GLiDE: Generalizable Quadrupedal Locomotion in Diverse Environments with a Centroidal Model

Zhaoming Xie[1,2], Xingye Da[2], Buck Babich[2],
Animesh Garg[2,3], and Michiel van de Panne[1]

[1] University of British Columbia
[2] NVIDIA
[3] University of Toronto, Vector Institute

Abstract. Model-free reinforcement learning (RL) for legged locomotion commonly relies on a physics simulator that can accurately predict the behaviors of every degree of freedom of the robot. In contrast, approximate reduced-order models are commonly used for many model predictive control strategies. In this work we abandon the conventional use of high-fidelity dynamics models in RL and we instead seek to understand what can be achieved when using RL with a much simpler centroidal model when applied to quadrupedal locomotion. We show that RL-based control of the accelerations of a centroidal model is surprisingly effective, when combined with a quadratic program to realize the commanded actions via ground contact forces. It allows for a simple reward structure, reduced computational costs, and robust sim-to-real transfer. We show the generality of the method by demonstrating flat-terrain gaits, stepping-stone locomotion, two-legged in-place balance, balance beam locomotion, and direct sim-to-real transfer.

Keywords: Legged Locomotion · Reinforcement Learning · Centroidal Model

1 Introduction

Tremendous progress has been made recently in legged locomotion, as achieved using both model predictive control (MPC) and reinforcement learning (RL). MPC methods leverage modern optimization techniques and known models of the physics, possibly simplified, to synthesize responsive control at run time. However, they can be prone to local minima, can require substantial manual tuning, and are difficult to generalize to complex terrains and rich perceptual streams. Moreover, real-time MPC often uses linear models for computational efficiency, making it non-trivial to represent the legged system's nonlinear and hybrid nature. Alternatively, model-free methods such as RL utilize Monte-Carlo sampling strategies and can learn policies for general tasks. This comes at the expense of careful system modeling, extensive offline simulation using the full dynamics model, and careful reward-crafting to produce results that are non-destructive for physical robots.

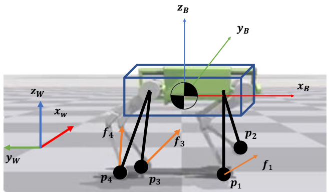

Fig. 1: Our centroidal model. Control is realized by generating ground reaction forces at foot locations.

In this paper, we present GLiDE, Generalizable Quadrupedal Locomotion in Diverse Environments with a Centroidal Model, where we eschew the conventional wisdom of applying RL to a high-fidelity model, and instead we ask what can be achieved when RL is used with a highly-abstracted centroidal model, as shown in Fig. 1. The action consists of the body linear and angular accelerations, which are realized by the ground reaction forces (GRFs), as computed via a quadratic program (QP). We further assume a specified gait pattern and a foot-placement function. Taken together, these choices allow for a simple task reward specification, in contrast to the more complex reward commonly required for full-model RL. Additional constraints such as no-slip constraints and leg lengths are enforced via the QP and the foot placement. The GRFs are realized on the full model using the Jacobian transpose for the stance legs to compute joint torques, and a trajectory-tracking approach for the swing legs. A system overview is given in Fig. 2. The resulting policies are validated on simulations of the Laikago and A1[4] robots as well as on a physical A1 robot. Please refer to the video for these results.

Our core contributions are as follows:

- We introduce an RL framework for learning control policies suited for centroidal dynamics models. This enables the anticipatory behavior required for quadrupedal locomotion and balance tasks, and is therefore a promising alternative to RL methods that require accurate full body dynamics models. The benefits of using a centroidal model include simple reward design, efficient simulation during training, and flexible and robust motion control.
- We introduce a novel action space for quadrupedal RL that operates in the centroidal model of the robot. Conversion between centroidal action space and the joint action space is realized via control methods such as quadratic programming and Jacobian transpose.
- We demonstrate the effectiveness of our approach by solving locomotion with multiple gaits, stepping stone scenarios, balance beam locomotion, and two-legged balancing. We also show successful transfer to a physical robot.

2 Related Work

Our work combines techniques used in quadrupedal locomotion in MPC and learning literature to solve challenging tasks. A comparison with some key papers is provided in Table 1. In this section we review more related work in this area.

[4] Laikago and A1 are quadrupedal robots made by Unitree Robotics.

	Model Used	Reward /Cost Function	Demonstrated Tasks	Training Time (Flat Terrain)
Centroidal MPC [1,8,50]	Linearized Centroidal (✓)	Linear Quadratic Regulator (✓)	Flat, Back Flip	n/a
Centroidal PD [9,28,41]	Centroidal	n/a	Flat	n/a
Laikago RL [14,15]	Full Physics	12 terms (✗)	Flat	6-8 hours
Anymal RL [12,16,17]	Full Physics	9-12 terms (✗)	Flat, Get Up, Rough Terrain	3-4 hours
DeepGait [22]	Centroidal + Full Physics (✓)	10 terms (✗)	Flat, Stepping Stone, Stairs	58 hours
GLiDE (ours)	Centroidal	6 terms (✓)	Flat, Two-Legged Balancing Stepping Stone, Balance Beam	1-2 hours

Table 1: We compare with recent work on quadrupedal locomotion. MPC often uses a centroidal model for efficient control synthesis but frequently fails to generalize to challenging scenarios. RL makes use of accurate full-body simulations and can handle challenging scenarios at the expense of significant computational cost and complicated reward structures. We use the centroidal model for RL to efficiently train policies for challenging tasks with a simple reward function.

