2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Online Optimization of Central Pattern Generators for Quadruped Locomotion

Zewei Zhang, Guillaume Bellegarda, Milad Shafiee, Auke Ijspeert

***Abstract*— Typical legged locomotion controllers are designed or trained offline. This is in contrast to many animals, which are able to locomote at birth, and rapidly improve their locomotion skills with few real-world interactions. Such motor control is possible through oscillatory neural networks located in the spinal cord of vertebrates, known as Central Pattern Generators (CPGs). Models of the CPG have been widely used to generate locomotion skills in robotics, but can require extensive hand-tuning or offline optimization of inter-connected parameters with genetic algorithms. In this paper, we present a framework for the *online* optimization of the CPG parameters through Bayesian Optimization. We show that our framework can rapidly optimize and adapt to varying velocity commands and changes in the terrain, for example to varying coefficients of friction, terrain slope angles, and added mass payloads placed on the robot. We study the effects of sensory feedback on the CPG, and find that both force feedback in the phase equations, as well as posture control (Virtual Model Control) are both beneficial for robot stability and energy efficiency. In hardware experiments on the Unitree Go1, we show rapid optimization (in under 3 minutes) and adaptation of energy-efficient gaits to varying target velocities in a variety of scenarios: varying coefficients of friction, added payloads up to 15 kg, and variable slopes up to 10 degrees.**

I. Introduction

Animals display impressive abilities to swiftly refine their locomotion capabilities. Notably, newborn animals like baby goats or giraffes exhibit an innate ability to walk almost immediately post-birth, a phenomenon largely attributed to the role of Central Pattern Generators (CPGs) [1], [2]. These neural networks, embedded within the spinal cord, facilitate rhythmic movements, forming the biological foundation of their rapid locomotor improvements [3].

Biology-inspired control based on modeling the CPG has been extensively utilized to facilitate various types of robot locomotion. Such approaches have proven effective in controlling a wide range of systems [4]–[6]. CPG-based control is characterized by stable limit cycle behavior, a limited number of control parameters, and simplicity in the integration of sensory feedback [7]. To optimize the control performance, it is essential to optimize for the best CPG parameters. This is usually achieved through manual tuning, evolutionary algorithms (such as genetic algorithms and particle swarm optimization) [8]–[11], and more recently with deep reinforcement learning [12]–[16].

A recent surge in the utilization of Bayesian Optimization (BO) has opened new avenues for online robotic controller tuning. Through efficient interaction with the environment, BO is capable of tuning parameters at an exceptionally rapid pace. For example, Calandra et al. demonstrate the sample efficiency of BO compared to other methods on bipedal locomotion tasks [17], [18]. Concurrently, Rai et al. focus on integrating simulation data into BO to enhance sample efficiency while learning bipedal locomotion in real-world settings [19], [20]. Moreover, Widmer et al. leverage Contextual BO for fine-tuning MPC gains and gait parameters for trot and crawl gaits, all while adhering to safety constraints [21]. Our previous work applies BO for optimizing force profiles for online optimization of quadruped jumping [22].

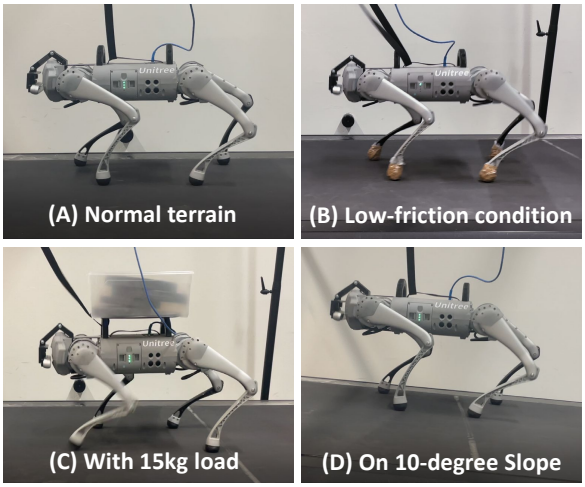

Fig. 1: Optimized CPG-based locomotion tracking of v_x^*=0.4 m/s, rapidly adapting to various conditions, including normal flat terrain, slippery terrain (i.e. taped feet), a 15 kg payload, and a 10-degree slope.

Despite the existing success with BO, it is widely acknowledged that BO has better efficacy and performance when limiting the number of parameters that need to be tuned. Consequently, it is reasonable to anticipate good performance from using BO to tune a CPG-based controller, which has a small number of parameters [7]. In line with this, Ruppert and Sproewitz successfully matched the CPG parameters and passive compliant dynamics, thereby enhancing locomotion energy efficiency within one hour in simulation [23].

Building on these developments, we aim to explore the benefits of BO in the context of rapidly optimizing quadruped locomotion directly on hardware in a variety of scenarios that necessitate adaptation. In contrast to time-intensive optimization methods, we utilize BO to learn locomotion online using CPG-based control in both simulation and real-world environments, all within a few minutes of physical interaction. Furthermore, we utilize Contextual Bayesian Optimization

This research is supported by the Swiss National Science Foundation (SNSF) as part of project No.197237. The authors are with the BioRobotics Laboratory, Ecole Polytechnique Federale de Lausanne (EPFL). {firstname.lastname}@epfl.ch

for challenging tasks to continuously learn new parameters in changing environments. Our main contributions include:

- Rapid online learning of CPG parameters to locomote at varying desired velocities across terrains with varying friction, and swift adaptation to constantly changing target velocities.
- Continuous adaptation to changing slopes and added payloads, incorporating corresponding contextual information through Bayesian Optimization.
- A comparison of performance across various CPG controllers, including those with and without force feedback, and with and without Virtual Model Control (VMC), in the context of locomotion optimization, showing the benefits of both force feedback and VMC for energy-efficient and stable gaits.

The rest of this paper is organized as follows. Section II reviews background on abstract oscillators (to represent Central Pattern Generators), Virtual Model Control, Bayesian Optimization, and Contextual Bayesian Optimization. Section III presents the comprehensive optimization framework including implementation details. Section IV evaluates the performance of online adaptation under various conditions in both simulation and hardware experiments, and presents the comparison results of different CPG controllers. Finally, a brief conclusion is given in Section V.

