RL + Model-based Control: Using On-demand Optimal Control to Learn Versatile Legged Locomotion

Dongho Kang, Jin Cheng, Miguel Zamora, Fatemeh Zargarbashi, and Stelian Coros

***Abstract*— This letter presents a control framework that combines model-based optimal control and reinforcement learning (RL) to achieve versatile and robust legged locomotion. Our approach enhances the RL training process by incorporating on-demand reference motions generated through finite-horizon optimal control, covering a broad range of velocities and gaits. These reference motions serve as targets for the RL policy to imitate, resulting in the development of robust control policies that can be learned efficiently and reliably. Moreover, by considering whole-body dynamics, RL overcomes the inherent limitations of modelling simplifications. Through simulation and hardware experiments, we demonstrate the robustness and controllability of the RL training process within our framework. Furthermore, our method demonstrates the ability to generalize reference motions and handle more complex locomotion tasks that may pose challenges for the simplified model, leveraging the flexibility of RL.**

I. Introduction

Model-based optimal control (MOC) and *reinforcement learning* (RL) are two powerful tools widely used in legged locomotion research. In MOC, designers make various design choices regarding motion primitives, system dynamics, and control parameters based on simplifying assumptions [1]. This approach offers the advantage of transparency and analyzability, facilitating rapid iteration to achieve desired controller behaviors. However, it often constrains the controller's versatility and adaptability due to excessive reliance on assumptions and overly-simplified models.

On the other hand, RL-based approaches offer an alternative by discovering optimal control policies through trial-and-error iterations, eliminating the need for manual design choices. RL is particularly beneficial in scenarios where control and planning strategies based on simplifying assumptions are impractical [2, 3, 4]. However, the standard RL setup requires hand-crafted reward functions, which impede the induction of user-desired behaviors of a policy.

In this letter, we introduce a control framework that combines MOC and RL for dynamic and robust legged locomotion. In creating this control framework, our main goal is to determine how to complement the individual strengths of MOC and RL while circumventing their intrinsic limitations. To this end, we formulate the motion imitation problem through RL by imitating on-demand reference motions generated using an optimal control framework [5, 6].

The authors are with the Computational Robotics Lab in the Institute for Intelligent Interactive Systems (IIIS), ETH Zurich, Switzerland. {kangd, jicheng, mimora, fzargarbashi, scoros}@ethz.ch

We thank Zijun Hui for his assistance with the robot experiments.

This work has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 866480).

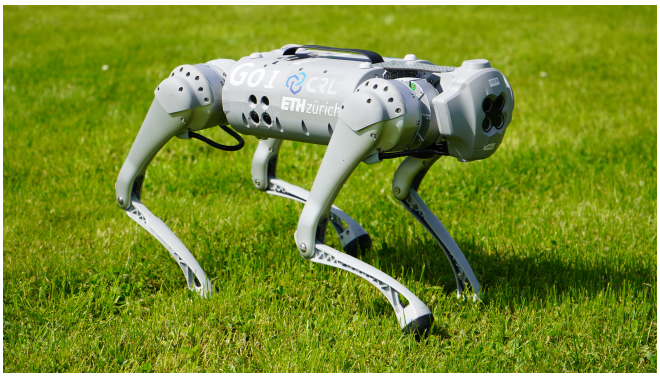

Fig. 1: The quadruped robot *Unitree Go1* executing the target gait pattern *trot* while walking on a grass field.

These reference motions, encompassing various gaits and target velocities, enables the training of a policy capable of generating diverse gait patterns and responding to arbitrary velocity commands.

Our extensive evaluations demonstrate that the learning process within our framework effectively preserves gait styles, facilitates seamless integration of user-defined motion parameters, and requires minimal reward shaping or additional engineering interventions. By tightly integrating MOC into the learning process, the reliability of RL training is significantly improved, as it no longer needs to discover motion skills from scratch. Furthermore, through a carefully designed imitation reward scheme, RL is able to mitigate the adverse effects of the inevitable modeling simplifications employed in our optimal control problem (OCP). Consequently, the learned policy demonstrates the ability to generalize the MOC demonstrations by considering the overall impact of actions, effectively leveraging rotational dynamics, and adapting to uneven terrain. These capabilities are beyond the scope of the model employed in our OCP.

In summary, we introduce an RL method that incorporates the advantages of the MOC approach, allowing for precise control over the behavior of a policy, while retaining the flexibility of RL to generalize to complex scenarios. We demonstrate the effectiveness of our method across a range of quadrupedal locomotion tasks, highlighting its potential applications in various real-world scenarios.

II. Related Work

In this section, we review the relevant literature in the two domains of legged locomotion control integrated into our method: MOC-based and RL-based legged locomotion.

A. Model-based Optimal Control for Legged Locomotion

The model-based optimal control (MOC) approach entails the formulation of an optimal control problem (OCP) with the goal of finding control signals that minimize an objective function, while respecting the constraints imposed by the system dynamics and physical limitations [1]. A current trend in this domain is to formulate locomotion tasks as long-horizon OCPs that reason the results of the robot's actions. By employing modern trajectory optimization (TO) techniques, this approach enables the ability to generate complex behaviors, such as jumping [7, 8], backflipping [9], and overcoming obstacles [10, 11, 12].

