LAGOON: Language-Guided Motion Control

Shusheng Xu[1]*, **Huaijie Wang**[1]*, **Jiaxuan Gao**[1], **Yutao Ouyang**[2], **Chao Yu**[1], **and Yi Wu**[1,3]
[1]Tsinghua University, Beijing, China [2]Xiamen University, Xiamen, China
[3]Shanghai Qi Zhi Institute, Shanghai, China
{xuss20, wanghuai19, gaojx19, yc19}@mails.tsinghua.edu.cn
OYYTTYYO@outlook.com jxwuyi@gmail.com

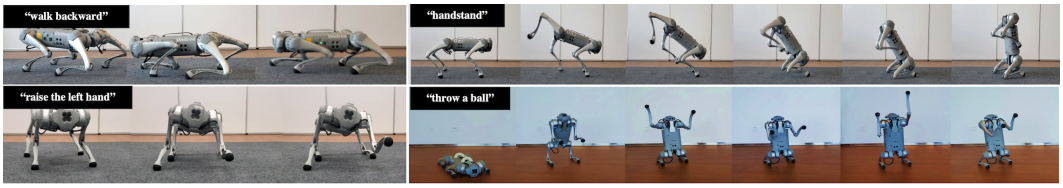

Figure 1: We developed a system called LAGOON. Given a *high-level* language command, LAGOON can autonomously train a control policy according to the provided command.

Abstract:
We aim to control a robot to physically behave in the real world following any high-level language command like "cartwheel" or "kick". Although human motion datasets exist, this task remains particularly challenging since generative models can produce physically unrealistic motions, which will be more severe for robots due to different body structures and physical properties. Deploying such a motion to a physical robot can cause even greater difficulties due to the sim2real gap. We develop LAnguage-Guided mOtion cONtrol (LAGOON), a multi-phase reinforcement learning (RL) method to generate physically realistic robot motions under language commands. LAGOON first leverages a pretrained model to generate a human motion from a language command. Then an RL phase trains a control policy in simulation to mimic the generated human motion. Finally, with domain randomization, our learned policy can be deployed to a quadrupedal robot, leading to a quadrupedal robot that can take diverse behaviors in the real world under natural language commands.

1 Introduction

Reinforcement learning (RL) has been a trending paradigm for addressing intricate challenges in robotic control, encompassing domains such as bipedal [1] and quadrupedal locomotion [2], drone racing [3], and robotic arm manipulation [4]. Specifically, an RL-based approach trains a neural policy within a simulated environment through the formulation of task-specific reward functions [5], followed by the transfer of the policy to a physical robot via domain randomization techniques [6].

Despite the successes, most existing RL methods focus on low-level robust control tasks, such as walking [7], and rely on a heavily engineered task-specific reward [2, 7]. Whenever the task goal changes, the creation of a new reward function requires substantial efforts. It remains an open challenge whether we can directly train control policies to generate complex behaviors according to *high-level semantic* commands, such as "throw a ball" or "handstand", without the need to specify any sophisticated reward function.

The prospect of controlling robots through high-level language commands becomes increasingly promising with the advancement of pre-trained models [8]. Recently, with diffusion models, it has

*Equal contribution.

become possible to generate diverse human trajectories based on high-level language commands [9], which suggests a feasible direction for language-guided control: initiating motion generation using a pre-trained model followed by the implementation of imitation learning. However, a common pitfall of the existing motion generation methods is that the generated motion may often violate real-world physical constraints since no physics simulation is performed during such an end-to-end generation process. When generating robot motions, this issue is more severe, since most existing motion datasets are collected from human demonstrators while a robot can have a drastically different body structure from humans. Further applying such a motion to a physical robot can introduce even greater challenges due to the sim2real gap.

We propose a novel RL-based method, LAnguage-Guided mOtion cONtrol (LAGOON), to address all the aforementioned challenges. LAGOON is a multi-stage approach benefiting from both motion generation and RL training. First, LAGOON adopts a motion diffusion model to generate a human motion from a language command. Then, we convert the generated human motion into a semantically desired yet physically unrealistic robot motion. Next, an RL phase is performed to learn a policy in a physics engine to control a simulated robot to mimic the target motion. Finally, with domain randomization, the learned RL policy can be deployed on a real-world robot.

We emphasize that effectively training an RL policy to mimic a target motion is non-trivial in our setting. Existing algorithms that can learn motion control from demonstration videos often assume *perfect* demonstrations produced by human professionals [10, 11, 12, 13]. In contrast, our target motion is completely synthetic and can be highly unrealistic, including physically impossible poses or even missing frames that lead to teleportation or floating behaviors. We adopt a special rewarding mechanism for motion imitation, which combines both adversarially learned critic reward for high-level semantic consistency and an optimal-matching-based state-error reward to enforce fine-grained consistency to key frames from the target motion.

