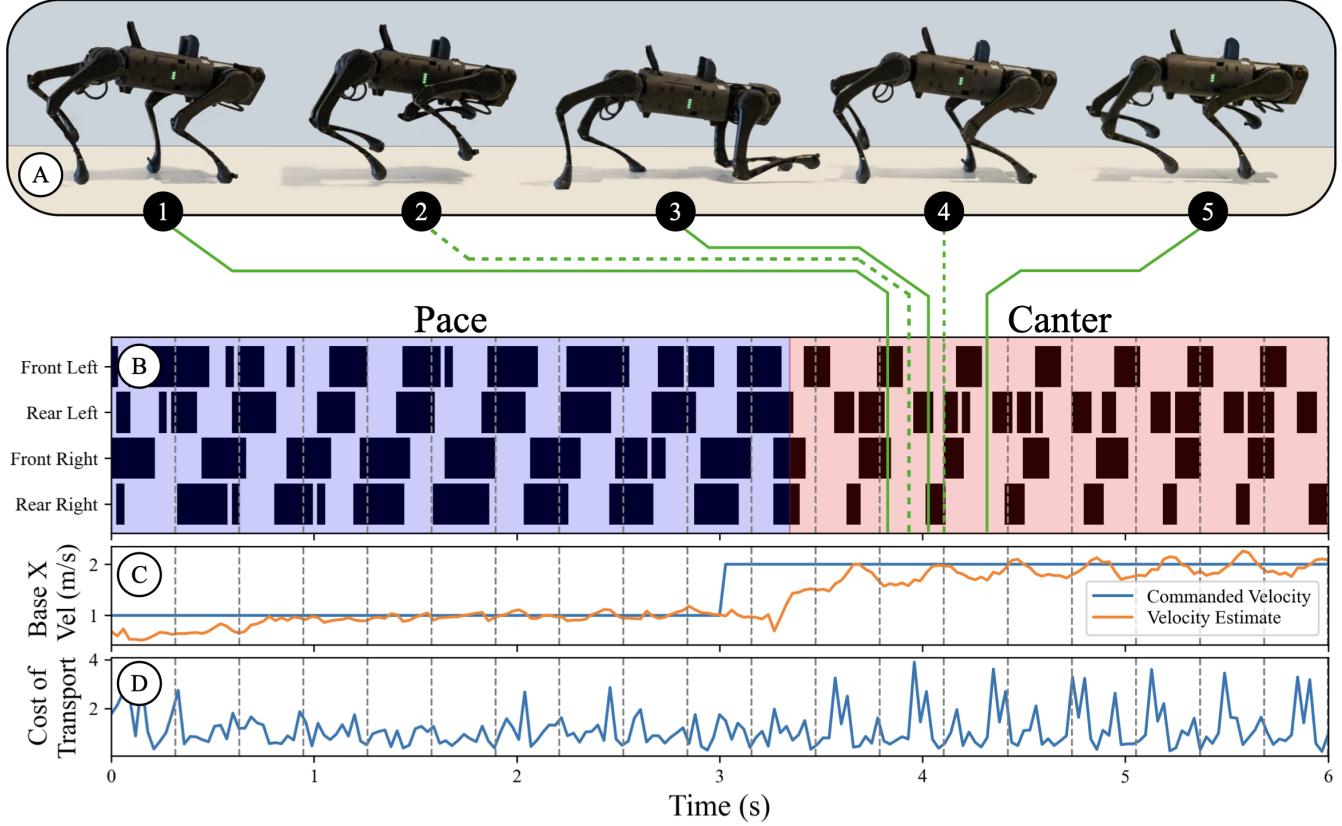


Fig. 2. Key frames, gait pattern, velocity tracking, and energy-efficiency of the robot dog throughout a trajectory **A**: Key frames of A1 during a canter motion overlaid on a plain background for contrast. **B**: Gait diagram indicating contact timing and duration for each foot in black. Training with Adversarial Motion Priors enables the policy to synthesize behaviors which lead to natural gait transitions at different velocities. **C**: Plot of commanded forward velocities and estimated velocities during the rollout. **D**: Estimated Cost of Transport (COT) during the rollout. While pacing the COT remains constant with small oscillations. However, when the robot enters a canter phase the COT exhibits spikes corresponding to the robot pushing off its hind legs and troughs corresponding to the flight phase where energy consumption is low. This gait transition phenomenon closely relates to the behavior of quadrupedal mammals, which modulate their gait according to their speed of travel, leading to minimal energy consumption [11].

results in locomotion, but defining custom action spaces and hand-designed 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.