Reduced-Order Models for Legged Locomotion Due to the complexity of legged robots, MPC approaches often make use of reduced-order models for control synthesis. An inverted pendulum and variations thereof are commonly used as abstract models of bipedal robots to synthesize control policies, e.g., [3–7]. A centroidal model can be used to generate control policies for bipeds [46–49] and quadrupeds [1,8,9,50]. To further simplify control synthesis, reference motions based on these models are generated offline, and feedback is then realized around these assuming local linearity. Direct nonlinear optimization can also be used for MPC at the cost of a slower control frequency [10] or additional heuristic objectives to guide the optimization [11] out of local minima. Although centroidal model has been used extensively in the MPC literature, it is rarely used for RL, where a very detailed, accurate model is often assumed to be necessary. In this paper, we employ the centroidal dynamics model for RL. The resulting policy directly optimizes for long-horizon nonlinear problems without the aid of a reference trajectory or heuristic cost, and can traverse more challenging terrains such as stepping stone compared to MPC approaches.

Deep Reinforcement Learning for Quadrupedal Robots Deep RL has become a viable approach for synthesizing control policies for quadrupedal robots. It is often realized via a comprehensive simulation of the robots, e.g., [12,13,19]. Rewards designed based on reference trajectories [14,15] or carefully tuned reward terms [16–18,42] are necessary to regularize undesirable behaviors as to be feasible for a physical robot. Training a policy often requires millions to billions of transition tuples of the full physics simulation. In this paper, we do not rely on a multibody dynamics simulator to train our policies. Instead, we simulate a low-cost centroidal dynamics model, which can be easily parallelized without resorting to special-purpose GPU physics simulators, e.g., [20,21]. When used in combination with a simple foot placement strategy, it requires significantly less computation for scenarios that require complex contact modeling, such as stepping stone and balance beam traversal, as compared to recent work, e.g., [22,23].

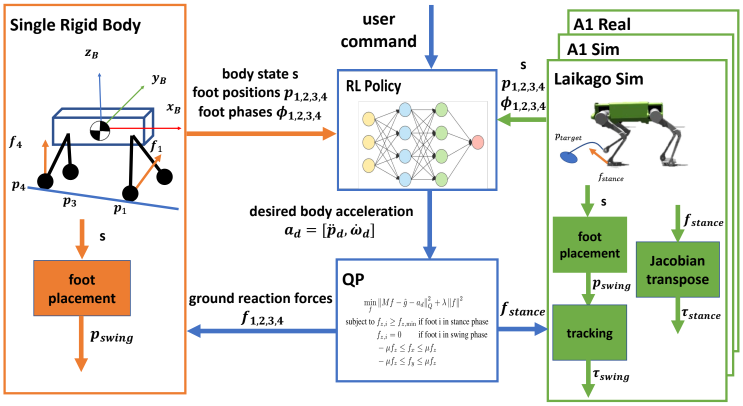

Fig. 2: Overview of our system. A control policy is trained with RL on the centroidal dynamics. The policy generates desired body acceleration commands, which are transformed to ground reaction forces using a QP. The resulting policy is then realized on the full-order model using the Jacobian transpose for stance leg control and trajectory tracking for swing leg control. All components are executed at 100 Hz.

Our work has some parallels with model-based RL, as we do not rely on a full physics simulator or a physical robot to perform rollouts. Recent work using model-based RL for legged robot control still needs to collect simulation data or physical robot data to learn either a transition model that contains the full state [24] or center of mass [25] information. We rely instead on the simple centroidal dynamics to perform rollouts, which does not require data from a full model or a physical robot. Our simulation is easily adapted to robots with different physical parameters or morphology by changing the compact set of parameters that define the centroidal model, such as the mass and inertia.

Combination of MPC and Learning MPC comes with the benefit of explainability but often at the expense of significant online computation, manual tuning, and feature engineering. Alternatively, learned RL policies enable fast online computation and can in principle cope with complex terrains and rich perceptual observations but suffer from expensive offline computation and a control policy that is highly specific to the given robot morphology.

A combination of MPC and RL can offer the best of both approaches. The use of RL in task spaces together with an MPC controller has yielded generalization in learning for manipulation skills [43, 44]. An MPC controller can be used to collect data offline and then efficiently queried online using supervised learning, e.g., [26, 27, 31]. Heuristics or dynamics can be learned to enable faster optimization or planning, e.g., [11, 32]. Hierarchical control structures have been proposed to allow MPC controllers and RL policies to operate at different time scales to leverage their respective advantages, e.g., [28–30, 33, 45].

Our work also combines RL and MPC. A high-level learned policy operates in the space of centroidal dynamics, producing desired linear and angular accelerations. The low-level control policy uses MPC based methods to realize these high-level commands, aided by a quadratic program that transcribes the desired accelerations into appropriate ground reaction forces.

3 GLiDE: RL with Centroidal Model

In this section, we describe GLiDE for learning policies using a centroidal dynamics model and the related approach for realizing the resulting policy on the full robot model. We describe the centroidal dynamics simulation, the control of the centroidal dynamics by transforming the desired body acceleration to ground reaction forces via a quadratic program (QP), and the use of a Raibert-style heuristic for foot placement. Further, we detail the procedure for training RL policies using GLiDE, and policy transfer to full-order model using MPC based approaches. The system is outlined in Fig. 2.

Centroidal Dynamics Model With Virtual Legs Our centroidal dynamics model consists of a single rigid body with four massless legs attached; see Fig. 1. Note that this is a fixed centroidal model, where we ignore the dependency of the COM position and inertia tensor on the pose of the legs, as a result of massless legs assumption. Related literature also refers to this as a single rigid body model. The rigid body has mass m and inertia $I \in \mathbb{R}^{3\times 3}$. Each leg has a phase variable ϕ that advances linearly in time. The state of the rigid body $s = [p, \dot{p}, R, \omega]$ consists of the linear position and velocity $p, \dot{p} \in \mathbb{R}^3$, orientation $R \in SO(3)$, and angular velocity $\omega \in \mathbb{R}^3$, expressed in a global coordinate frame. Following [1], we represent R with a rotation matrix.