II. Background

A. Central Pattern Generators

Abstract oscillators as Central Pattern Generators (CPG) are a bio-inspired approach used to generate oscillatory signals. With the coupling of abstract oscillators, synchronized behavior with a locked phase difference can produce different gaits. For quadruped robots, each leg is often represented with a single oscillator, which is coupled with other legs/oscillators.

1) Open-Loop CPG: We first consider an open-loop CPG capable of generating rhythmic motions without relying on external feedback, previously used in [12]. This model has four coupled oscillators, where the equations for the amplitude and phase of oscillator i are:

$$\dot{r}_i = \alpha(\mu^2 - r_i^2)r_i \tag{1}$$

$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}), \tag{2}$$

where r_i is the amplitude, θ_i is the phase, μ and ω_i are the intrinsic amplitude and frequency. α is the convergence rate, w_{ij} is the coupling weight between oscillators, and ϕ_{ij} is the phase lag between oscillators i and j. The values of the intrinsic frequencies ω_i are set based on the leg phase (i.e. swing or stance phase), and we use a trot gait coupling matrix Φ for the phase lags:

$$\omega_i = \begin{cases} \omega_{swing} & \text{if } \sin\theta_i > 0 \\ \omega_{stance} & \text{if } \sin\theta_i \leq 0 \end{cases}; \quad \Phi = \begin{bmatrix} 0 & \pi & \pi & 0 \\ -\pi & 0 & 0 & -\pi \\ -\pi & 0 & 0 & -\pi \\ 0 & \pi & \pi & 0 \end{bmatrix} \tag{3}$$

where the leg order is: Front Right, Front Left, Rear Right, Rear Left. The oscillator states are mapped to foot trajectories (and then mapped to joint commands with inverse kinematics) with the following equations:

$$x_{i,foot} = x_{i,offset} - d_{step} r_i \cos\theta_i, \tag{4}$$

$$z_{i,foot} = \begin{cases} -h + g_c \sin\theta_i & \text{if } \sin\theta_i > 0 \\ -h + g_p \sin\theta_i & \text{otherwise,} \end{cases} \tag{5}$$

where d_{step}, h, g_c, g_p are the desired foot step length, robot height, ground clearance, and ground penetration. The offset distance of the foot trajectory in the x direction is denoted as $x_{i,offset}$, and varies depending on the foot index i, with the front legs ($i = 1,2$), and hind legs ($i = 3,4$).

2) Force feedback (Tegotae) in CPG: Physical interaction can play a significant role in inter-limb coordination during leg movements. For example, decoupled oscillators with local force feedback can produce variable quadrupedal gaits at varying speeds [24]. To improve locomotion stability, we incorporate local force feedback into Equation 2 as follows:

$$\dot{\theta}_i = \omega_i + \sum_{j=0}^{3} r_j w_{ij} \sin(\theta_j - \theta_i - \phi_{ij}) - \sigma_N N_i \cos\theta_i \tag{6}$$

where N_i is the normal force at foot i, and σ_N represents the coefficient of force feedback. Such feedback slows down the frequency when a limb is loaded, and this makes sure that a limb that carries the load of the robot (i.e. is in stance) is not lifted prematurely (it "waits" until other limbs are taking over the load before switching to swing). As suggested in [25], [26], we anticipate that both oscillator couplings and sensory feedback will enhance gait synchronization and robustness.

B. Virtual Model Control

Virtual Model Control (VMC) has previously been used for enhancing the capabilities of quadruped robots, especially for stability on challenging terrains [27], [28]. It is a simple and effective approach for posture control. Augmenting the CPG with VMC helps to correct the contact forces based on the current robot attitude and heading. We use the attitude and direction controllers introduced in [27] to control the orientation (roll, pitch, yaw) of the trunk.

C. Bayesian Optimization

Bayesian optimization (BO) is a sequential model-based optimization approach used to find optimal parameters through fitting the unknown model with sampling [29]. This sample-efficient optimization approach can compute the delegate model based on all input-output pairs obtained from strategically sampling the environment. The general form of the Bayesian Optimization problem is written as: $\mathbf{x} = \text{argmax}_{\mathbf{x}} \hat{f}(\mathbf{x})$, where $\mathbf{x}$ denotes the parameters to be optimized, and $\hat{f}(\cdot)$ represents the surrogate model of the unknown function based on the sampling.

Several methods can be used for acquiring such a delegate model, for example Gaussian Processes (GP) and Tree Parzen Estimators (TPE). Maximizing the acquisition function determines which set of parameters $\mathbf{x}$ to observe

next. There are many options for the acquisition function such as Upper Confidence Bound (UCB), Expected Improvement (EI), and Probability Improvement (PI). In this work, we use Gaussian Processes to regress all of the samples, and we select UCB as our acquisition function defined as: $a(\mathbf{x};\beta)=\mu(\mathbf{x})+\beta\sigma(\mathbf{x})$. Here, $\mu(\mathbf{x})$, $\sigma(\mathbf{x})$ denote the mean and standard deviation of the Gaussian Process, respectively. β is an explicit parameter designed to balance exploitation and exploration in UCB, and we define it as a decaying parameter. Further details regarding β will be elaborated in Section III-B.

D. Contextual Bayesian Optimization

Contextual Bayesian Optimization (CBO) extends standard Bayesian Optimization (BO) to address multi-task problems occurring in different contexts, as opposed to single-task scenarios [30]. By incorporating additional contextual variables, which typically relate to environmental information, CBO enables the derivation of specific solutions tailored to corresponding environments. CBO and the corresponding UCB acquistion function can be determined by modifying the standard BO formulation:

$$\mathbf{x}=\underset{\mathbf{x}}{\operatorname{argmax}}\ \hat{f}(\mathbf{x},\mathbf{c}), \tag{7}$$

$$a(\mathbf{x};\beta,\mathbf{c})=\mu(\mathbf{x};\mathbf{c})+\beta\sigma(\mathbf{x};\mathbf{c}), \tag{8}$$

where $\mathbf{c}$ denotes the contextual variable that relates to the general environmental conditions. This approach also enhances sample efficiency by leveraging optimization results from nearby contexts [31]. Our framework is grounded in the theoretical framework proposed by Krause and Ong [32], which involves constructing a product of kernels of the optimized and contextual parameters.