In the realm of computer graphics, *Adversarial Motion Priors* (AMP) [17] 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 [18]. A simple task-specific reward can then be specified in conjunction with the style reward to produce policies that match the style of the dataset while performing the specified task (Fig. 1). Animators have leveraged this flexible approach to animate characters that perform complex and highly dynamic tasks while remaining human-like. However, the viability of this approach to train policies for the real world has not been studied, even though it could provide a promising alternative to the hand-defined complex rewards that are prevalent in recent literature [6], [7]. In this work, we substitute complex hand-specified style reward formulations with a motion prior

learned from a few seconds of German Shepherd motion capture data. We propose the following contributions:

- We introduce a learning framework that leverages small amounts of motion capture data to encode a style reward that – when trained in conjunction with an auxiliary task objective – produces policies that can be effectively deployed on a real robot.
- We study the energy efficiency of agents trained with complex style reward formulations [5], [6], [19] and policies trained with Adversarial Motion Priors. We find that training policies with motion priors results in a lower Cost of Transport, and analyze the benefits of leveraging the energy-efficient prior provided by the data. We also find that policies trained with motion priors produce natural gait transitions which result in more energy-efficient motions across different speeds<sup>1</sup>.

<sup>1</sup>Videos and the project repository can be found at: <https://bit.ly/3hpvbD6>

## II. RELATED WORK

### A. Deep Reinforcement Learning for Robot Control

Recent works in robotics have shown promising results in applying Deep Reinforcement Learning (DRL) to a variety of robotic control tasks such as manipulation [20]–[23], locomotion [5], [6], [8], [9], and navigation [24], [25]. DRL provides an effective paradigm for automatically synthesizing control policies for a given objective function, thus avoiding the need for manually designed controllers. However, controllers trained using DRL often lead to jerky, unnatural behaviors that may maximize the objective function, but may not be suitable for real-robot deployment [10], [26]. As a result, manually-designed priors are often required to regularize the policy’s behavior. For example, legged locomotion researchers have investigated complex reward functions [5], [8], [27], task-specific action spaces [12], [13], [28], [29], or curriculum learning [15], [16] to encourage robot behavior that is amenable to physical deployment. Despite the compelling results in these works, these approaches are often task-specific and require substantial effort to tune for each skill of interest. In this work, we explore the idea of automatically learning these behavioral priors directly from reference motion data.

### B. Motion Imitation

Imitating reference motion data provides a general approach for developing controllers for a wide range of skills that would otherwise be difficult to manually encode into controllers [30]–[33]. These techniques often utilize some form of motion tracking, where a controller imitates a desired motion by explicitly tracking the sequence of target poses specified by a reference trajectory [34]–[37]. In simulated domains, motion tracking combined with reinforcement learning has been shown to be highly effective for reproducing a large corpus of complex and dynamic motor skills [38]–[41]. While motion tracking can be very effective for imitating individual motion clips, the tracking objective tends to constrain a controller to closely follow the reference motion, which can limit an agent’s ability to develop more versatile and diverse behaviors as needed to fulfill auxiliary task objectives. Furthermore, it can be difficult to apply tracking-based techniques to imitate behaviors from diverse motion datasets, often requiring substantial overhead in the form of motion planners and task-specific annotation of the motion clips [42]–[45]. In this work, we utilize a more flexible motion imitation approach based on adversarial imitation learning, which allows our system to shape the behavior of an agent using unstructured motion datasets, while also affording the agent more flexibility to develop new behaviors as needed to achieve task objectives.

### C. Adversarial Imitation Learning

Adversarial imitation learning provides a flexible and scalable approach for imitating behaviors from diverse demonstration datasets (e.g. reference motions) [46]–[48]. Rather than explicitly tracking individual motion clips, adversarial techniques aim to learn policies that match the

state/trajectory distribution of the dataset [49], [50], which can provide the agent more flexibility in composing and interpolating between behaviors shown in the dataset. This is done by training an adversarial discriminator to differentiate between behaviors produced by a policy and behaviors depicted in the demonstration data. The discriminator then serves as the style reward for training a control policy to imitate the demonstrations. While these methods have shown promising results in low-dimensional domains [48], [51], when applied to high-dimensional continuous control tasks, the quality of the results produced by these methods generally falls well behind state-of-the-art tracking-based techniques [52], [53]. Recently, *Peng et al.* [17] proposed *Adversarial Motion Priors* (AMP), which combines adversarial imitation learning with auxiliary tasks objectives, thereby enabling simulated agents to perform high-level tasks, while imitating behaviors from large unstructured motion datasets. We will leverage this adversarial technique to learn locomotion skills for legged robots. We show that the learned motion prior leads to more natural, physically plausible, and energy-efficient behaviors, which are then more amenable to transfer from simulation to a real-world robot.

## III. METHOD

### A. Background and Problem Formulation

We model the problem of learning legged locomotion as a Markov Decision Process (MDP):  $(\mathcal{S}, \mathcal{A}, f, r_t, p_0, \gamma)$ , where  $\mathcal{S}$  is the state space,  $\mathcal{A}$  is the action space,  $f(s, a)$  is the system dynamics,  $r_t(s, a, s')$  is the reward function,  $p_0$  is the initial state distribution, and  $\gamma$  is the discount factor. The goal of Reinforcement Learning (RL) is to find the optimal parameters  $\theta$  of a policy  $\pi_\theta : \mathcal{S} \mapsto \mathcal{A}$  to maximize the expected discounted return  $J(\theta) = \mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{T-1} \gamma^t r_t \right]$ .