We follow the derivation from [1] to simulate the centroidal dynamics. For completeness, we describe our notation and key equations used for simulation. For each leg i, ground reaction forces $f_i \in \mathbb{R}^3$ can be generated to drive the state of the rigid body. When walking on flat ground, if leg i is in swing phase, $f_i = 0$; otherwise f_i needs to obey the friction cone constraints in order to prescribe feasible forces. Given the foot position $p_i \in \mathbb{R}^3$ and f_i for each leg i, the net force and torque $f_{\text{net}}, \tau_{\text{net}} \in \mathbb{R}^3$ can be calculated as:

$$f_{\text{net}} = \sum_{i=1}^{4} f_i - mg,\ \tau_{\text{net}} = \sum_{i=1}^{4} \hat{r_i} f_i \tag{1}$$

where g is the gravitational constant, $r_i = p_i - p$, and $\hat{\cdot} : \mathbb{R}^3 \to \mathfrak{so}(3)$ defines a mapping from a vector to its skew-symmetric matrix form.

The Euler integration update of the rigid body state can then be written as

$$\begin{aligned} p &= p + \dot{p}\Delta t, & \dot{p} &= \dot{p} + \frac{1}{m} f_{\text{net}} \Delta t \\ R &= R \exp(\widehat{\omega \Delta t}), & \omega &= \omega + I^{-1}(R^T \tau_{\text{net}} - \hat{\omega} I \omega)\Delta t \end{aligned} \tag{2}$$

where Δt is the simulation time step and the exponential map $\exp : \mathfrak{so}(3) \to SO(3)$ ensures that R stays on the $SO(3)$ manifold.

We define a phase ϕ_i for each foot i to be an integer that increments by 1 at each time step. We define a cycle time, T, and a swing time, T_{swing}. Foot i is in swing phase if $(\phi_i \mod T) \leq T_{\text{swing}}$ and in stance phase otherwise.

Quadratic Programming Based Control While the centroidal model is driven by the ground reaction forces (GRFs), it is often unintuitive to directly provide the set of GRFs. Furthermore, in the RL setting, it is difficult to guarantee inequality constraints such as a friction cone if we use the GRFs as the control action. Instead, we choose the control action to be the desired acceleration $a_d = [\ddot{p}_d, \dot{\omega}_d]$. We then solve a QP to transcribe this into a set of GRFs: $f = [f_1, f_2, f_3, f_4]$ that satisfy an approximation of friction cone constraints:

$$\begin{aligned} &\min_f \|Af - \hat{g} - a_d\|_Q^2 + \lambda \|f\|^2 \\ \text{subject to } & f_{z,i} \geq f_{z,\min} \text{ if foot i in stance phase} \\ & f_{z,i} = 0 \qquad \text{if foot i in swing phase} \\ & -\mu f_z \leq f_x \leq \mu f_z \\ & -\mu f_z \leq f_y \leq \mu f_z \end{aligned} \tag{3}$$

where $A \in \mathbb{R}^{6\times 12}$ is the matrix that transforms GRFs to body acceleration, which is a function of foot position p_{foot} and the centroidal state s, $\hat{g} = [g, 0_3] \in \mathbb{R}^6$ is the augmented gravity vector, $Q \succ 0$ is for weight adjustment of the cost terms, μ is the friction coefficient of the terrain and $\lambda > 0$ regularizes the GRFs used. The resulting GRFs are then used to simulate the centroidal dynamics.

Foot Placement Strategy We use a simple Raibert-style heuristic [36] for foot placement. Specifically, in the centroidal model, for each foot i, if $(\phi_i \mod T) = 0$, i.e., at the instant where it switches from stance foot to swing foot, the foot position relative to the body on the xy-plane, $r_{i,xy}$, will be updated following:

$$r_{i,x} = r_{\text{ref},i,x} + k_{\text{foot},x}\,\dot{p}_x, \,, \; r_{i,y} = r_{\text{ref},i,y} + k_{\text{foot},y}\,\dot{p}_y, \tag{4}$$

where $r_{\text{ref},i}$ are the default foot placements, typically obtained from the neutral standing pose of the robot, and k_{foot} is a constant that adjusts the foot placement based on the velocity of the body $\dot{p}_{xy}$. In addition, a leg-length constraint is applied such that when $\|p_{\text{foot}} - p\| \geq l_{\max}$ for some maximum leg length $l_{\max}$, the point closest to p_{foot} that satisfies the leg length constraint will be used. Since the GRF for the swing foot is zero, the swing foot is assumed to not affect the dynamics of the centroidal model until it becomes the stance foot again. Importantly, the control policy based on the centroidal model can observe the planned foot placement positions, which allows it to anticipate the terrain, without having direct access to the terrain map.

Reinforcement Learning with Centroidal Model We apply RL on the centroidal model to synthesize the control policy. We use an actor-critic algorithm optimized with Proximal Policy Optimization [2]. The input to the policy is the observations $\{s, \{r_i\}, \{\phi_i \mod T\}\}$, where $s = \{p_z, \dot{p}, R, \omega\}$ is the state of the rigid body excluding displacement; $r_i = p_i - p$ is the stance foot position or swing-leg target position, relative to the center of mass; and $\phi_i \mod T$ is the normalized phase of foot i. The reward is based on the rigid body state without concern for many of the issues that lead to more complex reward for RL solutions based on a full model.