In general, the solution of a long-horizon OCP with highly nonlinear dynamics models demands substantial computational power. Model predictive control (MPC) methods, which optimize control signals in a receding horizon fashion, often utilize simplified dynamics models and relatively short time horizons to overcome this challenge while improving robustness. The inverted pendulum model [13] for legged robots offers a significant reduction in the dimensionality of an OCP and facilitates the seamless integration of foothold optimization within MPC frameworks. With appropriate modifications, this model can capture dynamic motions that involve flying phases and up-and-down body motions [5, 6]. Another widely used model in quadrupedal locomotion is single rigid body model (SRBM) with the ability to capture the rotational motions of the system [14, 15, 16], which allows for highly dynamic maneuvers characterized by significant body rotation around pitch and roll axes [17].

Selecting a model requires careful consideration of the trade-off between fidelity, expressiveness, and computational complexity. Achieving higher levels of realism and expressiveness comes at the cost of increased computational burdens. In our approach, which involves RL training to imitate MOC generated on demand, solving OCP efficiently is crucial to reduce training time. Therefore, we adopt the low-dimensional variable height inverted pendulum model (VHIPM) and the associated OCP formulation proposed by Kang et al. [5, 6]. While the VHIPM may not accurately represent pitch and roll body motions or locomotion on uneven terrains, we demonstrate how our motion imitation framework enables the production of dynamic motions involving body rotation and locomotion on challenging terrains in the following sections.

B. RL-based Legged Locomotion

In recent years, there has been extensive research on utilizing RL for legged locomotion. RL enables the discovery of control policies by maximizing the cumulative discounted reward through trial-and-error, automating a significant portion of the control or planning strategy design process. Notably, RL can learn policies that handle modeling and environmental uncertainties by incorporating randomized training scenarios [18, 19, 20, 21]. However, it should be noted that this approach often requires careful reward shaping to achieve the desired behavior, necessitating the design and balancing of auxiliary rewards to promote smooth actions, energy efficiency, and stable gait patterns. An alternative approach is to adopt a hierarchical control structure that incorporates prior knowledge to address these challenges. Iscen et al. [22] proposed an architecture to learn a high-level RL policy that outputs parameters for the foot trajectory generator and joint signal correction. Following this paradigm, Lee et al. [2], Miki et al. [3] utilized the RL policy to refine foot trajectories and correct footholds achieving great robustness in blind and perceptive locomotion in the wild.

Another RL-based approach is motion imitation, where a policy is trained to replicate reference motions obtained from real-world demonstrations or other controllers. Peng et al. [23] proposed an RL method to train control policies for simulated characters to imitate prerecorded motions from animals or humans. The training process utilizes rewards that incentivize the policy to match motions from a motion clip. This approach has enabled reproducing a wide range of behaviors without requiring excessive reward tuning. In their subsequent work, Peng et al. [24] successfully applied this approach to a quadruped robot using a dataset collected from dogs. Peng et al. [25], Escontrela et al. [26], Li et al. [27] extended this paradigm by incorporating a generative adversarial structure with a discriminator that differentiates between motions generated by a policy and pre-recorded demonstrations.

Our approach utilizes the motion imitation technique introduced by Peng et al. [23], employing a reference motion generator based on MOC as a substitute for prerecorded motion datasets, similar to recent studies by Fuchioka et al. [28], Miller et al. [29]. This allows us to elicit desired behavior from an RL policy effectively without motion retargeting. Unlike Fuchioka et al. [28] focusing on motion imitation of a single pre-generated trajectory nor Miller et al. [29] utilizing a dataset of pre-generated trajectories, our method generates reference motions on-demand during training. Similar works only generate foot trajectories online Shao et al. [30] and Jin et al. [31], while our work incorporates MOC as a motion generator on demand. This enables us to develop a more versatile legged locomotion controller that supports run-time switching of desired gait patterns and body velocities. We demonstrate the ability of our method to produce diverse gait patterns on uneven terrain using a single RL policy.

III. Overview

Our method combines the MOC and RL approaches by training an RL policy that imitates reference robot motions generated from solving a finite-horizon OCP. This integration provides a clear and intuitive means of incorporating design choices rooted in prior knowledge and heuristics into RL training. Additionally, we harness the flexibility of RL to improve the policy's ability to generalize across various challenging scenarios that cannot be accurately addressed solely with the simplifications in MOC.

Our framework, depicted in Fig. 2, develops a control policy that generates joint-level actions based on the robot's current state and high-level user commands, such as the desired body velocity and gait pattern. In each training

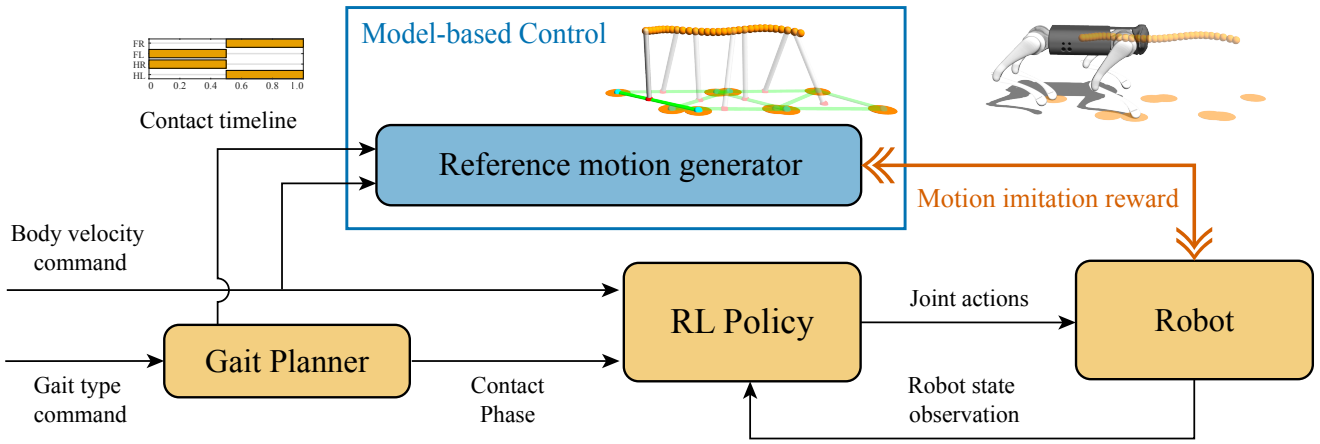

Fig. 2: Overview of our framework. The objective is to train a policy that outputs joint actions to imitate the reference. The reward signal quantifies the similarity between the robot's state and reference motions generated by the MOC-based motion generator.