In this work, we are interested in learning a locomotion controller that is agile and controllable. As such, we design a task reward function for encouraging the robot to track a command velocity  $\vec{v}_t = [v_t^x, v_t^y, \omega_t]$  at time  $t$ , where  $v_t^x$  and  $v_t^y$  are the desired forward and lateral velocities specified in the base frame, and  $\omega_t$  is a desired global yaw rate. In particular, we define the reward function to be:

$$r_t^g = w^v \exp(-\|\hat{v}_t^{xy} - \vec{v}_t^{xy}\|) + w^\omega \exp(-|\hat{\omega}_t^z - \omega_t^z|) \quad (1)$$

where  $\hat{v}_t^{xy}$  and  $\hat{\omega}_t^z$  are the desired linear and angular velocity. The desired base forward velocity  $v_t^x$ , base lateral velocity  $v_t^y$ , and global yaw rate  $\omega_t$  are sampled randomly from the ranges  $(-1, 2) \frac{\text{m}}{\text{s}}$ ,  $(-0.3, 0.3) \frac{\text{m}}{\text{s}}$ , and  $(-1.57, 1.57) \frac{\text{rad}}{\text{s}}$ , respectively. Training with this reward grants a high degree of controllability over the robot’s movement, and causes the resulting controller to exhibit locomotion behaviors at different speeds. However, as we will show in our experiments, training with only the task reward  $r_t^g$  can lead to undesired behaviors such as violent vibrations due to the under-specified nature of the reward. To tackle this problem, we will regularize behaviors of the policy using a data-driven motion prior acquired through adversarial imitation learning.

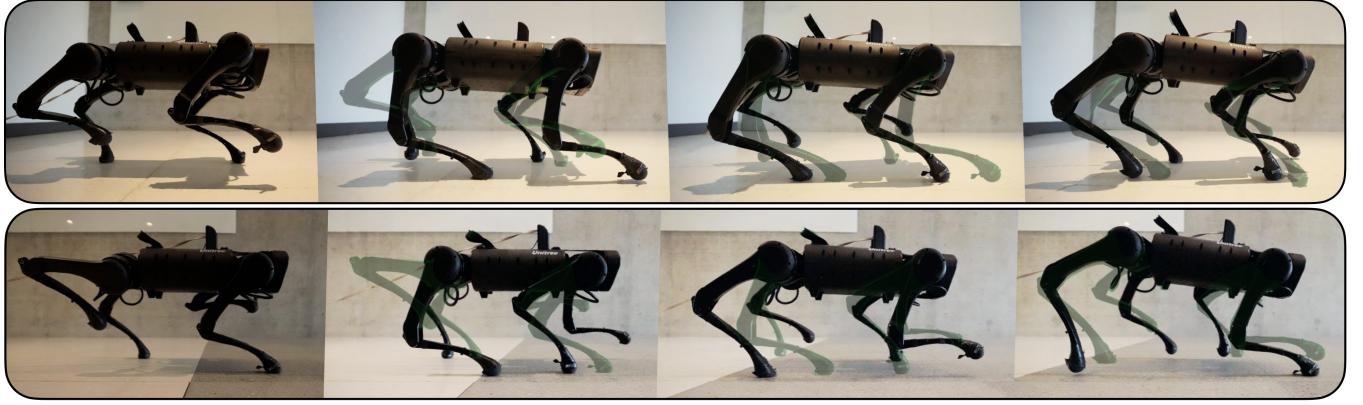


Fig. 3. An agent trained with Adversarial Motion Priors extracts the naturalistic locomotion strategies found in the dataset and can change its gait based on the desired velocity. Top: when commanded to move at a low forward velocity ( $0.8 \frac{m}{s}$ ), the agent select a pacing gait. Bottom: when the commanded forward velocity increases to ( $1.7 \frac{m}{s}$ ), the agent switches to a trotting gait. The green low-opacity overlaid images show the previous frame for reference.

### B. Adversarial Motion Priors as Style Rewards

In the adversarial imitation learning setting, we consider reward functions that consist of a “style” component  $r_t^s$  and a task component  $r_t^g$ , such that

$$r_t = w^g r_t^g + w^s r_t^s. \quad (2)$$

The style reward  $r_t^s$  encourages the agent to produce behaviors that have the same style as the behaviors from a reference dataset. Whereas the task reward is specified by the user, the style reward is learned from a dataset of reference motion clips. Formally, we define a discriminator as a neural network parameterized by  $\phi$ . The discriminator  $D_\phi$  is trained to predict whether a state transition  $(s, s')$  is a real sample from the dataset or a fake sample produced by the agent. We borrow the training objective for the discriminator proposed in AMP [17]:

$$\begin{aligned} \arg \min_{\phi} & \mathbb{E}_{(s, s') \sim \mathcal{D}} [(D_\phi(s, s') - 1)^2] \\ & + \mathbb{E}_{(s, s') \sim \pi_\theta(s, a)} [(D_\phi(s, s') + 1)^2] \\ & + \frac{w_{gp}}{2} \mathbb{E}_{(s, s') \sim \mathcal{D}} [\|\nabla_\phi D_\phi(s, s')\|^2], \end{aligned} \quad (3)$$