Since we have the Euler integration result in analytic form, the centroidal dynamics can be implemented on a GPU to allow thousands of parallel simulations to collect data. A bottleneck for simulation with a control policy in our setting is that we need to solve a small QP for each time step. We currently find no off-the-shelf QP solver that can reliably solve thousands of small QPs in parallel. We therefore implement a custom parallel QP solver specific to our needs. The solver is implemented with PyTorch [34] for easy prototyping and is based on the interior point method [35]. In our experiments, we run 1600 simulations in parallel, with each environment step taking around 0.25 s on average, including reward computation and policy query. This allows for a data collection rate of 6-7k steps per second. The QP solves consume about 80% of the time, thus in principle we could achieve a significant additional speedup with further improvements to the QP solver. The benefit becomes more obvious in the presence of irregular terrain such as a field of stepping stones. Instead of simulating complex contact dynamics between a height field and the robot links, as in related work [17, 22], we need only select the appropriate foothold at negligible cost.

Realization on the Full Model Since training the policy does not rely on any knowledge of how the low-level controller will be implemented on the full-order model, any controller can be used as long as it can generate the joint commands to realize the required GRFs and the given swing foot placements. Recent work uses another learned policy to track the behavior of some reduced-order models, e.g., [5,22], but the policy will only be applicable to the specific robot it is trained on. We use a simple approach that only leverages knowledge of the kinematics.

For stance foot control, we transform the GRFs into joint torques using the Jacobian transpose: $\tau_{\text{stance}} = J^T f$, where J is the stance feet position Jacobian with respect to motor states (without considering the state of the floating base).

We realize the foot placement strategy for the swing foot using a virtual force, f_{PD}, arising from a cartesian-space PD controller. If $(\phi_i \mod T) = 0$, we set the swing foot target to be the updated position p_i as in the centroidal model; a spline curve parameterized by time is then constructed to connect the current foot position and p_i, with a given foot height clearance. The curve is tracked using $\tau_{\text{swing}} = J^T f_{\text{PD}}$. If early foot contact is detected, the torques for the leg are set to zero until the planned stance phase. For the case of late contact, there is no special treatment, as a desired GRF will naturally result in leg extension.

GLiDE-full During training, we can use the full model for simulation while keeping other components the same as GLiDE. We call this variant: GLiDE-Full. We use the same observation and action space and the action will be transformed into joint torques as discussed in the previous paragraph for simulation. There are trade off between using GLiDE and GLiDE-full. GLiDE allows for easy parallel simulation at the cost of compromised model fidelity while GLiDE-full provide precise modeling at the cost of more expensive data collection.

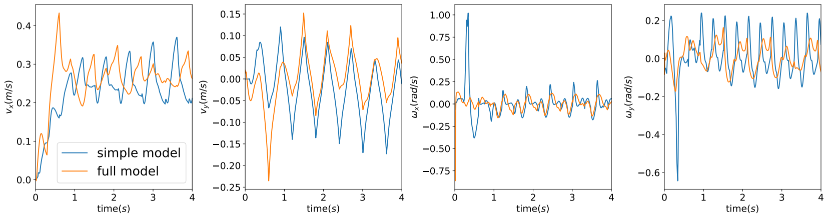

Fig. 3: Linear velocity in the x-y plane as well as the roll and pitch velocity of full-order model and centroidal model of Laikago. The general behaviors are close.

4 Results

To demonstrate the generality of GLiDE, we apply GLiDE to learn control policies for Laikago and A1. The mass and inertia of the centroidal model as well as default foot placement locations and foot length constraints are adapted from the robot parameters provided by the manufacturer. Policies are parametrized with two-layer feedforward neural networks, where the hidden layer has size 128 with ReLU activation. TanH is applied to the output.

Our method benefits from an exceptionally simple reward structure. We use the following reward across all the tasks presented: $r = 0.5r_p + 0.5r_o$, where r_p regularizes the linear velocity and body height, and r_o regularizes the body orientation. We reward the policy for moving forward/backward in a straight line at desired velocity while keeping a default orientation. More specifically:

$$\begin{aligned} r_p &= \exp(-10(\dot{p} - \dot{p}_{x,d})^2 - 50\dot{p}_y^2 - 50(p_z - p_{z,d})^2) \\ r_o &= \exp(-10 \left\| \Theta \right\|^2), \end{aligned}$$

where $p_{z,d}$ is the nominal height of the robot, and Θ is the Euler angle representation of the robot orientation. Aside from locomotion on flat terrain, we also learn control policies for challenging tasks such as stepping stone traversal, navigating across a narrow balance beam, and two-legged balancing. We use IsaacGym [20] to test the policies on the simulated Laikago and A1 and we test some of the policies on a physical A1 robot to demonstrate robust sim-to-real. Please refer to the supplementary video for example motions.

4.1 Learning Challenging Tasks with GLiDE

Flat Terrain We first demonstrate our system for the generation of locomotion on flat terrain. A trotting gait can be realized by synchronizing the leg phases of diagonal legs. We set the cycle time $T = 60$ and swing time $T_{\text{swing}} = 30$. We initialize $\phi_1 = 0, \phi_4 = 0$ and $\phi_2 = 30, \phi_3 = 30$ so the swing phase for nondiagonal legs are offset by half a cycle. Similarly, we can realize a walking gait with $T = 120, T_{\text{swing}} = 30$, and initialize $\phi_1 = 0, \phi_2 = 90, \phi_3 = 60, \phi_4 = 30$.

Control policies can be learned with the centroidal models of Laikago and A1 in 1–2 hours for trotting (up to 0.5 m/s) and walking (up to 0.2 m/s). These policies are then deployed directly on their respective full-order robot models.

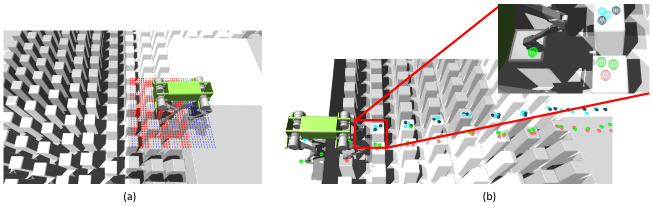

Fig. 4: Illustration of the stepping stone. (a) The system gets a local map of the terrain around the robot, illustrated by the grid of red and blue points. Blue points indicate regions suitable for foot placement while the red points indicate otherwise. The edges of the stepping stones are also deemed unsuitable to encourage safety. The foot placement strategy chooses the feasible point that is closest to the default foot placement, as given by the Raibert heuristic. (b) Footstep pattern of Laikago trotting across a stepping stone terrain. Different colors indicate stepping locations of different feet.