episode, we randomly select a body velocity command from a range and choose a desired gait pattern from a predefined set. The gait planner generates a contact timeline for each leg based on the desired gait pattern. Subsequently, the reference motion generator produces a sequence of reference motions over a time horizon that follows the body velocity command and the contact timeline. These reference motions are generated on-demand whenever new motions are required and are stored in a queue. We employ a MOC method in this stage to generate dynamically-consistent reference motions using a simplified model. Dynamically-coherent reference motions contribute to more effective RL training, particularly for motions in which dynamic effects play a crucial role, on which further details are discussed in the Section VI.

During the training process, an RL policy learns to map the robot's body velocity command, contact phase variables, and current state observation to joint actions by aligning the robot's state with the corresponding reference motion within a physically simulated environment. Once the policy finishes matching the entire queue of waiting reference motions, a new request for series of reference motions is sent to the motion generator. Upon completing the training process, only the gait planner and the trained RL policy are deployed to the robot hardware.

IV. Reference Motion Synthesis

In the phase of generating reference motions, we solve a finite-horizon OCP based on the variable height inverted pendulum model (VHIPM) [6, 5] to generate trajectories for the robot's base and feet. It is essential to maintain a low computational burden during training, as the reference motion generator is queried on demand. Therefore, we use a simplified dynamics model to minimize training time. However, it is equally important to ensure that the model is capable of accurately capturing dynamic effects. The VHIPM strikes a good balance between model expressiveness and computational complexity.

In formulating the OCP, our objective is to find the system input, which includes footstep locations, a sequence of vertical accelerations, and the position of the center of pressure, that minimizes the discrepancies between the kinematic references of the base and feet motions and the predictions made by the dynamics model.

A. The Variable Height Inverted Pendulum Model

The VHIPM represents a robot as an inverted pendulum extending from its center of pressure (CoP) to its center of mass (CoM). The CoP of a robot, denoted by $\mathbf{x}_{\text{cg}} \in \mathbb{R}^3$, is determined by a convex combination of the stance foot positions $\mathbf{s}^i \in \mathbb{R}^3$, which collectively define a support polygon represented by a set σ:

$$\mathbf{x}_{\text{cg}} = \sum_{\mathbf{s}^i \in \sigma} w^i \, \mathbf{s}^i, \tag{1}$$

where $w^i \in \mathbb{R}_{\geq 0}$ is a non-negative scalar weight corresponding to $\mathbf{s}^i$ that satisfies $\sum_i w^i = 1$.

Using this representation, we can express the linear acceleration of a robot's CoM denoted by $\ddot{\mathbf{r}} \in \mathbb{R}^3$ as a function of the position of the robot's CoM $\mathbf{r} \in \mathbb{R}^3$, vertical acceleration $\ddot{h} \in \mathbb{R}$ and the CoP $\mathbf{x}_{\text{cg}}$ according to the following equation:

$$\begin{aligned} \ddot{\mathbf{r}} &= \boldsymbol{f}_{\text{VHIPM}}(\mathbf{r}, \mathbf{u}, \sigma) \\ &\coloneqq (\mathbf{r} - \mathbf{x}_{\text{cg}}) \frac{\ddot{h} + \|\mathbf{g}\|_2}{r_z} + \mathbf{g} \\ &= (\mathbf{r} - \sum_{\mathbf{s}^i \in \sigma} w^i \, \mathbf{s}^i) \frac{\ddot{h} + \|\mathbf{g}\|_2}{r_z} + \mathbf{g}. \end{aligned} \tag{2}$$

Here, we define $\mathbf{u} \coloneqq \begin{bmatrix} \ddot{h} \; w^1 \; w^2 \; \dots \; w^{\|\sigma\|} \end{bmatrix}^\top$ as the control input vector. For a visual representation of the VHIPM, please refer to Fig. 3. A detailed derivation of the equations of motion can be found in our previous work [5, 6].

It is important to highlight that the VHIPM differs from its predecessor, the linear inverted pendulum model, by considering the robot's base height as a dynamic variable rather than a constant. This allows the VHIPM to effectively capture frequent and continuous flying phases, resulting in substantial up-and-down body movements. As a result, the

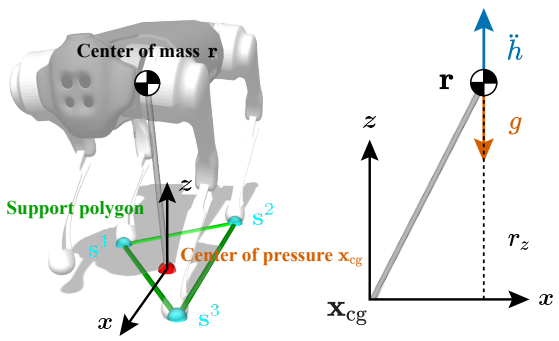

Fig. 3: Quadrupedal robot *Unitree Go1* represented as a variable-height inverted pendulum (**left**) and a diagram of its xz-plane projection (**right**). The VHIPM expresses lateral and vertical acceleration of a robot's CoM as a function of the position of the robot's CoM and the position of CoP.