where the first two terms encourages the discriminator to distinguish whether a given input state transition is from the reference dataset  $\mathcal{D}$  or produced by the agent. The least squares GAN formulation (LSGAN) used in Eq. 3 has been shown to minimize the Pearson divergence  $\chi^2$  divergence between the reference data distribution and the distribution of transitions produced by the agent. The final term in the objective is a gradient penalty, with coefficient  $w_{gp}$ , which penalizes nonzero gradients on samples from the dataset. The gradient penalty mitigates the discriminator’s tendency to assign nonzero gradients on the manifold of real data samples, which can cause the generator to overshoot and move off the data manifold. The zero-centered gradient penalty has been shown to reduce oscillations in GAN training, and improve training stability [54]. The style reward is then defined by:

$$r_t^s(s_t, s_{t+1}) = \max[0, 1 - 0.25(D(s, s') - 1)^2], \quad (4)$$

where an additional offset and scaling is applied to bound the reward in the range  $[0, 1]$ . The style reward and the task reward are then combined into the composite reward in Eq. 2. We optimize the parameters of the policy  $\pi_\theta$  to maximize the total discounted return of Eq. 2, and the parameters of the discriminator  $D_\phi$  to minimize the objective presented in Eq. 3.

The process of training the policy and discriminator is shown in Fig. 1. First, the policy takes a step in the environment to produce a state transition  $(s, s')$ . This state transition is fed to the discriminator  $D_\phi(s, s')$  to obtain the style reward  $r_t^s$ . The state transition is also used to compute the task reward  $r_t^g$ . Finally, the combined reward and states from the environment and reference motion dataset are used to optimize the policy and discriminator.

### C. Motion Capture Data Preprocessing

The raw motion capture data is a time-series of keypoints corresponding to various frames in the subject’s motion. In this work, we use the German Shepherd motion capture data provided by Zhang and Starke *et al.* [55]. The dataset consists of short clips of a German Shepherd pacing, trotting, cantering, and turning in place, with a total duration of 4.5 seconds. We follow the process described by Peng *et al.* [38] to retarget the German Shepherd motion to the morphology of the A1 quadrupedal robot. We use inverse kinematics to obtain the joint angles and compute the end-effector positions using forward kinematics. We compute the joint velocities, base linear velocities, and angular velocities using finite differences. These quantities define the states in the motion capture dataset  $\mathcal{D}$ . State transitions are sampled from  $\mathcal{D}$  to serve as real samples for training the discriminator. When training in simulation, we also use reference state initialization [38] at the start of each episode to initialize the agent from states randomly sampled from  $\mathcal{D}$ .

### D. Model Representation

We parametrize the policy as a shallow MLP with hidden dimensions of size [512, 256, 128] and exponential linear



Fig. 4. By using Adversarial Motion Priors, the policy can deviate from the reference motion data to satisfy the desired velocity commands and navigate carefully through a route with sharp turns.

unit activation layers. The policy outputs both the mean and standard deviation of the output distribution from which target joint angles are sampled. The standard deviation is initialized to  $\sigma_i = 0.25$ . The policy is queried at 30Hz, and the target joint angles are fed to PD controllers which compute the motor torques. The policy is conditioned on an observation  $o_t$  derived from the state, which contains the robot's joint angles, joint velocities, orientation, and previous actions. The discriminator is an MLP with hidden layers of size [1024, 512] and exponential linear unit activation layers.

#### E. Domain Randomization

We apply domain randomization to facilitate transfer of learned behaviors from simulation to the real world [56]. Namely, we randomize the terrain friction, base mass, PD controller gains, and perturb the robot's base velocity by adding a sampled velocity vector to the current base velocity at random intervals during training. The randomization variables and the range of the uniform distribution from which they are sampled are shown in Table I.

TABLE I  
RANDOMIZED SIMULATION PARAMETERS.

Parameter	Randomization Range
Friction	[0.35, 1.65]
Added Base Mass	[−1.0, 1.0] kg.
Velocity Perturbation	[−1.3, 1.3] m/s
Motor Gain Multiplier	[0.85, 1.15]

#### F. Training

We use a distributed PPO [57] implementation across 5280 simulated environments in Isaac Gym [19], [58]. The policy and discriminator are trained for 4 billion environment steps,

constituting approximately 4.2 years worth of simulation data gathered over 16 hours on a single Tesla V100 GPU. For each training iteration, we collect a batch of 126,720 state transitions  $(s, s')$  and optimize the policy and discriminator for 5 epochs with minibatches containing 21,120 transitions. We automatically tune the learning rate to maintain a desired KL divergence of  $KL_{\text{desired}} = 0.01$  using adaptive LR scheme proposed by Schulman *et al.* [57].

The discriminator is optimized with the supervised learning objective in Eq. 3. We use the Adam optimizer and a gradient penalty weight of  $w^{\text{gp}} = 10$ . The style reward and task reward weights are  $w^s = 0.65$  and  $w^g = 0.35$ , respectively.