We evaluate LAGOON on two types of robots in simulation: a humanoid robot, which resembles the human body structure for easier motion imitation, and a quadrupedal robot, posing more challenges for RL training due to its distinct body structure. Our empirical findings demonstrate that LAGOON is able to generate robust control policies for both robots, producing physically realistic behaviors following various language commands. For the humanoid robot, LAGOON produces better policies on high-level commands like "cartwheel" as well as commands requiring fine-grained control like "kick", outperforming all the RL baselines consistently. For the quadrupedal robot, despite the substantial differences in body structure, LAGOON is able to produce a policy to execute challenging commands like "*throw a ball*" by controlling the robot to stand up and wave its front legs.

We also successfully deployed our system to a physical quadrupedal robot in the real world, resulting in a quadrupedal robot automatically performing diverse motions that are semantically consistent with a variety of language commands, such as "walk backward", "handstand", "raise the left hand" and "throw a ball", as shown in Fig. 1.

2 Related work

2.1 Language-Conditioned Motion Generation

Early attempts at translating text descriptions into human motion employed deterministic encoder-decoder architectures [14, 15]. Recent efforts have shifted towards deep generative models such as GANs, VAEs [16, 17], or diffusion models [18, 9, 19] due to the stochastic nature of motions. Note that these motion generation methods are trained on large datasets typically limited to human motions. Despite their superior performances, standard deep generative models do not explicitly incorporate the law of physics into the generation process. [20] integrate the imitation policy trained in a physics simulator into the sampling process of the diffusion model. However, this approach still relies on a manually designed residual force [21] at the root joint to compensate for the dynamics mismatches between the physics model and real humans, which is not suitable for real robot control. In comparison, we concentrate on physically realistic robots with various structures.

2.2 Learning Methods for Robot Control

To attain natural behaviors, researchers carefully designed heuristics for symmetry [22], energy consumption [23], and proper contact with the environment [24]. Nonetheless, these methods typically demand substantial domain expertise and are thus limited to simpler tasks. In contrast, imitation learning (IL) is more general. It can learn from expert demonstrations and deploy the learned policies in physics simulators [25, 26, 27] or the real world [28, 29]. IL assumes *perfect* real-world data, while we only have access to *imperfect* synthetic demonstrations. Some approaches use pre-trained models for robot control, such as using a language model for representation learning [30] or semantic planning [31]. However, these methods demand human demonstrations for low-level control, while we do not require additional control data. Others [32, 33] train a video diffusion model to generate a sequence of trajectory states and use inverse dynamics to infer actions. In our setting, the reference motion and the actual policy trajectory cannot be precisely aligned, making it infeasible to produce actions through inverse dynamics.

2.3 State-Based Imitation Learning

In cases where expert actions are unavailable, policies have to learn from states. One approach is to train a dynamics model to predict actions from state transitions and then apply behavior cloning [34, 35]. Other methods perform RL directly with a state-based imitation reward, such as differences in state representations [26, 25, 11, 12] or an adversarially learned discriminator [36, 37, 38, 27, 39, 40, 41]. We leverage both of these reward types. Since the policy motion and the reference motion can be largely mismatched, both reward terms are critical for empirical success.

2.4 Motion and Control of Quadrupedal Robots

Previous approaches have focused on generating controllable or natural quadrupedal motions and gaits either by imitating animals [28] or by relying on heavily engineered reward functions [2, 7]. [42] build a human-to-quadrupedal control interface by collecting matching pairs of human and robot motions. Our work demonstrates the ability to generate diverse motions without the need for domain-specific data or meticulously designed reward functions.

3 Preliminary

3.1 Human Motion Generation

Human motion generation aims to produce a sequence $x_{0:H} = \{x_h\}_{h=0}^{H}$. $x_h \in \mathbb{R}^{J\times K}$ represents a human pose using K-dimensional features of J joints. Here K-dimensional features can be either the joint angles or positions. A language-conditioned motion generation model aims to generate a motion matching the language command c.

Diffusion models [43, 44, 45] are able to generate high-quality human motions [18, 9, 19]. These works model the data distribution by injecting noises into it and gradually denoise a sample from a Gaussian distribution. The forward diffusion process injects i.i.d. Gaussian noises, namely

$$x^l_{0:H} \sim \mathcal{N}(\sqrt{\alpha_l}x^{l-1}_{0:H}, (1-\alpha_l)\mathbf{I}),$$

where $x^0_{0:H}$ denotes samples drawn from the real data distribution $p^0(x^0)$. For large enough l, $x^l_{0:H}$ approximately follows the Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$. A denoiser $F_d(x^l, l)$ is trained to gradually denoise $x^l_{0:H}$ back to $x^{l-1}_{0:H}$. The training objective of F_d is usually given by

$$\mathbb{E}_{x^0\sim p^0(x^0), l\sim q(l), \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[\lambda(l)\|x^0 - F_d(x^l, l)\|^2\right],$$

where $q(l)$ is a distribution from which l is samapled and $\lambda(l)$ is a weighting factor.