We compare the behaviors of the simplified model and full model in Fig. 3. While the exact trajectories differ, they indicate similar general behavior.

Since we are learning with a centroidal model, the policy is agnostic to the morphology of the robot. We test the learned policy on a version of Laikago where the hind-leg knee configuration is inverted. The learned policy works directly for the inverted-configuration robot. We also train a trotting policy for the default Laikago, utilizing the full physics simulation and directly generate commands at the joint control level, similar to [14,15]. This policy with joint level action space works well with the default configuration of Laikago, as expected, but fails to generalize to the inverted configuration due to the learned control policy being very specific to the morphology it was trained on.

We also test the generality of our approach by testing a policy trained with Laikago robot parameters on the A1 robot. The sizes of these two robots are not significantly different. We therefore choose not to apply scaling to the action space. However, we do scale the most relevant input features to the policy, namely the body height and foot positions. This policy achieves a normalized reward of 0.93 on the Laikago robot and a normalized reward of 0.90 on the A1 robot. Directly applying the policy without scaling the input causes the A1 robot to fall over because the the input is not in the distribution seen during training.

Stepping Stones To showcase that GLiDE is compatible with tasks that require perception of the terrain, we train policies to drive the centroidal model across stepping stones. This terrain restricts the set of valid stepping locations, so the foot placement strategy needs to avoid infeasible locations on the ground. This task has been considered for quadrupedal locomotion in recent work using MPC for trotting [37] and static walking [38], and a simpler version of the task has been considered using RL [22]. We demonstrate the generality of GLiDE by solving it with both Laikago and A1, with either a trotting or a walking gait. Fig. 4 illustrates the stepping stone problem. As before, we use the Raibert-style heuristic to plan foot placement. However, if the planned foot placement is infeasible, the closest feasible position is chosen instead.

We train policies to traverse random stepping stone terrains with feasible footholds being placed between 10 cm to 20 cm apart for Laikago and 5 cm to

Method	Energy	Motion quality
GLiDE(Flat)	891 ± 21	good
GLiDE-full (Flat)	913 ± 21	good
Centroidal PD (Flat)	869 ± 4	good
Joint RL (Flat)	3476 ± 75	bad
GLiDE(Step)	1080 ± 46	good
GLiDE-full (Step)	1113 ± 52	good
Centroidal PD (Step)	1460 ± 70	bad

Table 2: Energy and motion quality of different methods. See the text for qualitative result for the motion quality assessment.

15 cm apart for A1. The training takes 4 hours, and the policies can be realized on the simulated full model of Laikago and A1. The policies also generalize to stepping stone patterns not seen during training. We command the robot to walk across the stepping stones while turning, while the trained policy has only seen straight-line forward motion. The policy successfully generalizes to this scenario.

Balance Beam Next we tackle the balance beam tasks to showcase the ability of our system to make control decisions that optimize performance over a time horizon. A similar capability was also demonstrated in [39] using MPC method.

In this task, the robot must walk on a narrow path with limited foot placement choice in the lateral direction. We train a balance beam policy for the centroidal model of the A1. During training, we restrict the foot placement in world coordinates with $|p_{i,y}| \leq 0.05\,\mathrm{m}$ while the nominal leg distance between left and right foot is 0.2 m. The y-coordinate of the rigid body p_y is also provided to the policy to avoid drift. We train a trotting policy with desired velocity $\dot{p}_{x,d} = 0.1\,\mathrm{m/s}$. The learned policy successfully transfers to the full simulated model of the A1 trotting across a balance beam.

Two-Legged Balancing We then consider the two-legged balancing task. This is more challenging than the balancing beam task since the robot cannot regain balance by taking steps. In the two-legged balancing scenario, we set $\phi_1, \phi_4 = 31$ and $\phi_2, \phi_3 = 0$ for all time such that the left front leg and the right rear leg are set to be the stance legs. We also include p_x, p_y as input to the policy. With the full-order model, the right front leg and the left hind leg are held in place in midair using position control, and the stance legs are controlled as before.

4.2 Sim-to-Real Tests

We transfer some of the policies to the physical A1 robot. These policies do not rely on a full-body physics simulator and thus avoid the most common types of overfitting problems. Sim-to-real transfer is as straightforward as the reduced-order to full model transfer. This further demonstrates the generality and robustness of our approach.

To generate gaits of different styles, we train a trotting policy on flat terrain with the default foot position $r_{\mathrm{ref},x} = \pm 0.1\,\mathrm{m}, r_{\mathrm{ref},y} = \pm 0.05\,\mathrm{m}$, twice as close to the center of mass as the default. We set $T = 40$ and $T_{\mathrm{swing}} = 20$ so the policy is trotting at a higher frequency. This policy can transfer to the full-order model

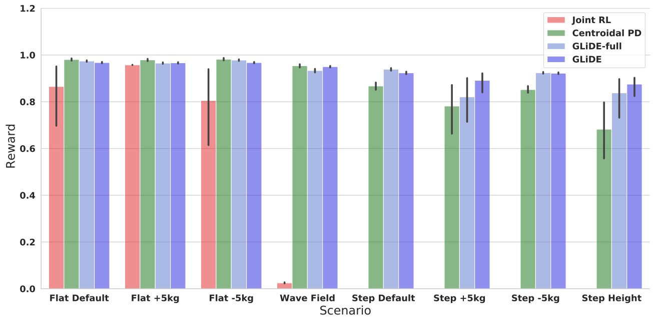

Fig. 5: Normalized reward of different methods. GLiDE and GLiDE-full are comparable to baseline in the default scenarios and perform better in challenging scenarios.

in simulation and directly transfer to the physical A1. The learned policies are also robust to perturbations. In the video, we show the physical robot is able to handle obstacle-strewn terrain using a policy trained only on flat terrain, with normal foot spacing and stepping frequency.