VHIPM plays a crucial role in generating dynamically-coherent reference motions for dynamic gait patterns.

B. Finite-horizon Optimal Control

In this section, we present a concise overview of our approach to formulating and solving a finite-horizon OCP using the discretized version of the equation (2).

To discretize the VHIPM, we employ a semi-implicit Euler method, resulting in the following formulation:

$$\begin{aligned}\mathbf{r}_{k+1} &\approx 2\mathbf{r}_k - \mathbf{r}_{k-1} + \ddot{\mathbf{r}}_k \Delta t^2 \\ &= 2\mathbf{r}_k - \mathbf{r}_{k-1} + \boldsymbol{f}_{\text{VHIPM}}(\mathbf{r}_k,\, \mathbf{u}_k,\, \sigma_k)\Delta t^2 \\ &=: \mathbf{g}_{\text{VHIPM},k}\left(\mathbf{r}_{k-1},\, \mathbf{r}_k,\, \mathbf{u}_k,\, \sigma_k\right)\,, \end{aligned} \tag{3}$$

where the variables at time $t + k\,\Delta t$, $k \in \{0,\, 1,\, \ldots,\, N_{\text{T}}\}$ are denoted using the subscript k. Here, t is the current time, $\Delta t \in \mathbb{R}_{>0}$ is the time step duration and N_{T} is the time horizon.

We define the stacked state $\mathbf{X} = [\mathbf{r}_1\ \mathbf{r}_2\ \ldots\ \mathbf{r}_{N_{\text{T}}}]^\top$ and the stacked control signal $\mathbf{U} = [\mathbf{u}_0\ \mathbf{u}_2\ \ldots\ \mathbf{u}_{N_T-1}\ \mathbf{s}]^\top$, where $\mathbf{s} \coloneqq \left[\mathbf{s}^1; \mathbf{s}^2; \ldots; \mathbf{s}^{N_{\text{f}}}\right]^\top$ represents the foothold vector sorted into footfall order. Here, N_{f} is the total number of footsteps over the time horizon N_{T}.

Our goal is to find an optimal system input $\mathbf{U}^\star$ that minimizes the cost function,

$$\mathbf{U}^\star = \arg\min_{\mathbf{U}} \mathcal{J}(\mathbf{X}(\mathbf{U}),\, \mathbf{U})\,, \tag{4}$$

where the stacked state vector $\mathbf{X}(\mathbf{U})$ is obtained through forward integration of (3) given the initial conditions $\mathbf{r}_0$ and $\mathbf{r}_{-1} \coloneqq \mathbf{r}_0 - \mathbf{v}_0\,\Delta t$. The cost function $\mathcal{J}(\mathbf{X}, \mathbf{U})$, as defined in our prior work [6], ensures that the predicted velocity and base height of the robot align with the target values. Additionally, it regularizes the footholds to avert potential kinematic singularities in the robot's legs. Moreover, it applies constraints on the input, specifically that the sum of all w_k^i equals one, and each individual w_k^i is greater than or equal to zero.

It should be noted that the OCP (4) is a non-linear program due to the nonlinearity of the VHIPM model where finding a global minimum of the problem is practically infeasible. Instead, we focus on obtaining a reasonably good local minimum by employing a second-order gradient-based method [32, 5, 6].

V. Motion Imitation with Deep RL

We train control policies using Deep RL to matches the robot's state with the reference base and foot trajectories with Proximal Policy Optimization (PPO) [33]. The training process involves a reward signal that measures the correspondence between the robot's state and the reference motions.

A. Observation and Action Space

The policy observes base height, gravity vector in the body frame, body linear and angular velocity, joint position and velocity, the contact phase variables in the $\sin(\cdot)$ and $\cos(\cdot)$ form, body velocity command and previous actions. We use the contact phase parameterization proposed by Shao et al. [30] for each leg, which represents a swing phase as a value in $[-\pi, 0)$ and a stance phase as a value in $[0, \pi)$. This parameterization allows us to train a single policy producing diverse quadrupedal gait patterns as we demonstrate in Section VI-B.

The action of the control policy is defined as joint target positions, which a PD controller will later take to produce target joint torques.

B. Reward Definition

The reward function is defined as a product of the imitation reward r^I and the regularizer r^R, namely

$$r = r^I \cdot r^R. \tag{5}$$

To encourage the policy to minimize the error between the current robot state and the reference motion, the imitation reward r^I is defined as

$$r^I = r^h \cdot r^v \cdot r^{\dot{\psi}} \cdot r^{ee}, \tag{6}$$

where r^h minimizes the error in base height $h \in \mathbb{R}$, r^v in base velocity $\mathbf{v} \in \mathbb{R}^3$, $r^{\dot{\psi}}$ in yaw rate $\dot{\psi} \in \mathbb{R}$ and r^{ee} in feet positions $\mathbf{P}_{ee} \in \mathbb{R}^{4\times 3}$. Notably, the feet positions P_{ee} are defined in the coordinate frame where the origin corresponds to the base position projected to the ground, and the orientation only retains the yaw angle of the base. This choice is made because any disturbances could potentially lead to deviation from the reference motion at a global position level.

Each reward maps the error into a scalar through the following function

$$r^x = \exp\left(-\left\|\frac{\hat{x} - x}{\sigma^x}\right\|^2\right), \tag{7}$$

where $\hat{x}$ and x are the value from the reference and the policy respectively, $\|\cdot\|$ is the proper norm function according to the dimension of x, σ^x is the sensitivity of the mapping and can either be a scalar or in the same dimension as x.