3.2 RL for Control

Rather than generating a motion directly, RL methods learn a policy in simulation to control a robot to perform the desired motion according to some given reward function.

3.2.1 Markov Decision Process

The robot control problem can be formulated as a Markov Decision Process (MDP) denoted by $M = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma \rangle$. Here $\mathcal{S}$ is the state space, and $\mathcal{A}$ is the action space. $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function. $\mathcal{T}(s'|s, a)$ denotes the probability of reaching state s' from state s under action a. $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function and γ is the discounted factor. In our task setup, the reward function r is generated based on a language command c. In practice, when applying RL for robot control, complex reward designs are often required due to a lot of motor joints and movement constraints [7]. At each time step t, the control policy produces an action $a_t \sim \pi(\cdot|s_t)$, and receives a reward $r(s_t, a_t)$. The objective of RL is to find the optimal policy $\pi^\star$ that could maximize the discounted accumulated reward, where s_0 is the initial state:

$$\pi^\star = \arg\max_\pi \mathbb{E}\left[\sum_{t\geq 0} \gamma^t r(s_t, a_t) \mid a_t \sim \pi\left(\cdot \mid s_t\right), s_0\right]$$

3.2.2 RL-Based Motion Imitation

It is difficult to design reward functions directly for robot control tasks. One approach to solve this problem is imitation learning (IL). The goal of IL is to find the optimal policy π^* that covers the distribution of state-action pairs in the dataset $\mathcal{D} = \{(s_i, a_i)\}_{i=1}^m$. We are interested in cases where expert actions are not available and the dataset only contains a trajectory of states, i.e. $\mathcal{D} = \{x_i\}_{i=0}^H$. One approach to solve this problem is state-based IL [36, 25, 27]. State-based IL typically designs a state-error reward [25], which encourages the imitator to reach the reference states $x_{0:H}$. Let s_t denote the imitator state at timestep t, the state-error reward r_t^{err} is defined as $r_t^{\text{err}} = Sim(x_t, s_t)$, which represents the similarity between the reference state x_t and the imitator state s_t at timestep t. A crucial assumption of r_t^{err} is a strict timing alignment between the demonstration and the roll-out trajectory. This can be problematic when the reference motion $x_{0:H}$ is not physically realistic. Adversarial imitation learning (AIL) tackles this issue by training a discriminator D_ψ to differentiate behaviors generated by the imitator from the reference motion $x_{0:H}$, where ψ is the network parameters. D_ψ then scores the states s_t generated by the imitator. Specifically, D_ψ is trained to discriminate between state transitions in the reference motion and the samples generated by π_θ:

$$\mathcal{L}_{\text{disc}} = \mathbb{E}_h[-\log D_\psi(x_h, x_{h+1})] + \mathbb{E}_t[-(1 - \log D_\psi(s_t, s_{t+1}))] + w_{\text{gp}}\mathcal{L}_{\text{gp}},$$

where $\mathcal{L}_{\text{gp}}$ is a gradient penalty term. As discussed in [46], this zero-centered gradient penalty stabilizes training and helps convergence. AIL optimizes the policy π_θ to maximize the discounted accumulated adversarial reward. The adversarial reward is given by,

$$r_t^{\text{adv}} = -\log(1 - D_\psi(s_{t-1}, s_t)). \quad (1)$$

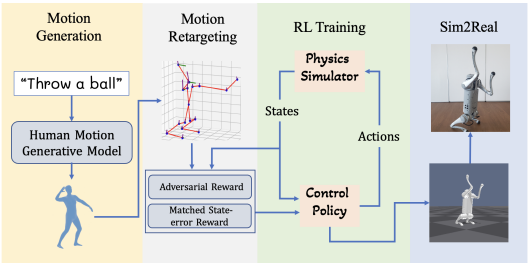

Figure 2: Overview of the multi-phase method LAnguage-Guided mOtion cONtrol (LAGOON). We first generate a human motion using the motion generation model. Then the human motion can be retargeted to a robot skeleton that differs largely from humans. By introducing RL training, we train a robust control policy in the physics simulator. Finally, we deploy the control policy to the real-world robot.

4 Methodology

As shown in Fig. 2, we derive a robot control policy following a language command c through a multi-phase method. We first generate a motion sequence $x_{0:H}$ conditioned on c using a human motion generation model. $x_{0:H}$ is then retargeted to the robot skeleton to produce a robot motion $y_{0:H}$. Finally, we adopt RL training to obtain a control policy π_θ and transfer the learned policy to the real world via domain randomization.