III. Optimization Framework

In this section we detail our Contextual Bayesian Optimization framework to tune our CPG-based controller for energy-efficient quadruped locomotion to track varying target velocities in a variety of scenarios. A high-level control diagram is depicted in Figure 2, and we explain all components below.

A. Locomotion Optimization

In each optimization loop, the optimizer selects a new set of CPG parameters based on the velocity command input from the user, estimated contextual information, and the objective value computed from data collected in the previous cycle. We optimize eight parameters introduced in Section II-A during the optimization: $\mathbf{x} = [g_c, g_p, \omega_{swing}, \omega_{stance}, \mu, x_{offset,front}, x_{offset,hind}, \sigma_N]$. The ranges $\mathscr{X}$ for these parameters are specified in the Appendix in Table III, and additional fixed control variables are detailed in Table IV. With the updated parameters, the CPG module cyclically outputs reference joint torques $\boldsymbol{\tau}_{ref}$, computed with inverse kinematics and PD control based on the reference foot trajectory of the CPG with force feedback. The final command torques, $\boldsymbol{\tau}_{cmd}$, is the sum of the reference torques from the CPG $\boldsymbol{\tau}_{ref}$ and the VMC torques $\boldsymbol{\tau}_{vmc}$, which are input to the robot for the locomotion task.

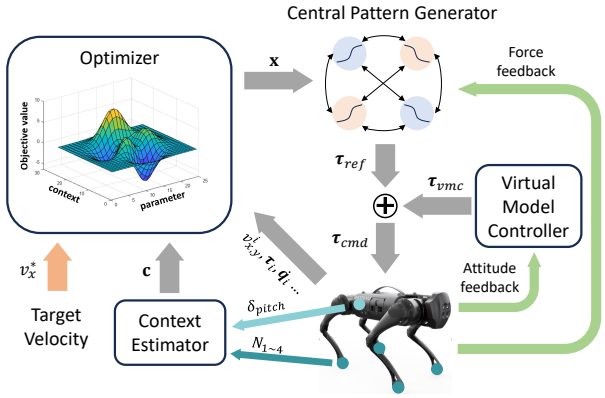

Fig. 2: Control architecture for online CPG optimization. The optimizer updates the CPG parameters $\mathbf{x}$ at the beginning of each trial based on the current target velocity v_x^* and contextual information $\mathbf{c}$. It receives the base velocity and joint data during the steady-state phase of each trial to compute the objective value. Force feedback to the CPG phase equations and Virtual Model Control help to stabilize the robot and promote energy-efficient gaits.

1) Objective Function: We define a simple objective function, and the optimal parameters $\mathbf{x}^*$ will be found to maximize the objective value computed as follows:

$$J=w_1\Delta t\sum_{i=0}^{T}\min\left(\exp\left(-\frac{\|v_{x,y}^i-v_{x,y}^*\|^2}{0.05}\right),l_r\right)-w_2\mathrm{CoT}, \tag{9}$$

where the weights w_1,w_2 are equal to 1 and 0.5 respectively, $v_{x,y}^i$ is the instantaneous body linear velocity, Δt is the time step (1 ms), $T(=3000)$ is the total number of steps in one trial, l_r is the maximum velocity reward. The function consists of two components. The first component is the velocity reward, which increases as the actual velocity comes closer to the target one $v_{x,y}^*$ (=$[v_x^*,0]$). We include the lateral velocity v_y in the function to penalize lateral movement, since some sets of parameters could lead to unexpected oscillations or drift in the lateral direction. The second component aims to lower the energy consumption by penalizing the Cost of Transport (CoT) [33] in each trial, defined as follows:

$$\mathrm{CoT}=\frac{\sum_{i=0}^{T}\langle|\boldsymbol{\tau}_i|,|\dot{\boldsymbol{q}}_i|\rangle\Delta t}{mgd}, \tag{10}$$

where $\boldsymbol{\tau}_i$ and $\dot{\boldsymbol{q}}_i$ denote the vectors of the torque and joint velocity at time i, m is the total mass of the robot and any payload, g is the gravitational constant, and d is the distance covered in one trial. The maximum velocity reward is bounded at $l_r = 0.85$ to expand the range of acceptable actual velocities around the target. This adjustment enables the optimizer to concentrate more on reducing energy consumption when the velocity is close to the target. Through the optimization of this function, we aim to fine-tune the parameters such that the robot achieves locomotion at the target velocity while minimizing energy consumption.

2) Data Reuse: We implement a data reuse technique to enhance the sample efficiency of our optimizer when transitioning between different target velocities. Upon receiving a new target velocity from the user, we leverage the data previously collected during the optimization for the former target. This approach allows the optimizer to recalibrate all prior objective values by merely adjusting the

Algorithm 1: Online optimization for quadruped locomotion learning with CPG using CBO.

Input: $\mathbf{x}_{init} \in \mathscr{X}$, β_{init}, N, v_x^*;
Initialize datasaver of objective values and parameters $\mathbb{S} \leftarrow \emptyset$, $\mathbb{P} \leftarrow \emptyset$;
$i \leftarrow 0$; $\mathbf{x} \leftarrow \mathbf{x}_{init}$; $\beta \leftarrow \beta_{init}$;
while $i \leq N$ **do**
  if v_x^* *is updated* **then**
    $\mathbb{S} \leftarrow$ Update previous objective values with v_x^*;
  end
  $\mathbf{c}$, $J_i \leftarrow$ Update current context and objective value;
  $\mathbb{S} \leftarrow \mathbb{S} \cup J_i$; $\mathbb{P} \leftarrow \mathbb{P} \cup [\mathbf{x}, \mathbf{c}]$;
  $\hat{f} \leftarrow$ Update surrogate model with $\mathbb{S}$, $\mathbb{P}$;
  $\beta \leftarrow$ Update with Equation 12;
  if $i \leq 2$ **then**
    $\mathbf{x} \leftarrow random(\mathbf{x} \in \mathscr{X})$;
  else
    $\mathbf{x} \leftarrow \text{argmax}_{\mathbf{x}} a(\mathbf{x}; \beta, \mathbf{c})$
  end
end

desired velocity value in Equation 9. Such re-utilization of existing data facilitates rapid adaptation to the new velocity, eliminating the necessity to restart the optimization process with each change in velocity target.