The regularizer r^R is defined as $r^R = r^{\Delta a} \cdot r^{\text{SLIP}} \cdot r^{\phi,\theta}$, where $r^{\Delta a}$ regulates the action rate, r^{SLIP} minimizes the contact feet velocity, and $r^{\phi,\theta}$ maintains the pitch and roll angles close to zero, thereby keeping the robot's base parallel

TABLE I: Reward hyperparameters

Reward terms r^x	Sensitivity σ^x
Base height r^h	0.05
Base velocity* r^v	[0.3, 0.1, 0.3]
Base yaw rate $r^{\dot{\psi}}$	0.5
Feet position* r^{ee}	[0.15, 0.025, 0.15]
Action rate $r^{\Delta a}$	1.5
Feet slip r^{SLIP}	0.1
Pitch and roll $r^{\phi,\theta}$	0.5

* Non-scalar values are applied in forward-vertical-sideways order.

TABLE II: PPO hyperparameters

Batch size	512	Discount factor	0.95
Number of epochs	10	Learning rate	5×10^{-5}
Value function coeff.	0.5	Episode length	128
Entropy coeff.	0.01	Initial std. deviation	$\exp(-1)$

We use five random seeds of 0, 1, 10, 42, 1234 for our experiments.

to the ground plane. The reward function transforms the value into a scalar using a similar exponential function as Equation (7). The sensitivity values of these rewards can be found in Table I.

At the start of each episode, we initialize the robot's pose using a reference motion selected randomly from the reference motion queue similar to Peng et al. [23]. The episode is terminated if the robot collapses. Implementing these measures has proved to be a successful way of efficiently learning dynamic gait patterns, especially those that include flying phases where the robot's feet are temporarily off the ground.

VI. Results

We demonstrate our method using both simulation and hardware experiments on the quadruped robot *Unitree Go1*. All of the policies used in the experiments are trained on simulation data generated by an in-house tool based on the Open Dynamics Engine (ODE) [34]. We use the PPO implementation of an open-source RL software framework, stable-baseline3 [35] with hyperparameters in Table II.

During training, we randomly sample 3-dimensional linear and angular impulses within the ranges of $[-1.5, 1.5]\,\mathrm{m/s}$ and $[-1.5, 1.5]\,\mathrm{rad/s}$ per dimension in the middle of each episode. The perturbation simulation allows us to train an RL policy capable of resisting disturbances and facilitating the transfer of the policy from simulation to the real world. In addition, we simulate a $30\,\mathrm{ms}$ actuator latency, as identified by Margolis and Agrawal [36], to help bridge the sim-to-real gap. All RL policies used in the experiments are structured with multi-layer perceptron architecture with two fully-connected hidden layers of 256 units and the ELU activation function. The policies are queried at a rate of $50\,\mathrm{Hz}$. The target values of base height and foot swing height are set to $0.32\,\mathrm{m}$ and $0.08\,\mathrm{m}$ respectively.

A. Simulation Experiments

In the subsequent simulation experiments, we evaluated the performance of the policy trained with our method in generating desired gait patterns, resilience against external disturbances, and traversal of challenging terrains.

TABLE III: The gait parameters and experiment results of control policies. A visual depiction of the gait patterns are provided in Fig. 7.

	Trot	Pace	Pronk	Bound	Gallop
Duration	0.5 s	0.5 s	0.4 s	0.4 s	0.5 s
Duty cycle	0.5	0.6	0.6	0.6	0.45
Phase Offsets	0.5 0.5 0	0.5 0 0.5	0 0 0	0 0.5 0.5	0.75 0.5 0.25
MPC	○	○	○	△	△
Baseline (Ours)	○	○	×	○	○
Ours	○	○	○	○	○

○ indicates a success over the entire range of commanded velocity, △ indicates instability or bad command tracking performance in some range of velocity command, and × indicates a failure to produce the target gait motion. The duty cycle of a gait pattern is a proportion of the duration of the contact phase to a gait duration, and phase offsets are relative offsets from the phase of the front-left leg of the front-right, hind-left, and hind-right legs respectively.

In the first experiment, we assessed the capability of our policy to produce quadrupedal gaits with parameters in Table III. We compared our approach to an MPC method proposed by Kang et al. [5, 6] which finds a finite-horizon optimal control signal with the same simplified model and OCP formulation for generating reference motions. To convert the MPC trajectory into joint-level control signals, we combine the MPC with a whole-body control method as proposed in the previous literature [6].

Additionally, we included a baseline policy trained using our motion imitation method with kinematic reference motions generated by numerical integration and a simple foothold planning rule [6, eq. (2) and (3)], instead of the OCP solution. We introduce this policy to emphasize the importance of accounting for dynamic effects in reference motion generation when learning dynamic maneuvers.

We evaluated each policy with forward velocity commands ranging from $-0.5\,\mathrm{m/s}$ to $1.0\,\mathrm{m/s}$ and observe their overall behavior. As described in Table III, the MPC failed to produce *bound* and *gallop* motions for high-velocity commands, resulting in falls. MPC's underlying dynamics model cannot capture the rotational dynamics resulting in zero base pitch angle all the time, which we identified as the main reason for the failure in the trajectory profiles depicted in Fig. 4. Rotational dynamics result in a certain level of inevitable fluctuation around the pitch axis, which is crucial for bound motions. Remarkably, our policy is trained to imitate the reference motion generated by the same dynamics model, but the RL training process can generalize the reference motions over the entire body dynamics of the robot. As a result, it learns to produce small rotational movements rather than strictly adhering to the reference motions.