4.1 Motion Generation and Retargeting

In the motion generation stage, we adopt a SOTA Human Motion Diffusion Model (MDM) [9] to generate human motion $x_{0:H}$ conditioned on a language command c. Since MDM can only generate human motion, we then adopt a retargeting stage to map the human motion $x_{0:H}$ to the desired robot motion $y_{0:H}$. Taking the quadrupedal robot as an example, we map the human skeleton to the robot's skeleton, with the human arms corresponding to the robot's two front legs and the human legs corresponding to the robot's two rear legs, and then retarget each joint and joint rotation. For the humanoid robot, the number of joints on different skeletons may vary and therefore also need to be retargeted accordingly. For those joints that are redundant, they are simply discarded[2].

4.2 RL Training

The RL phase trains a control policy π_θ to imitate the retargeted robot motion $y_{0:H}$. The policy takes in robot states and outputs an action to interact with the physics simulator. The reward is calculated by comparing the robot states with the retargeted reference robot motion. The motion generated by MDM is *physics-ignoring*, so floating, penetration and teleportation behaviors may often occur, which can be amplified in the retargeted robot motion $y_{0:H}$. Inaccurate reference motions pose significant challenges to motion imitation, which we tackle via a careful reward design.

4.2.1 Reward Design

We combine both the adversarial reward r^{adv} (Eq. (1)) and a variant of the state-error reward r^{me} (Eq. (2)). Intuitively, the adversarial reward is universal to capture high-level semantic consistency with the reference motion. However, we empirically observe that only using an adversarial reward can fail to match critical poses in the reference motion. For example, when given the command "kick", the policy trained with the adversarial reward alone only learns to stand but fails to perform a kick. Hence, for more fine-grained body control, we additionally leverage a state-error reward, which can be nontrivial since the policy trajectory and the reference motion are not well aligned.

To tackle the temporal mismatch issue, we employ a matching algorithm between the policy rollout trajectories and the reference motion to find the best temporal alignment leading to the highest state-error reward. More specifically, let $y_{0:H}$ be the reference motion sequence and $\tau = (s_0, s_1, \dots, s_T)$ be a trajectory from the policy. We define a matching M between $y_{0:H}$ and $\tau = (s_0, s_1, \dots, s_T)$ by

$$M = \{(u_{0:k}, v_{0:k}) \mid k \in \mathbb{N}_+, 0 \le u_0 < \cdots < u_k \le H, 0 \le v_0 < \cdots < v_k \le T\}.$$

where s_{v_m} is matched with y_{u_m} for all $0 \le m \le k$. Recall that $Sim(y_i, s_j)$ is a similarity measurement between the i-th motion state y_i and the robot state s_j at timestep j. We aim to find the optimal matching $M^* = (u^*_{0:k^*}, v^*_{0:k^*})$ that maximizes the total similarity, namely

$$(u^*_{0:k^*}, v^*_{0:k^*}) = \underset{(u_{0:k}, v_{0:k}) \in M}{\arg\max} \sum_{m=0}^{k} Sim(y_{u_m}, s_{v_m}),$$

which can be solved via dynamic programming. The optimal matching M^* helps filter out unrealistic poses and allows the robot to smoothly transit between two consecutive motion frames. With the optimal matching $(u^*_{0:k^*}, v^*_{0:k^*})$, the matched state-error reward is defined as

$$r^{\text{me}}_t = Sim(y_{u^*_m}, s_{v^*_m}) \cdot \mathbb{I}(t = v^*_m \in v^*_{0:k^*}) \tag{2}$$

Our final reward is a combination of adversarial reward and matched state-error reward.

$$r_t = \lambda_{\text{adv}} r^{\text{adv}}_t + \lambda_{\text{me}} r^{\text{me}}_t \tag{3}$$

where λ_{adv} and λ_{me} are weighting factors. We also remark that the adversarial reward remains critical since it provides much denser reward signals than the state-error reward.

[2] https://github.com/NVIDIA-Omniverse/IsaacGymEnvs/tree/main/isaacgymenvs/tasks/amp/poselib

4.2.2 PPO with Augmented Critic Inputs

We utilize PPO [47] for RL training, which adopts an actor-critic structure with two separate neural networks, i.e. a policy π_θ and a value function V_ϕ. The critic is only used for variance reduction at training time, so we can input additional information not presented in robot states to the value network to accelerate training. In particular, given the trajectory τ, the reference sequence y, and the optimal matching M^*, for each state s_t, we take the next future reference motion for s_t from M^* as the additional information to the critic. Such future information significantly improves training in practice. We also remark that similar techniques have been widely adopted in multi-agent RL [48].

4.2.3 Domain Randomization

In order to learn robust control policies, we adopt domain randomization [6] during RL training. We randomize both the terrains and physics parameters in the simulator so that the trained RL policy can generalize to different terrain conditions and even to the real world.