While we train policies to traverse stepping stones, we are not yet able to realize this on hardware due to lack of a localization system. We design a proxy test where a gap of 12 cm wide is placed in front of the robot. The foot placement strategy is modified to generate footholds that avoid this gap. The stepping stone policy is able to generate commands that enable the stance legs to stabilize the robot while the swing legs take larger than usual steps.

5 Evaluation

We compare GLiDE with alternatives using RL or MPC, as well as ablation on GLiDE-full. Policies are trained with Laikago, on flat terrain (Flat) and stepping stone (Step). We test the performance of different policies under different test scenarios. The statistics of 10 runs are recorded, where each run involves a policy running for 10 seconds in simulation. Variation between different runs include different initial body poses and different random seeds for terrain generation. Quantitative results are recorded in Fig. 5 and Table 2.

5.1 Test Scenarios

Body Mass Perturbation We perturb the mass of the body by ±5 kg, around 40% of the default body mass and record the normalized reward of the policies.

Height Variation We record the normalized reward of the policies walking against height perturbation. We create a wave field by adding sinusoidal height change along both x and y direction on the continous terrain. The wave period repeats in every 2.5 m and the peak to valley distance is 0.7 m comparing to the robot normal height of 0.4 m. For stepping stone, we create a random height field by passing a white noise height change into a second order low-pass filter. This returns a medium level random 1D signal and we apply it along both x and y direction with different random seed. This creates slopes of up to 20 degrees.

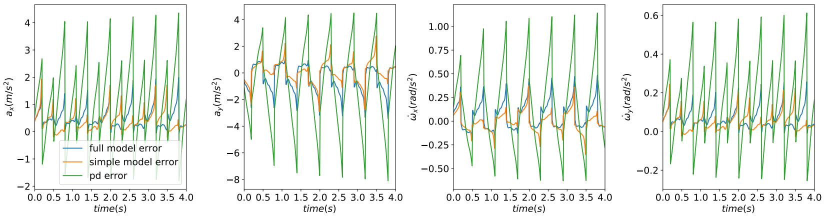

Fig. 6: Comparison of difference between the desired and the actual body acceleration. Due to the underactuation of the systems, non-zero error is unavoidable. We plot the error of a GLiDE policy on the centroidal model and the full-order model, and plot the error of a centroidal PD controller on the full-order model. The GLiDE policy produces commands with a much smaller discrepancy compared to a centroidal PD controller.

Energy Comparison We record $\frac{1}{T}\sum_{t=0}^{T-1}\|\tau_t\|^2$ where τ_t is the torque value of all joints at time step t, and T is the total number of time steps in 10 seconds.

Motion Quality We record the the z trajectory of the front left foot over 10 seconds and compute an energy spectral density in Fourier space.

5.2 Evaluation Results

RL baseline: Joint RL We train policies on the full model with joint torque as action space on flat terrain using the same reward. While these policies are able to obtain similar performance in term of reward, the policies fail to generalize to other scenarios and favor high-frequency shuffle gaits. We record the z trajectory of the front left foot for 10 s and compute an energy spectral density in Fourier space. The fraction of energy for $f > 10$ Hz is 0.13 for GLiDE and 0.68 for joint RL. Reward tuning is necessary to obtain suitable behavior, e.g, [12].

MPC baseline: Centroidal PD As an alternative to GLiDE, we implement a Centroidal PD controller used in [9, 28, 41]; with a_d is computed as

$$a_d = \begin{bmatrix} \ddot{p}_d \\ \dot{\omega}_d \end{bmatrix} = k_p \begin{bmatrix} p_d - p \\ \Theta_d - \Theta \end{bmatrix} + k_d \begin{bmatrix} \dot{p}_d - \dot{p} \\ \omega_d - \omega \end{bmatrix}, \tag{5}$$

with $p_d = [0, 0, p_{z,d}], \dot{p}_d = [\dot{p}_{x,d}, 0, 0]$, $\Theta_d = [0, 0, 0], \omega_d = [0, 0, 0]$ being the desired behavior while k_p and k_d are diagonal matrices defining the PD gain. Both GLiDE and Centroidal PD can produce the trotting and walking gaits on flat ground. However, Centroidal PD produces more aggressive motion. In Fig. 6, we plot $Mf - a$, the error term for the QP problem (Equation 3), for both the learned policy and the cenotridal PD controller with a trotting gait. Due to underactuation, the error is unavoidable, but the learned policy incurs much smaller error compared to a centroidal PD controller.

For the stepping stone tasks, centroidal PD can be tuned to navigate across the stepping stone terrain with $T = 40, T_{\text{swing}} = 20$ with a trotting gait on Laikago, which is unrealistically fast for the hardware. We failed to find a Centroidal PD controller that achieves a more realistic stepping frequency. We believe the failure is caused by aggressive tracking of the command, whereas the learned policy prioritizes safety over tracking error. One could also use convex MPC algorithm [8] that plans for a longer horizon instead of the Centroidal PD controller. However, convex MPC is only beneficial when significant flight phases are present in the desired motion [41]. In our scenarios, Centroidal PD and convex MPC perform similarly. Similar results are also demonstrated in [52].

For the balance beam task and the two-legged balancing task, centroidal PD falls sideways due to underactuation. Specialized MPC methods are necessary to solve these tasks, e.g., [37, 39], while GLiDE is able to solve all the tasks.