Our baseline policy cannot produce *pronk* motions because *pronk* motion involves a momentary flying phase where the base free-falls. As the baseline policy imitates the reference motion without reflecting such dynamic effect, the policy fails to learn jumping from the ground and converges to the behavior of skating on the ground, as shown in the foot height plot. This highlights the importance of addressing

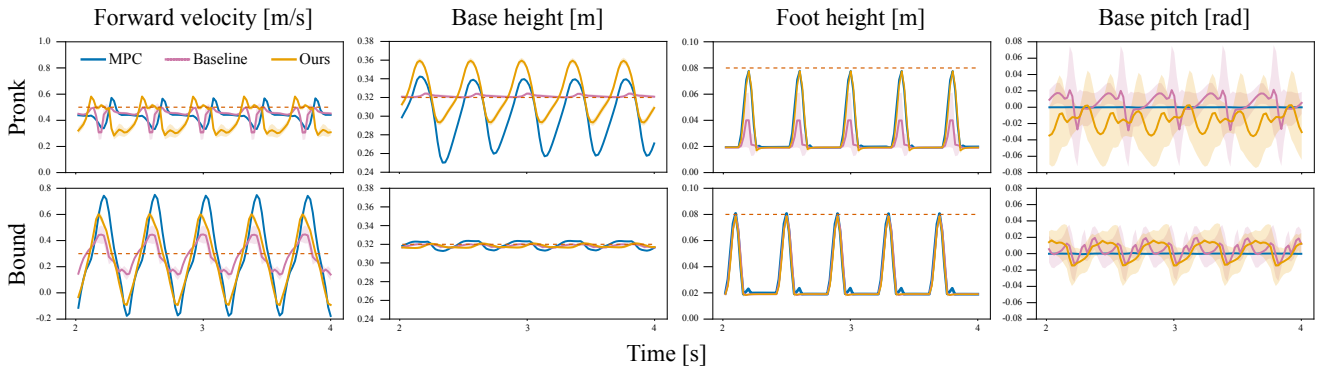

Fig. 4: To observe behaviors of MPC (in **blue**), Baseline (in **purple**) and Ours (in **yellow**) in more detail, we plot the profile of forward velocity, base height, and front-left feet trajectory for commanded velocity $0.5\,\mathrm{m/s}$ for pronk (**first row**) and $0.3\,\mathrm{m/s}$ for bound (**second row**) respectively. The colored lines are the mean of each quantity obtained from five trainings with different seeds and the shaded areas are the corresponding standard deviations. The red dotted lines stand for command velocity and motion parameters.

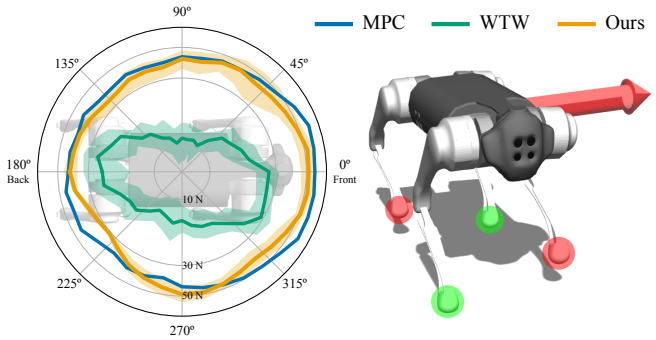

Fig. 5: Snapshot of perturbation test (**right**) and the maximum pushing force along different directions that each policy withstands (**left**). The radial axis of the plot is in log scale.

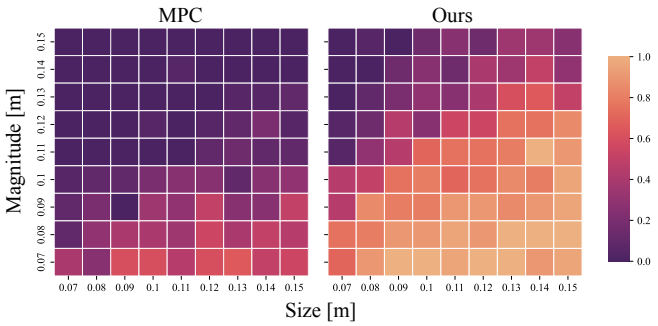

Fig. 6: The success rate of the step terrain locomotion test of the MPC (**left**) and ours (**right**) with respect to different step terrain parameters. A brighter color indicates a higher success rate.

the dynamic effect in generating reference motions for our motion imitation to generalize the policy over more dynamic and agile target motions.

In the second experiment, we assess the robustness of our policies in rejecting external perturbation by appling a pushing force to the robot's base in a specific direction for $1\,\mathrm{s}$ while the robot *trots* in place. In the assessment, we provide a comparison between our method with the MPC and Walk These Ways (WTW), an RL control method introduced by Margolis and Agrawal [36] in the assessment[1]. WTW employs a reward function that combines multiple sub-reward terms designed based on heuristics. Similar to our method, this method allows the specification of gait parameters and can produce all the gait patterns in Table III. It should be noted that WTW used a distinct definition of observation space in their paper and trained a policy to estimate the state of a robot as well. On the other hand, we utilize a Kalman filter-based state estimator [9] for reliable estimation of base height and velocity in our hardware test. Hence, in the following benchmark results, we assumed that both methods have access to these quantities and used an identical definition of the observation space.