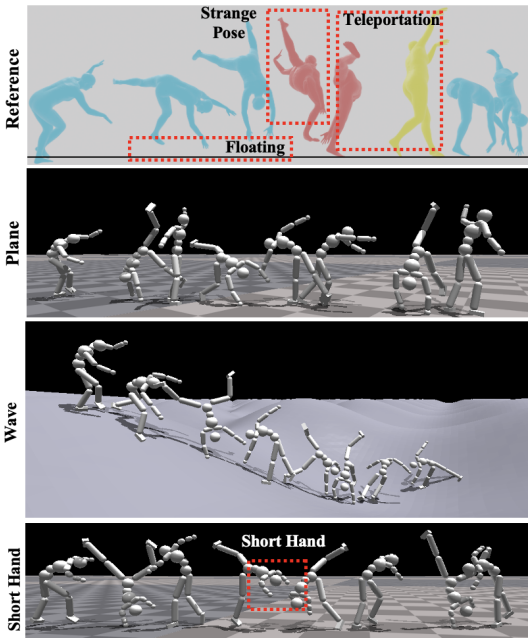

Figure 3: The reference motion sequence overlooks the law of physics. The trained policies robustly perform the "cartwheel" motion even in complex terrains (Wave) or a different skeleton with shorter arms (Short Hand).

5 Experiment

We conduct experiments on the humanoid and quadrupedal robots in the IsaacGym [49] simulator. We test LAGOON using the 28 DoF humanoid from AMP [27, 49] and the go1 quadrupedal robot[3] with 12 DoF. We use the target DoF angles of proportional derivative (PD) controllers as the actions. The action dimensions are 28 and 12 for the humanoid and quadrupedal robots, respectively. We also deploy the policies in the real-world quadrupedal robot.

5.1 Humanoid Robot

For the humanoid robot, We conduct experiments on the tasks specified by the texts "the person runs backward", "cartwheel", and "the person kicks with his left leg". These tasks include the movement and rotation of the entire body and fine-grained control of parts of the body.

Baselines: We compare LAGOON with other state-based imitation learning methods. BCO [34] is a non-RL method that learns an inverse dynamics model to label the reference motion with actions and adopt behavior cloning on the labeled motion. Other baselines are RL-based methods. State Err. only uses the state-error reward. GAILfO [37, 38, 27] and RGAILfO [41] utilize the adversarial rewards. In particular, RGAILfO tried to alleviate the problem of dynamics mismatch by introducing an adversary policy when collecting trajectories. We also conduct experiments using the hand-designed reward and denote it as pure RL. The reward for "run backward" is the backward velocity at each timestep. For the task "kick", let h_t denote the height of the robot's left foot at timestep t, and the reward is computed as $\max(0, h_t - \max_{s<t}(h_s))$, which measure the difference between the current foot height and the previous maximum height. The "cartwheel" task is excluded since it is difficult to manually design reward functions for doing cartwheels.

Training Details: The input states of the control policy include the root's linear velocity and angular velocity, the local velocity and rotation of each joint, and the 3D position of the end-effectors. We

[3] https://www.unitree.com/go1

train the policies by randomizing the terrains and the physics parameters to handle various complex situations. There are four terrains during training. "Plane" refers to a flat surface without variations in elevation. "Rand." is terrain with a bit of random undulation. "Pyramid" is square cone terrain with steps. "Wave" is the terrain with a great deal of undulation. We additionally train a policy for a humanoid robot with shorter arms.

We create 4096 parallel simulation environments in IsaacGym to collect training samples. The max episode length of each simulation is 300. Each environment would be reset when the robot in it falls (i.e., any part of the humanoid except hands and feet is in contact with the ground). We train the policy for 5,000 iterations and adopt the final policy for evaluation. All the RL-based baselines are trained using the same hyper-parameters.

5.1.1 Illustration of Learned Motion

A critical problem of the generated motion is physics ignoring. As shown in Fig. 3. The top picture is the motion generated by MDM conditioned on the language prompt "cartwheel", where some postures are floating or ground-penetrating. We mark the strange postures impossible to imitate in red, and the posture represents that teleportation occurs in yellow. After retargeting and RL training, the control policy can do cartwheels in various terrains. For example, the control policy can do cartwheels on a large slope. We also train the policy on the robot with short hands. We can observe that the robot can also do cartwheels steadily. Since these behaviors are conducted in the physics simulator, the generated robot motion would always be physically realistic.

Table 1: Success rates of different tasks. We train a single policy over all 4 terrains and evaluate the policy separately on each terrain.