## IV. EXPERIMENTS

In this section we perform quantitative and qualitative analysis of policies trained using various style reward formulations. Namely, we compare the complex reward introduced by Rudin *et al.* [19] (which is similar to the reward formulations used in other state-of-the-art systems [5]–[7]) and our approach, which learns a style reward from 4.5 seconds of German Shepherd Motion Capture data. We also include an analysis of policies trained with no style reward. Policies trained with no style reward are analyzed solely in simulation, as the behaviors exhibited by these policies are too violent to deploy on a real robot (Fig. 5). We seek to answer the following questions:

- Do policies trained with Adversarial Motion Priors achieve similar task performance as policies trained with complex style reward formulations?
- How energy-efficient are policies trained with the various style reward formulations?
- What is the qualitative performance of policies trained with Adversarial Motion Priors when deployed in the real world?

#### A. Task Completion and Energy Efficiency in Simulation

First, we train policies for a velocity tracking task (Eq. 1), where the goal is for a policy to closely track the target velocity specified by the user. Here, we compare the performance of policies trained with three reward functions in simulation: no style reward (task reward only), the adversarial motion prior style reward in Eq. 4, and the complex style reward proposed by Rudin *et al.* [19]. The complex style reward is composed of 13 style terms, most of which are designed to penalize behaviors that emerge from an under-specified reward function. Each reward component and the associated scaling factors are listed in the appendix (Table III).

We also estimate the Cost of Transport (COT) for each of these policies. COT is a dimensionless quantity commonly used in the field of legged locomotion, as it allows for energy-efficiency comparisons of dissimilar robots or controllers. We utilize the COT to measure the efficiency of different baselines at different speeds. We define the mechanical COT as:  $\frac{\text{Power}}{\text{Weight} \times \text{Velocity}} = \sum_{\text{actuators}} [\tau \dot{\theta}]^+ / (W \|v\|)$ , where  $\tau$  is the joint torque,  $\dot{\theta}$  is the motor velocity,  $W$  is the robot's weight, and  $\|v\|$  is the velocity.

We find that a policy trained using AMP successfully tracks the desired forward velocity commands while exhibiting a lower COT than competing baselines (Table II). Meanwhile, a policy trained with no style reward learns to move by vibrating its legs at high speeds with large torques (Fig. 5), producing high-impulse contacts with the ground. While this behavior leads to high tracking accuracy, applying such a control strategy to a real robot is infeasible due to the violent nature of the resulting motions and risk of damaging the robot. As shown in Table II, policies trained with the hand-designed style reward exhibit a relatively low COT, varying between 1.37 and 1.65. Our method produces a lower COT, varying between 0.93 and 1.12 for different target speeds. Meanwhile, a method trained with no style reward exhibits an extremely high COT due to the high torques and motor velocities that emerge from the jittery behaviors.

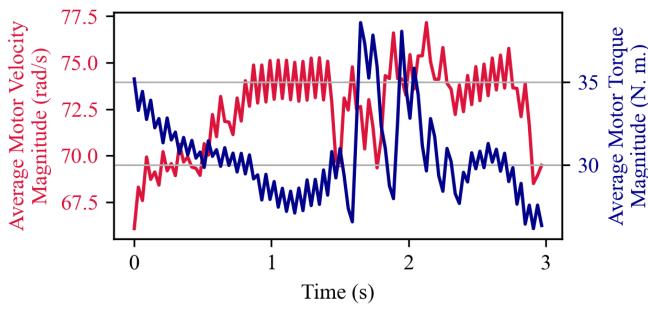


Fig. 5. The policy trained with no style reward learns to exploit inaccurate simulator dynamics and violently vibrates the simulated robot’s feet on the ground to move. The high motor velocities and torques make it impossible to deploy this control strategy on the real robot.

The energy efficiency of policies trained with AMP can likely be attributed to the policy extracting energy-efficient motion priors from the reference data. Millions of years of evolution has endowed dogs with energy-efficient locomotion behaviors. Training with AMP enables the policy to extract some of these energy-efficient strategies from the data. Additionally, animals often perform gait transitions when undergoing large changes in velocity, lowering the cost of transport across different speeds [11]. The same principle applies to policies trained using AMP. In Fig. 2, we see that the robot transitions from a pacing motion to a canter motion when the desired velocity jumps from 1 m/s to 2 m/s. While pacing is the optimal gait at low speeds, entering a canter motion with a flight phase is a more energy-efficient option at high speeds.

### B. Task Completion in Real

Imitation learning approaches that constrain the policy to explicitly track a specified reference motion [38] would make it difficult for the policy to produce behaviors that deviate significantly from the reference data, even though deviating from the reference data may be required to complete the desired task. As shown in Fig. 6, a policy trained using the AMP is able to track the commanded linear and angular velocities, even though the 4.5 second German Shepherd dataset does not contain motions of the dog moving at these