3) Context Estimator: As previously mentioned, CBO uses contextual variables to achieve multi-task learning. Therefore, we propose a context estimator to estimate contextual information from the environment. We take into account two contextual variables $\mathbf{c} = [c_{load}, c_{slope}]$ which relate to the load/mass of the robot and the slope information of the terrain. We implement two simple estimators to measure this information during each optimization cycle:

$$c_{load} = \frac{1}{4T} \sum_{i=0}^{T} \sum_{j=0}^{4} N_{i,j} \ , \quad c_{slope} = \frac{1}{T} \sum_{i=0}^{T} \delta_{pitch,i} \tag{11}$$

where c_{load} estimates the average normal force measurement $N_{i,j}$ collected by the force sensors on all four feet, while c_{slope} estimates the average pitch measurements $\delta_{pitch,i}$ obtained from the IMU. Both estimators are used to provide a two-dimensional contextual variable $\mathbf{c}$ to the optimizer throughout the experiment. As demonstrated in Section IV, each estimator is capable of providing consistent contextual information during the optimization process, which is crucial for learning the optimal parameters under varying contexts.

B. Algorithm

Algorithm 1 summarizes the overall optimization procedure with CBO. For the initial two parameter updates, we randomly sample new points. Beyond this, the acquisition function plays a crucial role in updating parameters for subsequent trials. With the UCB acquisition function, we achieve a balance between exploration and exploitation by adjusting the explicit hyperparameter β, as detailed in Equation 8. We employ a time-decaying variable β to foster increased exploration in the early stages of optimization. This approach gradually shifts the emphasis towards exploitation by reducing β's value, thus avoiding overly aggressive sampling. The update of the parameter β is governed by Equation 12:

$$\beta = \beta_{init} \cdot \gamma^{\max(n - n_{decay}, 0)}, \tag{12}$$

where $\beta_{init}, \gamma, n_{decay}, n$ represent the initial value of β, the decay rate, the trial number at which the decay begins, and the total number of trials considered in the current context, respectively. The values of these parameters and details for updating them are presented in the Appendix (Section VI-B).

C. Implementation Details

We designate a short-length horizon for one optimization cycle. Each trial is divided into two stages: the transition phase, which lasts for 1.5 seconds, and the steady-state phase, lasting for 3 seconds. The computation of the objective value and contextual variables is based solely on the data collected during the steady-state phase, in order to minimize the influence of parameter transitions. In the transition phase, we implement a simple linear interpolation to smooth the parameter transition between two trials.

Our optimization framework is implemented in the BoTorch framework [34] and is built upon the *bayesopt4ros* package, developed by Fröhlich et al. [35]. Compared to their framework, we include the decaying exploration factor β in the acquisition function, and apply the data reuse technique explained in Sec. III-A.2 to improve the sample efficiency.

D. Experimental Setup

We test our pipeline on the Unitree Go1 in both simulation and on hardware. For simulations, we use Gazebo with the ODE physics engine. In hardware experiments, we conducted the optimization trials with the robot placed on a treadmill. All proprioceptive sensing measurements are read by the Unitree sensors, namely the IMU, joint states, and foot contact sensors.

IV. Results

In this section, we report results from rapidly optimizing locomotion tasks in both simulation and hardware experiments. We demonstrate our framework's ability to optimize for energy-efficient single velocity tracking tasks, as well as its ability to adapt to changes in the environment in several different scenarios, including: adaptation to varying coefficients of friction, varying velocity commands, varying terrain slopes, and added payloads of up to 15 kg. We also conduct a comparative analysis of the CPG controller's performance with respect to both force feedback and the attitude controller (VMC), and discuss the effectiveness of each controller's corresponding module within our framework. The reader is encouraged to watch the supplementary video for clear visualizations of the discussed experiments.

A. Learning Single Velocity Tracking

In simulation, we define three types of terrains with friction coefficients of 0.3, 0.6, and 0.9, which can correspond to slippery, normal, and coarse terrain. We optimize the parameters of the CPG with both the force feedback and VMC controller to test the learning

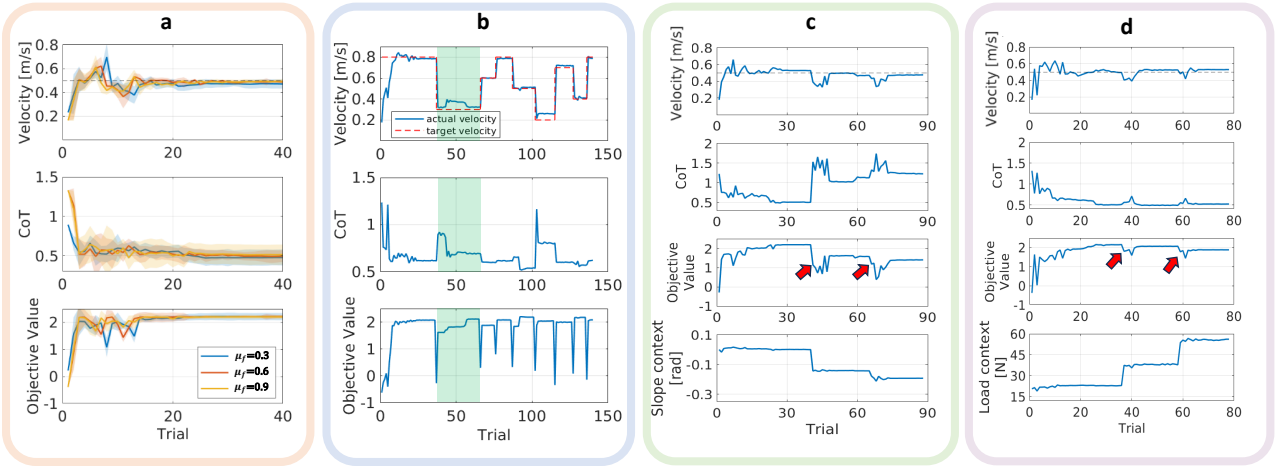

Fig. 3: Simulation results for online locomotion learning, where we report the mean forward velocity, Cost of Transport (CoT), objective value, and context information in different scenarios. **(a)**: adapting to different coefficients of friction while tracking a target velocity of 0.5 m/s. **(b)**: rapid adaptation to moving target velocities. **(c)**: tracking 0.5 m/s with adaptation from flat terrain to varying slopes of 10 and 12.5 degrees. **(d)**: tracking 0.5 m/s with adaptation to added loads of 7.5 kg and 15 kg. Red arrows indicate the transition time to a new contextual condition (slope or load).