GLiDE-full For ablation on GLiDE, we also use GLiDE-full to train policies for flat terrain and stepping stone. The performance of GLiDE-full policies in principle will be an upper bound for the performance of GLiDE policies in the default scenario since the training environment and the testing environment is identical for GLiDE-full. In Figure 5, we show that GLiDE and GLiDE-full demonstrate similar performance on most testing scenarios while GLiDE performs more consistently on challenging scenarios such as Step Height and Step +5 kg.

6 Discussion and Future Work

We present a framework that combines RL with a centroidal model to learn policies for quadrupedal locomotion, as a simple (and therefore highly practical) alternative to the common strategy of using RL with a high-fidelity model. We validate our approach by efficiently learning locomotion policies for different quadrupedal robots and solving challenging tasks such as locomotion across stepping stones, two-legged balancing, and locomotion across a balance beam. We further demonstrate successful sim-to-real transfer. The method contributes to the understanding of what is required, in terms of model-fidelity, for RL-based skill design for quadrupedal robots.

There remain constraints that are not captured by a simplified model, including joint limits and torque limits. These could be taken into account by using the full model for training, with the caveat that the policy becomes specific to the full model. Obtaining maximal performance may also require explicitly taking these limits into account, as can in principal be achieved using full-model RL.

We rely on a simple heuristic for foot placement. This limits our ability to deal with more challenging terrain, e.g., more challenging variations of the stepping stone task. Incorporating systems [33, 40] that directly learn foot placement will be crucial for further improving the capability of our system. Our system can in principle also handle terrains with height variations, e.g., for climbing stairs. We are in the process of extending our framework for learning to traverse more general terrains, as well as incorporating a vision system onto the physical A1 robot to enable sim-to-real transfer for tasks such as stepping stone traversal.

Currently we need to train different policies for different scenarios. A unified policy that can handle all scenarios will be crucial for real world adoption. This can potentially be achieved via progressive learning [51].

References

1. Ding, Yanran, Abhishek Pandala, and Hae-Won Park. "Real-time model predictive control for versatile dynamic motions in quadrupedal robots." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.
2. Schulman, John, et al. "Proximal policy optimization algorithms." arXiv preprint arXiv:1707.06347 (2017).
3. Xiong, Xiaobin, and Aaron Ames. "3d underactuated bipedal walking via h-lip based gait synthesis and stepping stabilization." arXiv preprint arXiv:2101.09588 (2021).
4. Gong, Yukai, and Jessy Grizzle. "Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller." arXiv preprint arXiv:2008.10763 (2020).
5. Green, Kevin, et al. "Learning spring mass locomotion: Guiding policies with a reduced-order model." IEEE Robotics and Automation Letters 6.2 (2021).
6. Kajita, Shuuji, and Kazuo Tani. "Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode." Proceedings. 1991 IEEE International Conference on Robotics and Automation.
7. Koolen, Twan, et al. "Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models." The international journal of robotics research 31.9 (2012): 1094-1113.
8. Di Carlo, Jared, et al. "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control." 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018.
9. Gehring, Christian, et al. "Control of dynamic gaits for a quadrupedal robot." 2013 IEEE international conference on Robotics and automation. IEEE, 2013.
10. Grandia, Ruben, et al. "Frequency-aware model predictive control." IEEE Robotics and Automation Letters 4.2 (2019): 1517-1524.
11. Bledt, Gerardo, and Sangbae Kim. "Extracting legged locomotion heuristics with regularized predictive control." 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020.
12. Hwangbo, Jemin, et al. "Learning agile and dynamic motor skills for legged robots." Science Robotics 4.26 (2019): eaau5872.
13. Tan, Jie, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. "Sim-to-real: Learning agile locomotion for quadruped robots." arXiv preprint arXiv:1804.10332 (2018).
14. Peng, Xue Bin, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. "Learning agile robotic locomotion skills by imitating animals." arXiv preprint arXiv:2004.00784 (2020).
15. Xie, Zhaoming, Xingye Da, Michiel van de Panne, Buck Babich, and Animesh Garg. "Dynamics randomization revisited: A case study for quadrupedal locomotion." In 2021 IEEE International Conference on Robotics and Automation (ICRA).
16. Gangapurwala, Siddhant, Alexander Mitchell, and Ioannis Havoutis. "Guided constrained policy optimization for dynamic quadrupedal robot locomotion." IEEE Robotics and Automation Letters 5.2 (2020): 3642-3649.
17. Lee, Joonho, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning quadrupedal locomotion over challenging terrain." Science robotics 5, no. 47 (2020): eabc5986.
18. Yang, Chuanyu, Kai Yuan, Qiuguo Zhu, Wanming Yu, and Zhibin Li. "Multi-expert learning of adaptive legged locomotion." Science Robotics 5, no. 49 (2020).