Fig. 5 shows the maximum force magnitude along different directions that our policy, WTW policy, and the MPC can withstand without collapsing. In this experiment, our policy demonstrates superior performance to the WTW policy and achieves similar results as the MPC in resisting pushing forces while maintaining the *trot* motion style. This outcome can be attributed to the robustness of our training process, which effectively preserves the walking motion styles even in the presence of random impulse perturbations. We note that we could not apply the same range of impulse perturbation to the WTW policy during the training as it cannot preserve the style of motions and converges to behaviors of crouching or dragging the robot's legs.

In the final simulation experiment, we demonstrate the generalization capability of our method in handling locomotion on rough terrains, overcoming the limitations of the VHIPM in dealing with non-flat surfaces. We introduce simple modifications by using terrain measurements to adjust the reference trajectories for the base and feet, aligning them with the vertical variations of the terrain. Additionally, we include a height scan of 7×11 measurements around the base as an observation, similar to [20]. This observation allows the robot to anticipate obstacles and react proactively, preventing conservative behaviors such as stepping in place blindly.

For the experiment, an RL policy is trained on terrains consisting of square steps of random height sampled from $[0, 0.15]\mathrm{m}$. The size of the square steps within an episode is uniformly sampled from $[0.07, 0.15]\mathrm{m}$. To evaluate the performance of the RL policy, we compare it with the

[1]To ensure a fair comparison, it is essential to match the foot contact timings for policies. We observed that the WTW policy does not always align with the commanded contact timeline. Hence, we trained the WTW policy with a *trot* command as described in Table III, identified the resulting duty cycle of 0.3, then trained our policy with the same duty factor. We also used the same duty factor for the MPC.

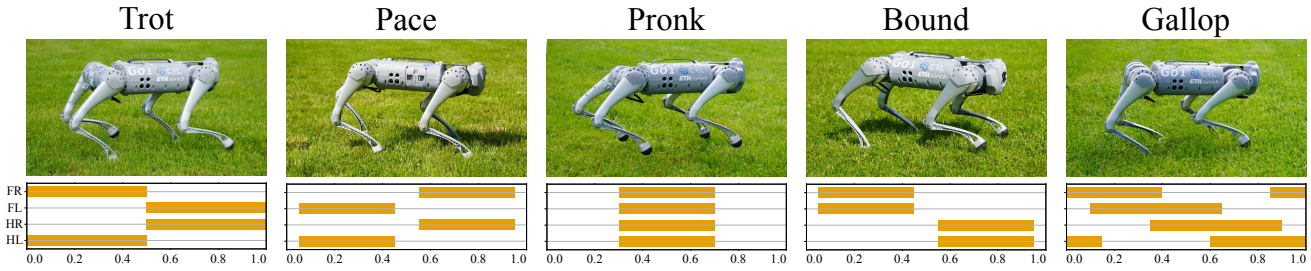

Fig. 7: The snapshots of the *Go1* robot (**top**) performing five different gait patterns that are graphically described in Hildebrand-style gait diagrams (**bottom**).

Fig. 8: The snapshots from the robustness tests conducted on the robot hardware. Our policy successfully withstands of external pushes (**left**) and navigates through terrain roughness (**right**).

modified MPC that considers the height offset of the terrain when generating footholds.

The heatmaps presented in Fig. 6 illustrate the success rate of 20 trials conducted on randomly generated terrains, comparing the performance of the RL policy and the MPC. The failure cases encompass collisions between the base and the ground, as well as failure to reach a boundary of $5\,\mathrm{m}$ within $10\,\mathrm{s}$. Notably, our controller significantly outperforms the MPC. While the MPC struggles to navigate the terrains due to limitations in its model, our policy successfully generalizes the reference motions generated using the same model and exhibits enhanced adaptability to locomotion on rough terrains. The supplementary video contains the recorded footage of the experiment.

B. Hardware Experiments

In hardware experiments, we validated our approach by training an RL policy to generate all the gait patterns described in Table III and deploying it onto the *Unitree Go1* hardware platform. The policy was trained using a total of $16{,}384{,}000$ data samples.

To ensure a reliable sim-to-real transfer, we incorporated additional measures during the training phase. Firstly, we introduced randomization of the friction coefficient of the terrain, ranging from 0.5 to 1.25. This variation in friction helps the policy adapt to different surface conditions and enhances its robustness. Secondly, for each episode, we generate a random uneven terrain using Perlin noise [2] with a frequency range of $[0, 0.9]$ and a magnitude range of $[0, 0.1]\,\mathrm{m}$. This terrain variability allowed the policy to learn to navigate through rough surfaces and further improve its adaptability. Thirdly, we simulated terrain inclination by randomly sampling the gravity vector within a cone of a half-angle of $10\,\mathrm{deg}$ in the physics simulation. This enabled the policy to handle uneven terrains with varying slopes and further expanded its capabilities. Finally, during training, we applied the same random impulses used for the perturbation test. This measure is crucial not only for enhancing the policy's resilience against external pushes but for achieving an effective sim-to-real transfer. The rationale lies in the following observations: as the policy converges, the robot's motions align closely with the generated reference motions. Consequently, the policy becomes more susceptible to perturbations, or potential deviations from other factors contributing to the sim-to-real gap. By applying impulses, we can generate a more diverse set of samples that assist the policy in effectively addressing the sim-to-real gap.

Our policy successfully produces diverse gait patterns while effectively withstanding external pushes and navigating rough and inclined terrains. We highlight that this robust and versatile behavior was achieved without excessive reward shaping or additional engineering. We provide snapshots of the hardware experiments in Fig. 7 and Fig. 8. For a more comprehensive view, we invite readers to watch the footage of the experiments available in our supplementary video.