performance to track a target forward velocity of 0.5 m/s. We randomly initialize the parameters, which gives a low initial objective value. As depicted in Figure 3a, our pipeline exhibits rapid adaptability to diverse terrains, as evidenced by the average performance obtained from five random seeds. We observe the rapid improvement of the objective value within very few trials, with 40 trials taking **less than 3 minutes**. For all cases, multiple prospective solutions (with high objective value) were identified within fewer than 10 trials (45 seconds). Ultimately, the optimizer consistently outputs a set of parameters that maximizes the objective value. This result indicates the system's capability to track the target velocity and improve energy-efficiency effectively.

B. Multi-Velocity Tracking

Our framework enables the robot to rapidly adapt to different target velocities on a single terrain, leveraging data reuse as discussed in Section III-A.2. Figure 3b shows the robot successfully learns to adapt its gait for consecutively changing target velocities. Initially, the robot learns to walk at a target speed of 0.8 m/s from scratch. Subsequently, when the target velocities were altered to 0.3 m/s and then to 0.6 m/s, the robot required fewer trials to achieve these speeds, benefiting from the knowledge acquired in previous learning phases. Furthermore, the CoT curve between Trials 40 and 60 (green highlight in the plot) illustrates the robot's capability to continuously improve gait energetic efficiency by fine-tuning around the most promising parameters. Within a learning span of approximately 140 trials (about 10 minutes), our framework demonstrates a comprehensive process through which the robot evolves from walking at a single velocity to efficiently tracking multiple velocities.

C. Slope and Load Adaptation

Our framework's adaptability to new environmental conditions, including changes in slope and added load, is further demonstrated with estimated contextual information.

Figure 3c shows the robot's learning process to maintain a target velocity of 0.5 m/s while transitioning from flat terrain to inclines of 10 and 12.5 degrees. The plot of contextual variable c_{slope} confirms the reliability of our context estimator across single terrains, with c_{slope} values inversely correlating with increased slope angles. Notably, upon encountering new terrain types, there is an initial drop in the objective value, followed by subsequent improvements after several optimization cycles.

TABLE I: Mean velocity, CoT, and objective value measured under 0 kg, 7.5 kg, and 15 kg load conditions with the CPG parameters before and after adaptation with our framework. Values in parentheses represent outcomes obtained by directly applying the optimal parameters identified on flat terrain without any load for the target velocity $v_x^* = 0.5$ m/s.

Load [kg]	Velocity [m/s]	CoT	Objective Value
0	0.491 (-)	0.510 (-)	2.175 (-)
7.5	0.524 (0.403)	0.489 (0.564)	2.213 (1.9735)
15	0.528 (0.341)	0.523 (0.905)	2.189 (1.270)

To verify the robustness of the learned parameters, as shown in the video, we then tested our optimized controller on slopes gradually increasing from 10 to 17.5 degrees in inference mode, by fixing β in Equation 8 to 10^{-6} to minimize exploration risk. We observe that although the robot had only optimized parameters to locomote on a maximum slope of 12.5-degrees, it is still able to adapt to slopes of up to 15 degrees, highlighting our framework's capability to efficiently select learned parameters based on the estimated contextual information. It also underscores the potential to select viable parameters for conditions which were not seen duirng the optimization.

For the load adaptation experiment in Figure 3d, we first add a 7.5 kg load (at Trial 36), followed by an additional load to reach a total of 15 kg (at Trial 58). The target velocity for this experiment was also set to 0.5 m/s. We observe the robot is able to adapt under progressively increasing payload conditions, quickly adapting the parameters to maintain the target velocity and lower energy consumption. Table I showcases the significant improvements in velocity tracking, CoT, and objective value following adaptation compared with the performance using the parameters optimized under normal conditions, thereby confirming the effectiveness of our framework.

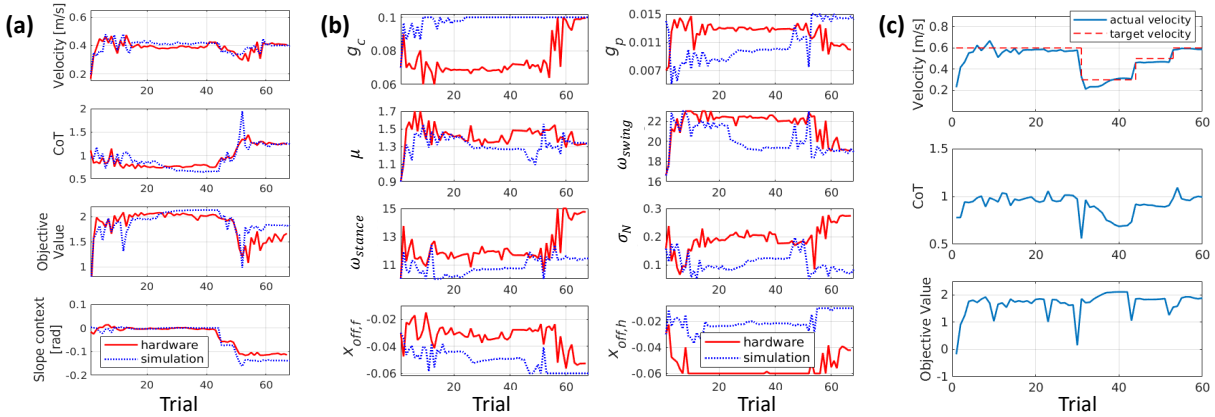

Fig. 4: Slope adaptation and velocity tracking in hardware experiments. **(a)** mean velocity, CoT, objective value, and slope context, increasing the slope to 10 degrees while tracking 0.4 m/s. **(b)** optimized parameters evolution during **(a)**, from flat terrain locomotion to slope adaptation. **(c)** adaptation to changing velocity commands (0.6→0.3→ 0.5→0.6 m/s) on the hardware.