19. Paigwar, Kartik, Lokesh Krishna, Sashank Tirumala, Naman Khetan, Aditya Sagi, Ashish Joglekar, Shalabh Bhatnagar, Ashitava Ghosal, Bharadwaj Amrutur, and Shishir Kolathaya. "Robust quadrupedal locomotion on sloped terrains: A linear policy approach." arXiv preprint arXiv:2010.16342 (2020).
20. Makoviychuk, Viktor, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller et al. "Isaac gym: High performance gpu-based physics simulation for robot learning." arXiv preprint arXiv:2108.10470 (2021).
21. Heiden, Eric, David Millard, Erwin Coumans, Yizhou Sheng, and Gaurav S. Sukhatme. "NeuralSim: Augmenting differentiable simulators with neural networks." In 2021 IEEE International Conference on Robotics and Automation.
22. Tsounis, Vassilios, Mitja Alge, Joonho Lee, Farbod Farshidian, and Marco Hutter. "Deepgait: Planning and control of quadrupedal gaits using deep reinforcement learning." IEEE Robotics and Automation Letters 5, no. 2 (2020): 3699-3706.
23. Jain, Deepali, Atil Iscen, and Ken Caluwaerts. "From pixels to legs: Hierarchical learning of quadruped locomotion." arXiv preprint arXiv:2011.11722 (2020).
24. Yang, Y., Caluwaerts, K., Iscen, A., Zhang, T., Tan, J., and Sindhwani, V. (2020, May). Data efficient reinforcement learning for legged robots. In Conference on Robot Learning (pp. 1-10). PMLR.
25. Li, Tianyu, Roberto Calandra, Deepak Pathak, Yuandong Tian, Franziska Meier, and Akshara Rai. "Planning in learned latent action spaces for generalizable legged locomotion." IEEE Robotics and Automation Letters 6, no. 2 (2021): 2682-2689.
26. Da, Xingye, and Jessy Grizzle. "Combining trajectory optimization, supervised machine learning, and model structure for mitigating the curse of dimensionality in the control of bipedal robots." The International Journal of Robotics Research 38, no. 9 (2019): 1063-1097.
27. Carius, Jan, Farbod Farshidian, and Marco Hutter. "Mpc-net: A first principles guided policy search." IEEE Robotics and Automation Letters 5, no. 2 (2020).
28. Da, Xingye, Zhaoming Xie, David Hoeller, Byron Boots, Animashree Anandkumar, Yuke Zhu, Buck Babich, and Animesh Garg. "Learning a contact-adaptive controller for robust, efficient legged locomotion." arXiv preprint arXiv:2009.10019 (2020).
29. Li, Tingguang, Krishnan Srinivasan, Max Qing-Hu Meng, Wenzhen Yuan, and Jeannette Bohg. "Learning hierarchical control for robust in-hand manipulation." In 2020 IEEE International Conference on Robotics and Automation (ICRA).
30. Li, Tianyu, Hartmut Geyer, Christopher G. Atkeson, and Akshara Rai. "Using deep reinforcement learning to learn high-level policies on the atrias biped." In 2019 International Conference on Robotics and Automation (ICRA).
31. Viereck, Julian, and Ludovic Righetti. "Learning a centroidal motion planner for legged locomotion." In 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 4905-4911. IEEE, 2021.
32. Lin, Yu-Chi, Brahayam Ponton, Ludovic Righetti, and Dmitry Berenson. "Efficient humanoid contact planning using learned centroidal dynamics prediction." In 2019 International Conference on Robotics and Automation (ICRA).
33. Meduri, Avadesh, Majid Khadiv, and Ludovic Righetti. "DeepQ Stepper: A framework for reactive dynamic walking on uneven terrain." In 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 2099-2105. IEEE, 2021.
34. Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen et al. "Pytorch: An imperative style, high-performance deep learning library." Advances in neural information processing systems 32 (2019).
35. Boyd, Stephen and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

36. Raibert, Marc H. Legged robots that balance. MIT press, 1986.
37. Grandia, Ruben, Andrew J. Taylor, Aaron D. Ames, and Marco Hutter. "Multi-layered safety for legged robots via control barrier functions and model predictive control." In 2021 IEEE International Conference on Robotics and Automation.
38. Mastalli, Carlos, Ioannis Havoutis, Michele Focchi, Darwin G. Caldwell, and Claudio Semini. "Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control." IEEE Transactions on Robotics 36, no. 6 (2020): 1635-1648.
39. Gonzalez, Carlos, Victor Barasuol, Marco Frigerio, Roy Featherstone, Darwin G. Caldwell, and Claudio Semini. "Line walking and balancing for legged robots with point feet." In 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3649-3656. IEEE, 2020.
40. Gangapurwala, Siddhant, Mathieu Geisert, Romeo Orsolino, Maurice Fallon, and Ioannis Havoutis. "Rloc: Terrain-aware legged locomotion using reinforcement learning and optimal control." arXiv preprint arXiv:2012.03094 (2020).
41. Bledt, Gerardo, Matthew J. Powell, Benjamin Katz, Jared Di Carlo, Patrick M. Wensing, and Sangbae Kim. "MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot." In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2245-2252. IEEE, 2018.
42. Bharadhwaj, Homanga, Aviral Kumar, Nicholas Rhinehart, Sergey Levine, Florian Shkurti, and Animesh Garg. "Conservative safety critics for exploration." arXiv preprint arXiv:2010.14497 (2020).
43. Martín-Martín, Roberto, Michelle A. Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. "Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks." In 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).
44. Allshire, Arthur, et al. "Laser: Learning a latent action space for efficient reinforcement learning." 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021.
45. Duan, Helei, et al. "Learning task space actions for bipedal locomotion." 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021.
46. Kuindersma, Scott, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot." Autonomous robots 40, no. 3 (2016): 429-455.
47. Koolen, Twan, et al. "Design of a momentum-based control framework and application to the humanoid robot atlas." International Journal of Humanoid Robotics 13.01 (2016): 1650007.
48. Feng, Siyuan, et al. "Optimization based controller design and implementation for the atlas robot in the darpa robotics challenge finals." 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids). IEEE, 2015.
49. García, Gabriel, Robert Griffin, and Jerry Pratt. "MPC-based Locomotion Control of Bipedal Robots with Line-Feet Contact using Centroidal Dynamics." In 2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids).
50. García, Gabriel, Robert Griffin, and Jerry Pratt. "Time-Varying Model Predictive Control for Highly Dynamic Motions of Quadrupedal Robots." In 2021 IEEE International Conference on Robotics and Automation.
51. Berseth, Glen, et al. "Progressive reinforcement learning with distillation for multi-skilled motion control." arXiv preprint arXiv:1802.04765 (2018).
52. Agrawal, Ayush, et al. "Vision-aided Dynamic Quadrupedal Locomotion on Discrete Terrain using Motion Libraries." arXiv preprint arXiv:2110.00891 (2021).