Terrian	Plane	Rand.	Pyramid	Wave
Task: Cartwheel				
LAGOON	$\mathbf{100.0 \pm 0.0}$	$\mathbf{98.8 \pm 0.5}$	$\mathbf{85.2 \pm 6.6}$	$\mathbf{86.4 \pm 10.6}$
GAILfO	66.7 ± 47.1	65.6 ± 46.3	33.1 ± 38.7	58.7 ± 34.7
RGAILfO	0.0 ± 0.0	6.3 ± 8.6	16.3 ± 22.7	19.9 ± 13.2
State Err.	0.0 ± 0.0	4.7 ± 6.6	15.0 ± 21.1	11.9 ± 15.3
Pure RL	-	-	-	-
BCO	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Task: Kick				
LAGOON	$\mathbf{100.0 \pm 0.0}$	$\mathbf{78.9 \pm 6.8}$	$\mathbf{89.9 \pm 7.0}$	69.7 ± 7.5
GAILfO	0.0 ± 0.0	46.6 ± 17.5	60.4 ± 15.1	45.5 ± 8.3
RGAILfO	33.3 ± 47.1	28.9 ± 18.9	39.7 ± 14.2	29.5 ± 14.0
State Err.	$\mathbf{100.0 \pm 0.0}$	75.3 ± 24.9	85.4 ± 20.7	$\mathbf{70.0 \pm 29.2}$
Pure RL	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0	100.0 ± 0.0
BCO	0.0 ± 0.0	0.0 ± 0	0.0 ± 0.0	0.0 ± 0.0
Task: Run Backwards				
LAGOON	$\mathbf{100.0 \pm 0.0}$	99.8 ± 0.2	$\mathbf{100.0 \pm 0.0}$	95.5 ± 1.2
GAILfO	$\mathbf{100.0 \pm 0}$	99.6 ± 0.4	99.9 ± 0.1	96.2 ± 1.7
RGAILfO	$\mathbf{100.0 \pm 0.0}$	$\mathbf{99.9 \pm 0.0}$	99.8 ± 0.3	$\mathbf{98.8 \pm 0.5}$
State Err.	$\mathbf{100.0 \pm 0.0}$	99.1 ± 0.4	$\mathbf{100.0 \pm 0.0}$	97.5 ± 1.0
Pure RL	97.2 ± 0.7	25.5 ± 2.1	96.6 ± 0.2	24.0 ± 0.2
BCO	14.0 ± 0.5	13.6 ± 0.4	12.8 ± 0.9	13.5 ± 0.9

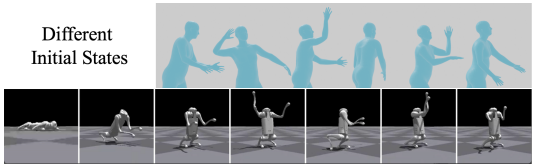

Figure 4: The task of "throw a ball" on the quadrupedal robot. Even though the getting up does not appear in the reference motion, the quadrupedal robot learns how to get up from the ground and then wave its front legs.

5.1.2 Comparision with Baselines

We compare LAGOON with various RL-based baselines. The results are listed in Tab. 1. We evaluate the policies on each task using success rates. The BCO baseline performs worst on all tasks and all terrains, as it is challenging to estimate the environment dynamics. Pure RL policies achieve high success rates. However, we observe the policy can't maintain balance. And it is usually difficult to design the reward for the pure RL method. LAGOON consistently outperforms the baselines on different terrains. For the task "cartwheel", the low success rates of State Err. indicate that tracking the states in the reference motion alone may not suffice for completing complex skills. For the task "kick", methods without state-error rewards (GAILfO, RGAILfO) have significantly lower success rates than methods with state-error rewards (LAGOON, State Err.). This result suggests that the state error reward encourages the policy to imitate the fine-grained poses.

5.2 Quadrupedal Robot

We evaluate LAGOON on the task given on the language text "the person throws a ball". The quadrupedal robot has an essentially different body structure from the humans, and the initial states

of the robots and the reference motion are largely different. The robot must first learn to "stand" like the humans without extra data. The result in Fig. 4 demonstrates that LAGOON makes the policy "stand" on two feet, then the robot successfully takes the behavior "throws a ball".

5.3 Real-World Robot Deployment

For the real-world experiments, besides the reward mentioned in Eq. 3, we add the following auxiliary rewards to protect the robots:

$$r_t^{\text{aux}} = [r_t^{\text{pl}}, r_t^{\text{a}}, r_t^{\text{tor}}, r_t^{\text{ar}}, r_t^{\text{col}}, r_t^{\text{slip}}] \quad (4)$$

Table 2: Randomization parameters.