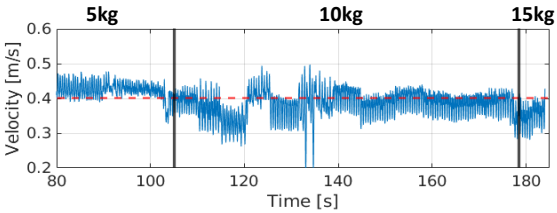

Fig. 5: Forward velocity during the hardware experiment of adaptation up to a 15 kg load, tracking a target velocity of 0.4 m/s. Before this adaptation, the robot was trained on flat terrain without any load for three minutes.

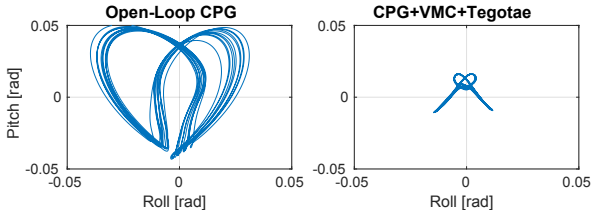

Fig. 6: Comparison of roll-pitch variations of the optimized open-loop CPG, and of the CPG with force feedback (Tegotae) and Virtual Model Control after learning in simulation.

D. Hardware Experiments

We validate our framework's effectiveness of learning optimal locomotion and adaptation in hardware experiments using the CPG with force feedback and VMC on the Unitree Go1 robot. Our experiments successfully demonstrate:

- single velocity tracking at 0.4 m/s with CoT $\approx$ 0.8 on normal terrain and with CoT $\approx$ 0.95 on slippery terrain
- multi-velocity tracking, transitioning from 0.6 m/s (CoT $\approx$ 0.95), to 0.3 m/s (CoT $\approx$ 0.70), to 0.5 m/s (CoT $\approx$ 0.90), Figure 4c
- adaptation to a 10-degree slope, Figure 4a and 4b
- adaptation to increasing loads up to 15 kg, Figure 5

Figures 4a and 4b demonstrate how our framework allows the Go1 robot to adapt online to a 10-degree slope while tracking a target velocity of 0.4 m/s. The optimizer first converges to a set of parameters which facilitates tracking of the target velocity on flat terrain within fewer than 40 trials (**less than 3 minutes**) of learning. Subsequently, after the slope was increased to 10 degrees, the robot stabilizes around this target once more within approximately 20 trials (**around 2 minutes**). Figure 4b compares the final parameters that enable reaching the target velocity on the 10-degree slope after adaptation, with those optimized for flat terrain in the hardware experiment. We observe a decrease in ω_{swing}, $x_{offset,front}$, and g_p. Conversely, there is an increase in ω_{stance}, g_c, σ_N, and $x_{offset,hind}$. When comparing the changes in parameters between the simulation and the hardware experiment, we find that $g_c, \mu, \omega_{swing}, x_{offset,front}$ converge to similar values under both conditions. Meanwhile, $x_{offset,hind}$ in the hardware experiment demonstrates a similar (but shifted) trend to that observed in simulation during adaptation.

Figure 5 shows the robot base linear velocity after placing loads of 5 kg, 10 kg, and 15 kg on the robot, which began on flat terrain without any load to track a velocity of 0.4 m/s. This result demonstrates the ability of our framework to enable the robot to adaptively locomote under various load conditions through the online optimization.

E. Ablation Study of CPG Controllers

We perform ablation studies towards understanding the importance of augmenting the CPG with both force feedback and posture control. We repeat the optimization by optimizing parameters for the following four controllers: 1) Open-Loop CPG, 2) CPG+VMC, 3) CPG+Tegotae, and 4) CPG+VMC+Tegotae (our method). When Tegotae is not active, we optimize one less parameter, since σ_N is not needed. Table II compares the final average performance of five optimization runs with different random seeds for each controller type in maintaining the same target velocity 0.5 m/s.

As demonstrated in Table II, controllers incorporating Tegotae (force feedback) consistently achieve lower CoT with their final parameters compared to their counterparts. This outcome suggests that force feedback could contribute to generating higher-quality and more energy-efficient gaits, as reflected by the lower average final CoT. While the VMC component alone appears to be less critical than Tegotae force feedback in enhancing gait quality — particularly when comparing the CPG controllers with and without VMC — the results indicate that combining VMC with force feedback further improves energy efficiency.

Moreover, controllers lacking force feedback exhibited correspondingly lower objective values, and higher standard

TABLE II: Mean velocity, CoT and objective value using different CPG controllers for the target velocity $v_x^* = 0.5$ m/s. These results were derived by averaging the outcomes of the last five trials in a single experiment and then further averaging these across five experiments, each with distinct random seeds. Each experiment consisted of 40 trials.

Controller	Velocity [m/s]	CoT	Objective Value
OpenLoop	0.477 ± 0.045	0.931 ± 0.138	1.638 ± 0.516
VMC	0.494 ± 0.028	0.919 ± 0.149	1.643 ± 0.139
Tegotae	0.500 ± 0.012	0.583 ± 0.024	2.153 ± 0.042
VMC+Tegotae	0.509 ± 0.023	0.518 ± 0.050	2.167 ± 0.072

deviations of these values, indicating less consistent and more variable optimization performance. This instability could stem from a reduced area of promising parameters in the search space for controllers without force feedback, thereby complicating the identification of optimal parameters within a limited time frame.

Figure 6 compares the roll-pitch trajectory of the robot's attitude when applying the best individual parameters for both the Open-Loop CPG controller and the CPG+VMC+Tegotae controller after optimization. The force feedback (Tegotae) and VMC greatly improve the stability by reducing the roll and pitch variations, contributing to better energy efficiency and velocity tracking.