VII. Conclusion and Future Work

This letter introduces a legged locomotion control method capable of generating diverse gait patterns while maintaining robustness against perturbations and uneven terrains. Our approach involves imitating dynamically-coherent reference motion generated using MOC with a simplified dynamics model. We further generalize these reference motions to handle more complex locomotion tasks that may challenge the simplified model, leveraging the flexibility of Reinforcement Learning. Our method offers an intuitive way to incorporate prior knowledge into the RL framework, enabling the learning of versatile policies that exhibit desired behaviors without excessive reward shaping or additional engineering.

To reduce the computational burden, we employ a simple VHIPM for reference motion generation. Remarkably, we have successfully extended the application of the VHIPM to tackle more complex locomotion scenarios. Nonetheless, our approach can potentially incorporate more sophisticated dynamics models, such as the SRBM [28, 29], which can handle more intricate skills. However, it is important to emphasize the significance of carefully considering the trade-off between strictly imitating behaviors derived from MOC and allowing flexibility within RL to generalize motions. This trade-off will be a key aspect of our future investigations.

We are interested in extending our approach to generate a wider range of legged robot behaviors, unrestricted by the constraints of periodic or symmetric gait patterns [37, 6]. The contact phase parameterization [30] we have adopted

is not limited to representing periodic or symmetric gaits. Therefore, we aim to investigate the applicability of our control method in generating more animal-like behaviors on legged robots, building upon our prior work on animal-inspired locomotion [37, 6].

REFERENCES

[1] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. D. Prete, "Optimization-based control for dynamic legged robots," 2022.

[2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, 2020.

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

[4] S. Choi, G. Ji, J. Park, H. Kim, J. Mun, J. H. Lee, and J. Hwangbo, "Learning quadrupedal locomotion on deformable terrain," *Science Robotics*, vol. 8, no. 74, p. eade2256, 2023.

[5] D. Kang, F. De Vincenti, and S. Coros, "Nonlinear model predictive control for quadrupedal locomotion using second-order sensitivity analysis," 2022.

[6] D. Kang, F. De Vincenti, N. C. Adami, and S. Coros, "Animal motions on legged robots using nonlinear model predictive control," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 955–11 962.

[7] Y. Ding, C. Li, and H.-W. Park, "Kinodynamic motion planning for multi-legged robot jumping via mixed-integer convex program," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 3998–4005.

[8] M. Chignoli and S. Kim, "Online trajectory optimization for dynamic aerial motions of a quadruped robot," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7693–7699.

[9] B. Katz, J. D. Carlo, and S. Kim, "Mini Cheetah: A Platform for Pushing the Limits of Dynamic Quadruped Control," in *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, QC, Canada: IEEE, May 2019, pp. 6295–6301.

[10] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and Trajectory Optimization for Legged Systems Through Phase-Based End-Effector Parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, Jul. 2018.

[11] C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1635–1648, 2020.

[12] M. Geilinger, S. Winberg, and S. Coros, "A computational framework for designing skilled legged-wheeled robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3674–3681, 2020.

[13] S. Xin, R. Orsolino, and N. Tsagarakis, "Online relative footstep optimization for legged robots dynamic walking using discrete-time model predictive control," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 513–520.

[14] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid: IEEE, Oct. 2018, pp. 1–9.

[15] G. Bledt, P. M. Wensing, and S. Kim, "Policy-regularized model predictive control to stabilize diverse quadrupedal gaits for the MIT cheetah," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vancouver, BC: IEEE, Sep. 2017, pp. 4102–4109.

[16] D. Kim, J. D. Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," 2019.

[17] Y. Ding, A. Pandala, and H.-W. Park, "Real-time Model Predictive Control for Versatile Dynamic Motions in Quadrupedal Robots," in *2019 International Conference on Robotics and Automation (ICRA)*. Montreal, QC, Canada: IEEE, May 2019, pp. 8484–8490.

[18] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

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

[20] 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*. PMLR, 2022, pp. 91–100.

[21] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," 2022.

[22] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Conference on Robot Learning*. PMLR, 2018, pp. 916–926.

[23] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[24] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems*, 07 2020.

[25] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–20, 2021.

[26] 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," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 25–32.

[27] C. Li, S. Blaes, P. Kolev, M. Vlastelica, J. Frey, and G. Martius, "Versatile skill control via self-supervised adversarial imitation of unlabeled mixed motions," *arXiv preprint arXiv:2209.07899*, 2022.

[28] Y. Fuchioka, Z. Xie, and M. van de Panne, "Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," 2023.

[29] A. Miller, S. Fahmi, M. Chignoli, and S. Kim, "Reinforcement learning for legged robots: Motion imitation from model-based optimal control," 2023.

[30] Y. Shao, Y. Jin, X. Liu, W. He, H. Wang, and W. Yang, "Learning free gait transition for quadruped robots via phase-guided controller," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1230–1237, 2022.

[31] Y. Jin, X. Liu, Y. Shao, H. Wang, and W. Yang, "High-speed quadrupedal locomotion by imitation-relaxation reinforcement learning," *Nature Machine Intelligence*, pp. 1–11, 2022.

[32] S. Zimmermann, R. Poranne, and S. Coros, "Optimal control via second order sensitivity analysis," 2019.

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

[34] R. Smith *et al.*, "Open dynamics engine," 2007.

[35] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.

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

[37] D. Kang, S. Zimmermann, and S. Coros, "Animal gaits on quadrupedal robots using motion matching and model-based control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8500–8507.