Parameter	Operation	Distribution	Unit
Obs.Gravity	Additive	$\mathrm{U}(-0.05, 0.05)$	$\mathrm{m/s^2}$
Obs.ROT	Additive	$\mathrm{U}(-0.01, 0.01)$	rad
Obs.VEL	Additive	$\mathrm{U}(-1.5, 1.5)$	$\mathrm{rad/s}$
Trunk Mass	Additive	$\mathrm{U}(-1, 1)$	kg
Body Friction	Scaling	$\mathrm{U}(0.3, 3)$	1
Proportional Gain	Scaling	$\mathrm{U}(0.7, 1.3)$	1
Derivative Gain	Scaling	$\mathrm{U}(0.7, 1.3)$	1

where r_t^{pl} is a strict penalty to prevent the motor position from exceeding its limit. r_t^{a} and r_t^{tor} are used to mitigate motor behavior by penalizing excessive acceleration and torque. r_t^{ar} is introduced with the purpose of smoothing the action. To enhance motion regulation and safety, we incorporate r_t^{col} to penalize collisions. Additionally, we introduce r_t^{slip} to penalize the velocity component of the robot's toe perpendicular to the ground normal when it makes contact with the ground. This measure significantly alleviates slip-related issues. Furthermore, we incorporate a contact reset mechanism when the robot falls. We also adopt domain randomization, the details are listed in Table 2.

We demonstrate the real-world results of four different motions in Fig.(1, 5), including walking backward, raising the left hand, handstand, and throwing a ball.

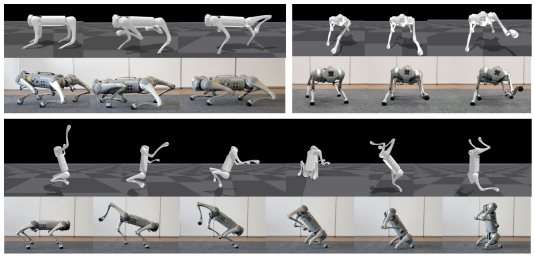

Figure 5: The retargeted reference motions and the behaviors of the real-world robot. The reference motions are imperfect and physics-ignorant.

We have identified multiple options for retargeting strategies, and LAGOON works well for different strategies and generates diverse control policies. For instance, in the case of the commands "handstand" and "throw a ball," we retarget all the robot joints, aligning the human hands with the robot's front legs and the human legs with the robot's rear legs Consequently, the robot executes these actions like a human. For the "walk backward" command, we realign the human legs with the robot's rear legs and mirror the states to the front legs, allowing the robot to mimic a dog's walking pattern. Furthermore, we also explored retargeting of partial joints. When given the command "raise the left hand," we solely retarget the joints of the left front hand. As a result, the robot stands on three legs while extending its left hand.

We present a comparison between the retargeted reference motions and the actual behaviors of the real-world robot, as illustrated in Fig. 5. Even though the reference motions appear to defy the laws of physics, our LAGOON policies excel at completing these tasks. For example, consider the command "walk backward." In the reference motion, the posture depicted cannot maintain stability, but our robot executes a secure backward walk. Similarly, when instructed to perform a "handstand," the reference motion begins from a standing posture and concludes in a precarious state. In contrast, our policy skips the two-legged stance, ultimately achieving a safe and steady handstand.

6 Conclusion

We propose a multi-phase method LAGOON to train robot control policy following the given language command. We first generate human motion using a language-conditioned motion diffusion model, and retarget the generated human motion to the robot skeleton. We adopt RL to train control policies. LAGOON finally produces a robust policy that controls a real-world quadrupedal robot to take behaviors consistent with the language commands.

References

[1] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath. Robust and versatile bipedal jumping control through multi-task reinforcement learning. *arXiv preprint arXiv:2302.09450*, 2023.

[2] G. B. Margolis and P. Agrawal. Walk these ways: Tuning robot control for generalization with multiplicity of behavior. *Conference on Robot Learning*, 2022.

[3] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.

[4] Y. Li, T. Kong, L. Li, Y. Li, and Y. Wu. Learning to design and construct bridge without blueprint. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2398–2405. IEEE, 2021.

[5] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5):698–721, 2021.

[6] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[7] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning*, pages 91–100. PMLR, 2022.

[8] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, M. Yan, and A. Zeng. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

[9] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-Or, and A. H. Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.

[10] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn. Reinforcement learning with videos: Combining offline observations with interaction. In J. Kober, F. Ramos, and C. Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 339–354. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/schmeckpeper21a.html.

[11] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018. doi:10.1109/icra.2018.8462891. URL http://dx.doi.org/10.1109/ICRA.2018.8462891.

[12] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi. XIRL: Cross-embodiment inverse reinforcement learning. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=RO4DM85Z4P7.

[13] Y. Seo, K. Lee, S. L. James, and P. Abbeel. Reinforcement learning with action-free pre-training from videos. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 19561–19579. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/seo22a.html.

[14] C. Ahuja and L.-P. Morency. Language2pose: Natural language grounded pose forecasting. In *2019 International Conference on 3D Vision (3DV)*, pages 719–728. IEEE, 2019.