V. Conclusion

In this study, we introduced a framework capable of efficient online learning of CPG-based locomotion for quadruped robots, both in simulation and in hardware experiments, utilizing Bayesian optimization. This is in contrast to other approaches for optimizing the CPG parameters, typically done through manual tuning or offline optimization. Our results demonstrate that our framework facilitates rapid learning and adaptation to varying target velocities, and to terrains with varying coefficients of friction. Furthermore, by introducing contextual variables into Bayesian optimization, we achieve rapid adaptation to a 10-degree slope and a 15 kg load (125% of the nominal mass) on the Unitree Go1 hardware. To the best of our knowledge, this represents the highest added payload ever achieved on the Go1 robot. Moreover, our findings suggest that energy efficiency and gait stability are much improved by incorporating force feedback (as well as adding posture control (VMC)) into the CPG.

Current limitations of this work include the absence of explicit safety guarantees during parameter searching, as well as the potential applicability to more rugged terrain. Future work will aim to incorporate more advanced feedback controllers and BO with safety constraints to address these issues. We also plan to learn parameters for different gaits, as well as optimize joint-space CPGs, towards a better understanding of the biological parallels in learning locomotion.

References

[1] N. Dominici, Y. P. Ivanenko, G. Cappellini, A. d'Avella, V. Mondì, M. Cicchese, A. Fabiano, T. Silei, A. D. Paolo, C. Giannini, R. E. Poppele, and F. Lacquaniti, "Locomotor primitives in newborn babies and their development," *Science*, vol. 334, no. 6058, pp. 997–999, 2011.

[2] M. Garwicz, M. Christensson, and E. Psouni, "A unifying model for timing of walking onset in humans and other mammals," *Proceedings of the National Academy of Sciences*, vol. 106, no. 51, pp. 21 889–21 893, 2009.

[3] G. Orlovsky, T. G. Deliagina, and S. Grillner, *Neuronal Control of Locomotion: From Mollusc to Man*. Oxford University Press, 09 1999.

[4] A. J. Ijspeert, A. Crespi, D. Ryczko, and J.-M. Cabelguen, "From swimming to walking with a salamander robot driven by a spinal cord model," *Science*, vol. 315, no. 5817, pp. 1416–1420, 2007.

[5] A. Crespi and A. J. Ijspeert, "Online optimization of swimming and crawling in an amphibious snake robot," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 75–87, 2008.

[6] P. Manoonpong, T. Geng, T. Kulvicius, B. Porr, and F. Wörgötter, "Adaptive, fast walking in a biped robot under neuronal control and learning," *PLOS Computational Biology*, vol. 3, no. 7, pp. 1–16, 07 2007.

[7] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, no. 4, pp. 642–653, 2008, robotics and Neuroscience.

[8] A. J. Ijspeert and J. Kodjabachian, "Evolution and development of a central pattern generator for the swimming of a lamprey," *Artificial Life*, vol. 5, no. 3, p. 247 – 269, 1999.

[9] A. J. Ijspeert, "A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander," *Biological Cybernetics*, vol. 84, no. 5, p. 331 – 348, 2001.

[10] H. Inada and K. Ishii, "Behavior generation of bipedal robot using central pattern generator(cpg) (1st report: Cpg parameters searching method by genetic algorithm)," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3, 2003, pp. 2179–2184 vol.3.

[11] C. Paul and J. Bongard, "The road less travelled: morphology in the optimization of biped robot locomotion," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 1, 2001, pp. 226–232 vol.1.

[12] G. Bellegarda and A. Ijspeert, "CPG-RL: Learning central pattern generators for quadruped locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 12 547–12 554, 2022.

[13] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Puppeteer and marionette: Learning anticipatory quadrupedal locomotion based on interactions of a central pattern generator and supraspinal drive," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1112–1119.

[14] G. Bellegarda, M. Shafiee, and A. Ijspeert, "Visual CPG-RL: Learning central pattern generators for visually-guided quadruped locomotion," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 1420–1427.

[15] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Viability leads to the emergence of gait transitions in learning agile quadrupedal locomotion on challenging terrains," *Nature Communications*, vol. 15, no. 1, p. 3073, 2024.

[16] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Manyquadrupeds: Learning a single locomotion policy for diverse quadruped robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 3471–3477.

[17] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "An experimental comparison of bayesian optimization for bipedal locomotion," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 1951–1958.

[18] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," *Annals of Mathematics and Artificial Intelligence*, vol. 76, no. 1, pp. 5–23, Feb 2016.

[19] A. Rai, R. Antonova, S. Song, W. Martin, H. Geyer, and C. Atkeson, "Bayesian optimization using domain knowledge on the atrias biped," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1771–1778.

[20] A. Rai, R. Antonova, F. Meier, and C. G. Atkeson, "Using simulation to improve sample-efficiency of bayesian optimization for bipedal robots," *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 1844–1867, 2019.

[21] D. Widmer, D. Kang, B. Sukhija, J. Hübotter, A. Krause, and S. Coros, "Tuning legged locomotion controllers via safe bayesian optimization," *arXiv preprint arXiv:2306.07092*, 2023.

[22] G. Bellegarda, M. Shafiee, M. E. Özberk, and A. Ijspeert, "Quadruped-Frog: Rapid online optimization of continuous quadruped jumping," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 1443–1450.