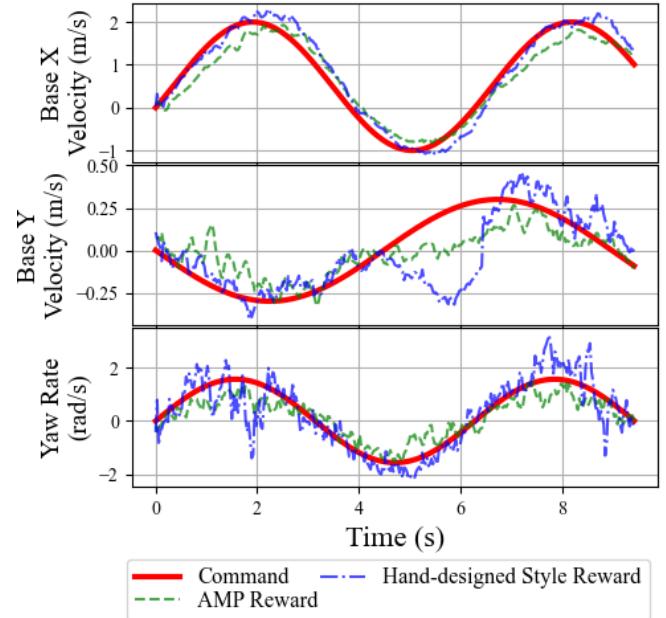


Fig. 6. Comparison of motion prior style reward, hand-designed style reward, and no style reward in ability to track a sinusoidal linear and angular velocity command. The policy trained with no style reward was evaluated in simulation due to the violent and jittery behaviors it exhibited (shown in Fig. 5).

TABLE II  
VELOCITY TRACKING AND MECHANICAL EFFICIENCY.

Commanded Forward Velocity (m/s)		0.4	0.8	1.2	1.6
Average Measured Velocity (m/s)	AMP Reward	<b>0.36 ±0.01</b>	<b>0.77 ±0.01</b>	<b>1.11 ±0.01</b>	<b>1.52 ±0.03</b>
	Complex Style Reward	0.41 ±0.01	0.88 ±0.02	1.28 ±0.03	1.67 ±0.03
	No Style Reward	0.42 ±0.01	0.82 ±0.01	1.22 ±0.01	1.61 ±0.01
Average Mechanical Cost of Transport	AMP Reward	<b>1.07 ±0.05</b>	<b>0.93 ±0.04</b>	<b>1.02 ±0.05</b>	<b>1.12 ±0.1</b>
	Complex Style Reward	1.54 ±0.17	1.37 ±0.12	1.40 ±0.10	1.41 ±0.09
	No Style Reward	14.03 ±0.99	8.00 ±0.44	6.05 ±0.28	5.18 ±0.20

particular velocities. This indicates that Adversarial Motion Priors enable the policy to learn behaviors that capture the essence of the reference motions while deviating from the dataset enough to complete the specified task. Take for example Fig. 4: navigating the quadruped through a route with sharp requires precision in tracking the velocity commands. A policy trained with Adversarial Motion Priors can learn to accurately track the velocity commands that guide the robot dog through the course while exhibiting naturalistic locomotion strategies.

### C. Qualitative Performance of Policies in Real

In addition to evaluating the performance of the policies in simulation, we also evaluate the effectiveness of the policies when deployed on a real robot. As mentioned in Section IV-A, a compelling benefit of training with Adversarial Motion Priors is that the policy can learn to extract the energy-efficient motion priors from the data. Part of this energy efficiency stems from the policy learning to change its gait depending on the velocity commands. Figure 2 demonstrates this phenomenon in practice. When the velocity command increases from 1 m/s to 2 m/s, the policy dramatically alters its gait from a pace to a canter. The pacing motion (Fig 3:A) is often used by animals at low speed and involves alternating swing and stance phases for the left and right feet. Meanwhile, the canter motion used by animals traveling at high speeds and is composed of an alternating placement of front feet and hind feet, followed by a flight phase (Fig. 2:A-2). Transitioning to a canter maneuver results in a dramatically different COT profile (Fig. 2:D). While the pace motion exhibits a fairly constant COT, the canter motion produces large spikes in COT corresponding to the lift-off phase and relatively low-valued troughs associated with the flight and touch-down phase. Also shown in Figure 3:B is the trotting motion that emerges from training with AMP rewards.

## V. CONCLUSIONS

We demonstrate that learning motion priors using adversarial imitation learning produces style rewards that encourage the policy to produce behaviors that are grounded in the reference motion dataset. Using this technique, we circumvent the need to define complex hand-designed style rewards while still enabling transfer to the real world. Additionally, we demonstrate that policies trained with Adversarial Motion Priors can deviate from the motions in the reference dataset as needed to achieve the specified task objectives. We also compare the energy efficiency of policies trained with hand-defined style reward, AMP style rewards, and no style reward, and demonstrate that AMP style rewards lead to energy-efficient locomotion strategies. We argue that this stems from the energy-efficient prior provided by motions in the dataset, as well as the policy’s ability to transition between the most optimal gaits corresponding to the commanded velocities.

## APPENDIX

### ACKNOWLEDGMENT

The authors would like to thank Adam Lau, Justin Kerr, Lars Berscheid, and Chung Min Kim for their helpful contributions and discussions.