[15] A. Ghosh, N. Cheema, C. Oguz, C. Theobalt, and P. Slusallek. Synthesis of compositional animations from textual descriptions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1396–1406, 2021.

[16] C. Guo, S. Zou, X. Zuo, S. Wang, W. Ji, X. Li, and L. Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5152–5161, 2022.

[17] M. Petrovich, M. J. Black, and G. Varol. Temos: Generating diverse human motions from textual descriptions. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 480–497. Springer, 2022.

[18] Z. Ren, Z. Pan, X. Zhou, and L. Kang. Diffusion motion: Generate text-guided 3d human motion by diffusion model. *arXiv preprint arXiv:2210.12315*, 2022.

[19] M. Zhang, Z. Cai, L. Pan, F. Hong, X. Guo, L. Yang, and Z. Liu. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, 2022.

[20] Y. Yuan, J. Song, U. Iqbal, A. Vahdat, and J. Kautz. Physdiff: Physics-guided human motion diffusion model. *arXiv preprint arXiv:2212.02500*, 2022.

[21] Y. Yuan and K. Kitani. Residual force control for agile human behavior imitation and extended motion synthesis. *Advances in Neural Information Processing Systems*, 33:21763–21774, 2020.

[22] W. Yu, G. Turk, and C. K. Liu. Learning symmetric and low-energy locomotion. *ACM Transactions on Graphics*, 37(4):1–12, Jul 2018. ISSN 1557-7368. doi:10.1145/3197517.3201397. URL http://dx.doi.org/10.1145/3197517.3201397.

[23] M. Al Borno, M. de Lasa, and A. Hertzmann. Trajectory optimization for full-body movements with complex contacts. *IEEE Transactions on Visualization and Computer Graphics*, 19(8): 1405–1414, 2013. doi:10.1109/TVCG.2012.325.

[24] I. Mordatch, E. Todorov, and Z. Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph.*, 31(4), jul 2012. ISSN 0730-0301. doi:10.1145/2185520.2185539. URL https://doi.org/10.1145/2185520.2185539.

[25] 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.*, 37(4), jul 2018. ISSN 0730-0301. doi:10.1145/3197517.3201311. URL https://doi.org/10.1145/3197517.3201311.

[26] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes. DReCon: Data-driven responsive control of physics-based characters. *ACM Trans. Graph.*, 38(6), nov 2019. ISSN 0730-0301. doi:10.1145/3355089.3356536. URL https://doi.org/10.1145/3355089.3356536.

[27] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa. AMP: Adversarial motion priors for stylized physics-based character control. *ACM Trans. Graph.*, 40(4), jul 2021. ISSN 0730-0301. doi:10.1145/3450626.3459670. URL https://doi.org/10.1145/3450626.3459670.

[28] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

[29] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel. Adversarial motion priors make good substitutes for complex reward functions. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2022. doi:10.1109/iros47612.2022.9981973. URL http://dx.doi.org/10.1109/IROS47612.2022.9981973.

[30] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[31] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[32] A. Ajay, Y. Du, A. Gupta, J. Tenenbaum, T. Jaakkola, and P. Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[33] Y. Dai, M. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

[34] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Jul 2018. doi: 10.24963/ijcai.2018/687. URL http://dx.doi.org/10.24963/ijcai.2018/687.

[35] A. Edwards, H. Sahni, Y. Schroecker, and C. Isbell. Imitating latent policies from observation. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1755–1763. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/edwards19a.html.

[36] 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, 2017.

[37] F. Torabi, G. Warnell, and P. Stone. Generative adversarial imitation from observation, 2018.

[38] H. Karnan, F. Torabi, G. Warnell, and P. Stone. Adversarial imitation learning from video using a state observer. *2022 International Conference on Robotics and Automation (ICRA)*, May 2022. doi:10.1109/icra46639.2022.9811570. URL http://dx.doi.org/10.1109/icra46639.2022.9811570.

[39] F. Liu, Z. Ling, T. Mu, and H. Su. State alignment-based imitation learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rylrdxHFDr.

[40] T. Gangwani and J. Peng. State-only imitation with transition dynamics mismatch. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJgLLyrYwB.

[41] L. Viano, Y.-T. Huang, P. Kamalaruban, C. Innes, S. Ramamoorthy, and A. Weller. Robust learning from observation with model misspecification. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '22, page 1337–1345, Richland, SC, 2022. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

[42] S. Kim, M. Sorokin, J. Lee, and S. Ha. Human motion control of quadrupedal robots using deep reinforcement learning. *arXiv preprint arXiv:2204.13336*, 2022.

[43] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[44] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[45] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[46] L. M. Mescheder, A. Geiger, and S. Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, 2018.

[47] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[48] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[49] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021.