[23] F. Ruppert and A. Badri-Spröwitz, "Learning plastic matching of

robot dynamics in closed-loop central pattern generators," *Nature Machine Intelligence*, vol. 4, no. 7, pp. 652–660, Jul 2022.
[24] D. Owaki, T. Kano, K. Nagasawa, A. Tero, and A. Ishiguro, "Simple robot suggests physical interlimb communication is essential for quadruped walking," *J. R. Soc. Interface*, vol. 10, p. 201200669, 2013.
[25] D. Owaki and A. Ishiguro, "A quadruped robot exhibiting spontaneous gait transitions from walking to trotting to galloping," *Scientific reports*, vol. 7, no. 1, pp. 1–10, 2017.
[26] R. Thandiackal, K. Melo, L. Paez, J. Herault, T. Kano, K. Akiyama, F. Boyer, D. Ryczko, A. Ishiguro, and A. J. Ijspeert, "Emergence of robust self-organized undulatory swimming based on local hydrodynamic force sensing," *Science Robotics*, vol. 6, no. 57, 2021.
[27] M. Ajallooeian, S. Pouya, A. Sproewitz, and A. J. Ijspeert, "Central pattern generators augmented with virtual model control for quadruped rough terrain locomotion," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3321–3328.
[28] A. Winkler, I. Havoutis, S. Bazeille, J. Ortiz, M. Focchi, R. Dillmann, D. Caldwell, and C. Semini, "Path planning with force-based foothold adaptation and virtual model control for torque controlled quadruped robots," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 6476–6482.
[29] P. I. Frazier, "A tutorial on bayesian optimization," 2018.
[30] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.
[31] B. H. Yang, G. Wang, R. Calandra, D. Contreras, S. Levine, and K. S. J. Pister, "Learning flexible and reusable locomotion primitives for a microrobot," *CoRR*, vol. abs/1803.00196, 2018.
[32] A. Krause and C. Ong, "Contextual gaussian process bandit optimization," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24. Curran Associates, Inc., 2011.
[33] G. Gabrielli, "What price speed? specific power required for propulsion of vehicles," *Mech. Eng.*, pp. 775–781, 1950.
[34] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems 33*, 2020.
[35] L. P. Fröhlich, C. Küttel, E. Arcari, L. Hewing, M. N. Zeilinger, and A. Carron, "Model learning and contextual controller tuning for autonomous racing," 2021.

VI. Appendix

A. Parameter Setting

TABLE III: Optimization parameter ranges

g_c [m]	[0.06, 0.10]
g_p [m]	[0.005, 0.015]
ω_{swing} [rad/s]	[16, 23]
ω_{stance} [rad/s]	[10, 15]
μ	[0.9, 1.7]
$x_{offset,front}$[m]	[-0.06, -0.01]
$x_{offset,hind}$[m]	[-0.06, -0.01]
σ_N	[0.05, 0.30]

TABLE IV: Control parameters

w_{ij}	1
α	50
d_{step} [m]	0.05
h [m]	0.3
f [kHz]	1
k_{dir}	300
$k_{att,r}$	150
$k_{att,p}$	160
$K_{p,joint}$	70
$K_{d,joint}$	1.3

1) Optimization Parameter Ranges: The ranges used for the optimization parameters in both simulation and hardware experiments are provided in Table III.

2) Control Parameters: In Table IV, f represents the control frequency, and $k_{dir}, k_{att,r}, k_{att,p}$ correspond to the gains for yaw, roll, and pitch control for the VMC controller, respectively.

3) Parameter Normalization: Both optimization and context parameters require normalization based on certain ranges before computing the surrogate model with a Gaussian Process. We normalize optimization parameters within the ranges provided above, and contextual variables as follows: average normal force $c_{load} \in [15,55]$, average pitch of the trunk $c_{slope} \in [-0.4,0.1]$. These ranges are determined based on the tasks and physical sensor capabilities of the Unitree Go1 robot, ensuring that the contextual variables remain within realistic bounds.

TABLE V: Parameters used to update exploration factor β

β_{init}	5
$\beta_{init,c}$	1.5
γ	0.7
n_{decay}	10
$n_{decay,c}$	3

B. Exploration Factor Updating

1) Parameters used in the UCB function: The values of these parameters can be found in Table V. After starting the optimization, we use the value of β_{init} and n_{decay} in the table in Eq. 8. When the estimator detects changes in the contextual variables c_{load} and c_{slope} during the optimization, the exploration factor β is updated based on the smaller value $\beta_{init,c}$, $n_{decay,c}$, which is shown as follows:

$$\beta = \beta_{init,c} \cdot \gamma^{\max(n-n_{decay,c},0)} \tag{13}$$

This is done because given the optimizer's ability to rapidly identify a suitable set of gait parameters for new contexts — leveraging knowledge from previous contexts via CBO — a reduced emphasis on exploration could be warranted. Moreover, excessive exploration, especially during such adaptations, could cause a robot fall due to the increased task difficulty. Consequently, we adopt a strategy that prioritizes reduced exploration during contextual adaptation, enhancing safety within our optimization framework.

2) Determination of Context Sharing: The assessment of whether trials occur within the same context is based on predefined thresholds, t_{slope} and t_{load}. A trial is considered to be in the same context if it meets the following criterion:

$$n = \text{count} \left\{ \begin{array}{c} c^i_{load} - t_{load} r_{load} \leq \mathbf{c}_{load} \leq c^i_{load} + t_{load} r_{load} \\ \text{and} \\ c^i_{slope} - t_{slope} r_{slope} \leq \mathbf{c}_{slope} \leq c^i_{slope} + t_{slope} r_{slope} \end{array} \right\}, \tag{14}$$

where $\mathbf{c}_{load}, \mathbf{c}_{slope}$ are vectors containing the history of load and slope contextual values, respectively. c^i_{load} and c^i_{slope} represent the contextual values for the current trial, and the variables $r_{load}(=40), r_{slope}(=0.5)$ denote the ranges of the load context and slope context. The threshold values are defined as $t_{load} = 0.2$, $t_{slope} = 0.1$.

C. Terrain Setup for Evaluating a Slope Adaptation Model

As mentioned in Section IV-C, the model that had been optimized for flat terrain, as well as slopes of 10 degrees and 12.5 degrees, was further evaluated on a terrain that featured a continuous range of slopes from 10 degrees up to 17.5 degrees. The terrain setup for this evaluation is shown in Figure 7.

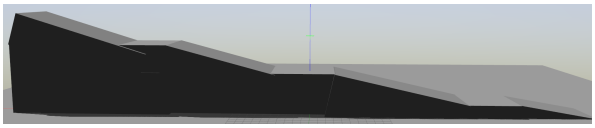

Fig. 7: Simulation settings for inference mode of slope adaptation in Gazebo.