### REFERENCES

- [1] A. W. Winkler, D. C. Bellicoso, M. Hutter, and J. Buchli, “Gait and trajectory optimization for legged systems through phase-based end-effector parameterization,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 3, pp. 1560–1567, July 2018.
- [2] M. Raibert, *Legged Robots that Balance*, ser. Artificial Intelligence. MIT Press, 1986.
- [3] M. Xie, A. Escontrela, and F. Dellaert, “A factor-graph approach for optimization problems with dynamics constraints,” *CoRR*, vol. abs/2011.06194, 2020.
- [4] K. Byl and R. Tedrake, “Dynamically diverse legged locomotion for rough terrain,” in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 1607–1608.
- [5] 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, 2022.
- [6] 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.
- [7] A. Kumar, Z. Fu, D. Pathak, and J. Malik, “RMA: rapid motor adaptation for legged robots,” *CoRR*, vol. abs/2107.04034, 2021.
- [8] 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)*. IEEE, 2021, pp. 7309–7315.
- [9] G. Ji, J. Mun, H. Kim, and J. Hwangbo, “Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion,” *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.
- [10] N. Heess, D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. A. Riedmiller, and D. Silver, “Emergence of locomotion behaviours in rich environments,” *CoRR*, vol. abs/1707.02286, 2017.
- [11] D. F. Hoyt and C. R. Taylor, “Gait and the energetics of locomotion in horses,” *Nature*, vol. 292, no. 5820, pp. 239–240, Jul 1981.
- [12] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, “Policies modulating trajectory generators,” 2019.
- [13] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, “Fast and efficient locomotion via learned gait transitions,” *CoRR*, vol. abs/2104.04644, 2021.
- [14] 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, p. eabk2822, 2022.
- [15] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. Panne, “Learning locomotion skills for cassie: Iterative design and sim-to-real,” in *Conference on Robot Learning*. PMLR, 2020, pp. 317–329.
- [16] W. Yu, G. Turk, and C. K. Liu, “Learning symmetric and low-energy locomotion,” *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–12, 2018.
- [17] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, “Amp: Ad-

TABLE III  
COMPLEX REWARD FORMULATION BASELINE.

Reward Term	Definition	Scale
z base linear velocity	$(v_t^z)^2$	-2
xy base angular velocity	$\ \vec{\omega}_t^{xy}\ $	-0.05
Non-flat base orientation	$\ \vec{g}_t^{xy}\ $	-0.01
Torque penalty	$\ \vec{\tau}\ $	-1e-5
DOF acceleration penalty	$\ \ddot{\theta}\ $	-2.5e-7
Penalize action changes	$\ \vec{a}_t - \vec{a}_{t-1}\ $	-0.01
Collision penalty	$ I_{c, \text{body}} \setminus I_{c, \text{foot}} $	-1
Termination penalty	$\mathbb{I}_{\text{terminate}}$	-0.5
DOF lower limits	$-\max(\vec{\theta} - \vec{\theta}_{\text{lim, low}}, 0)$	-10.0
DOF upper limits	$\min(\vec{\theta} - \vec{\theta}_{\text{lim, high}}, 0)$	-10.0
Torque limits	$\min( \vec{\tau} - \vec{\tau}_{\text{lims}} , 0)$	-0.0002
Tracking linear vel	$\exp(-\ \vec{v}_t^{xy} - \vec{v}_t^{xy}\ )$	1.0
Tracking angular vel	$\exp(- \omega_t^z - \omega_t^z )$	0.5
Reward long footsteps	$\sum_{\text{feet}} \mathbb{I}_{\text{swing}} t_{\text{swing}}$	1.0
Penalize large contact forces	$\ \min(\vec{f} - \vec{f}_{\text{max}}, 0)\ $	-1.0

- versarial motion priors for stylized physics-based character control,” *ACM Trans. Graph.*, vol. 40, no. 4, jul 2021.
- [18] X. Mao, Q. Li, H. Xie, R. K. Lau, Z. Wang, and S. Smolley, “Least squares generative adversarial networks,” in *2017 IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, oct 2017, pp. 2813–2821.
- [19] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, “Learning to walk in minutes using massively parallel deep reinforcement learning,” *CoRR*, vol. abs/2109.11978, 2021.
- [20] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, “Solving rubik’s cube with a robot hand,” *arXiv preprint arXiv:1910.07113*, 2019.
- [21] S. James and E. Johns, “3d simulation for robot arm control with deep q-learning,” *CoRR*, vol. abs/1609.03759, 2016.
- [22] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” *CoRR*, vol. abs/1603.02199, 2016.
- [23] S. Gu, E. Holly, T. P. Lillicrap, and S. Levine, “Deep reinforcement learning for robotic manipulation,” *CoRR*, vol. abs/1610.00633, 2016.
- [24] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, “Target-driven visual navigation in indoor scenes using deep reinforcement learning,” in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3357–3364.
- [25] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak, “Coupling vision and proprioception for navigation of legged robots,” 2021.
- [26] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, “Reward learning from human preferences and demonstrations in atari,” *Advances in neural information processing systems*, vol. 31, 2018.
- [27] J. Lee, J. Hwangbo, and M. Hutter, “Robust recovery controller for a quadrupedal robot using deep reinforcement learning,” *arXiv preprint arXiv:1901.07517*, 2019.
- [28] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohenz, and V. Vanhoucke, “Sim-to-real: Learning agile locomotion for quadruped robots,” *arXiv preprint arXiv:1804.10332*, 2018.
- [29] A. Iscen, G. Yu, A. Escontrela, D. Jain, J. Tan, and K. Caluwaerts, “Learning agile locomotion skills with a mentor,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2019–2025.
- [30] N. Pollard, J. K. Hodgins, M. Riley, and C. Atkeson, “Adapting human motion for the control of a humanoid robot,” in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA ’02)*, May 2002.
- [31] D. B. Grimes, R. Chalodhorn, and R. P. N. Rao, “Dynamic imitation in a humanoid robot through nonparametric probabilistic inference,” in *Robotics: Science and Systems*, G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, Eds. The MIT Press, 2006.
- [32] W. Suleiman, E. Yoshida, F. Kanehiro, J. Laumond, and A. Monin, “On human motion imitation by humanoid robot,” in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 2697–2704.
- [33] K. Yamane, S. O. Anderson, and J. K. Hodgins, “Controlling humanoid robots with human motion data: Experimental validation,” in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, Dec 2010, pp. 504–510.
- [34] C. Atkeson and S. Schaal, “Learning tasks from a single demonstration,” in *Proceedings of International Conference on Robotics and Automation*, vol. 2, 1997, pp. 1706–1712 vol.2.
- [35] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi, “Generating whole body motions for a biped humanoid robot from captured human dances,” in *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, vol. 3, Sep. 2003, pp. 3905–3910 vol.3.
- [36] S. Kim, C. Kim, B. You, and S. Oh, “Stable whole-body motion generation for humanoid robots to imitate human motions,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2009, pp. 2518–2524.
- [37] J. Koenemann, F. Burget, and M. Bennewitz, “Real-time imitation of human whole-body motions by humanoids,” *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2806–2812, 2014.
- [38] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, “Deepmimic: Example-guided deep reinforcement learning of physics-based character skills,” *ACM Trans. Graph.*, vol. 37, no. 4, pp. 143:1–143:14, Jul. 2018.
- [39] L. Liu and J. Hodgins, “Learning basketball dribbling skills using trajectory optimization and deep reinforcement learning,” *ACM Trans. Graph.*, vol. 37, no. 4, Jul. 2018.
- [40] X. B. Peng, A. Kanazawa, J. Malik, P. Abbeel, and S. Levine, “Sfv: Reinforcement learning of physical skills from videos,” *ACM Trans. Graph.*, vol. 37, no. 6, Nov. 2018.
- [41] S. Lee, M. Park, K. Lee, and J. Lee, “Scalable muscle-actuated human simulation and control,” *ACM Trans. Graph.*, vol. 38, no. 4, Jul. 2019.
- [42] N. Chentanez, M. Müller, M. Macklin, V. Makoviychuk, and S. Jeschke, “Physics-based motion capture imitation with deep reinforcement learning,” in *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, ser. MIG ’18. New York, NY, USA: Association for Computing Machinery, 2018.
- [43] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes, “Drecon: Data-driven responsive control of physics-based characters,” *ACM Trans. Graph.*, vol. 38, no. 6, Nov. 2019.
- [44] S. Park, H. Ryu, S. Lee, S. Lee, and J. Lee, “Learning predict-and-simulate policies from unorganized human motion data,” *ACM Trans. Graph.*, vol. 38, no. 6, Nov. 2019.
- [45] 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,” *CoRR*, vol. abs/2103.14295, 2021.
- [46] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML ’04. New York, NY, USA: Association for Computing Machinery, 2004, p. 1.
- [47] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, “Maximum mutual entropy inverse reinforcement learning,” in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI’08. AAAI Press, 2008, p. 1433–1438.
- [48] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4565–4573.
- [49] S. Nowozin, B. Cseke, and R. Tomioka, “f-gan: Training generative neural samplers using variational divergence minimization,” in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016, pp. 271–279.
- [50] L. Ke, M. Barnes, W. Sun, G. Lee, S. Choudhury, and S. S. Srinivasan, “Imitation learning as f-divergence minimization,” *CoRR*, vol. abs/1905.12888, 2019.
- [51] Y. Li, J. Song, and S. Ermon, “Infogail: Interpretable imitation learning from visual demonstrations,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
- [52] J. Merel, Y. Tassa, D. TB, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess, “Learning human behaviors from motion capture by adversarial imitation,” *CoRR*, vol. abs/1707.02201, 2017.
- [53] Z. Wang, J. S. Merel, S. E. Reed, N. de Freitas, G. Wayne, and N. Heess, “Robust imitation of diverse behaviors,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017, pp. 5320–5329.
- [54] L. M. Mescheder, “On the convergence properties of GAN training,” *CoRR*, vol. abs/1801.04406, 2018.
- [55] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, “Deep imitation learning for complex manipulation tasks from virtual reality teleoperation,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5628–5635.
- [56] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, “Domain randomization for transferring deep neural networks from simulation to the real world,” *CoRR*, vol. abs/1703.06907, 2017.
- [57] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *CoRR*, vol. abs/1707.06347, 2017.
- [58] J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox, “Gpu-accelerated robotic simulation for distributed reinforcement learning,” *CoRR*, vol. abs/1810.05762, 2018.