2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Quadruped-Frog: Rapid Online Optimization of Continuous Quadruped Jumping

Guillaume Bellegarda, Milad Shafiee, Merih Ekin Özberk, Auke Ijspeert

***Abstract*— Legged robots are becoming increasingly agile in exhibiting dynamic behaviors such as running and jumping. Usually, such behaviors are either optimized and engineered *offline* (i.e. the behavior is designed for *before* it is needed), either through model-based trajectory optimization, or through deep learning-based methods involving millions of timesteps of simulation interactions. Notably, such offline-designed locomotion controllers cannot perfectly model the true dynamics of the system, such as the motor dynamics. In contrast, in this paper, we consider a quadruped jumping task that we rapidly optimize *online*. We design foot force profiles parameterized by only a few parameters which we optimize for directly on hardware with Bayesian Optimization. The force profiles are tracked at the joint level, and added to Cartesian PD impedance control and Virtual Model Control to stabilize the jumping motions. After optimization, which takes only a handful of jumps, we show that this control architecture is capable of diverse and omnidirectional jumps including forward, lateral, and twist (turning) jumps, even on uneven terrain, enabling the Unitree Go1 quadruped to jump 0.5 m high, 0.5 m forward, and jump-turn over 2 rad.**

I. Introduction

Many animals exhibit a variety of dynamic and agile locomotion and jumping motor behaviors. For example, Springboks and Thomson's gazelles use stotting (or pronking) to gracefully locomote at high speeds [1]. Various hypotheses have been proposed as possible explanations for this stotting behavior, including pursuit deterrence, rapid escape, anti-ambush behavior, and predator detection [2]. Another explanation is play, often seen in baby goats. Many animals, and especially hoofed animals, are able to locomote within minutes of birth [3], [4], and baby goats can be seen to rapidly improve and fine-tune their jumping motor behaviors within their first hours/days. This is possible through primitive patterns of neural control known to exist in vertebrates, also known as Central Pattern Generators (CPGs) [5], which can produce locomotion in the absence of descending drive from higher centers.

In robotics, dynamic motor skills such as running and jumping for legged robots have recently drawn increased interest due to advances in both hardware capabilities and control architectures. Trajectory Optimization (TO) approaches have shown that optimizing over the full system dynamics allows for the generation and tracking of highly dynamic jumping motions on hardware [6]–[10]. Recent work extends such single jumps with Model Predictive Control (MPC) to transition between jumps, enabling continuous jumping on stepping stones [11]. When the TO can be solved online, i.e. using simplified dynamics models to run as MPC, highly dynamic and robust locomotion skills can be realized on quadruped hardware [12]–[15].

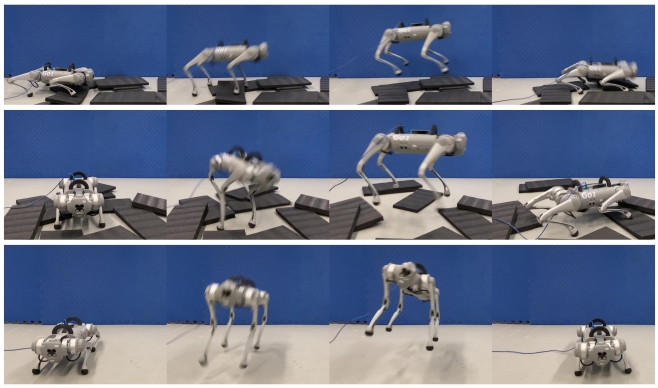

Fig. 1: Online optimized jumping. Top: forward jumping on rough terrain (0.5 m height, 0.5 m distance). Middle: twist jump, over 2 rad. Bottom: lateral jumping 0.3 m.

Another approach to generating running and jumping controllers is through deep reinforcement learning. For example, fast trot-running [16], [17] and bounding [18] have autonomously emerged end-to-end through learning frameworks. Advanced skills can also be learned by incorporating terrain-awareness for tasks such as climbing and jumping gaps [19], or rough terrain locomotion in the wild [20]. As reward function design and exploration can be issues, other works use reference motions, careful hierarchical training schemes, and/or policy experience transfer to learn difficult jumping skills [21], [22].

Leveraging ideas and methods from both model-based control and learning-based control can help surpass the performance of either method individually [23]. For example, some tasks may be difficult to model, or overly conservative, with model-based control. Conversely, learning-based methods can suffer from difficulties with appropriately exploring the state space, and reward function tuning. Choosing an alternative action space to desired joint positions is one such beneficial combination [18], [24]. For example, learning foot position residuals on top of optimized jumping trajectories allows for jumping off of uneven terrain and with significant noise [25]. Combining MPC with reinforcement learning also allows for dynamic gap crossing capabilities [26]–[29], or continuous jumping through learned action residuals [30]. Our previous work

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

uses a hierarchical biology-inspired control architecture which leverages deep reinforcement learning with dynamical systems (CPGs) in the loop for both robust locomotion [31]–[33], as well as to learn rapid gap crossing abilities [34], [35].

The above works can take significant engineering effort and time to properly tune various parameters in both optimal control frameworks (i.e. contact timing, cost function, constraints, gains) and learning-based frameworks (i.e. reward function design, training time, sim-to-real gap). There is also usually no online adaptation to continue to improve the methods, with some exceptions, for example continuing the training process from simulation to fine-tune locomotion controllers on hardware [36]. This is still in contrast to the rapid adaptation of animals, which start with innate locomotion skills, and rapidly improve them with few real-world interactions.

In this light, Bayesian Optimization has previously been applied directly on hardware to rapidly tune parameters to improve legged robot gaits for both bipeds [37]–[40] and quadrupeds [41], [42]. However, such approaches have not yet been demonstrated for more dynamic locomotion skills, such as continuous jumping.

A. Contribution

Inspired by animals and recent robotics works, we present an omnidirectional jumping controller which can be optimized online directly on hardware. We parameterize the jumps through desired force profiles to be applied at the feet, which are similar to force profiles for jumping and landing in animals such as frogs [43]. The force profiles are tracked at the joint level, along with Cartesian PD impedance control which regulates nominal foot positions. To stabilize the jumping motions, we add Virtual Model Control. Our framework allows us to rapidly optimize dynamic jumping behaviors to continuously perform forward, lateral, and twist jumps, even under disturbances of uneven terrain and varying coefficients of friction. This online optimization is in contrast to other works which optimize jumping offline, and this allows us to optimize directly on the real system without any dynamics uncertainties or mismatches (i.e. as is the case with simulating motor dynamics, friction, etc.).

The rest of this paper is organized as follows. In Section II we present our jumping parameterization design choices and integration of Bayesian Optimization. In Section III we discuss results and analysis from optimizing our controller to perform omnidirectional and continuous jumps, even when subjected to disturbances and noise in the form of uneven terrain. Section IV concludes the paper and suggests future directions for further work.

II. Method

In this section we describe our online jumping optimization framework and design decisions for developing omnidirectional jumping controllers for quadruped robots. Based on different cost functions associated with jumping different directions, the optimization updates the desired force profile impulses to be applied at each of the feet, varying both the frequency and magnitude of these parameters. These force profiles are tracked at the joint level with the foot Jacobian, and Cartesian PD impedance control helps to keep the feet at a nominal position beneath the hips. To avoid large Cartesian PD gains as well as improve the robot stability, we add Virtual Model Control to regulate the base roll and pitch. This additionally allows the robot to jump in uneven terrains. A high-level control diagram is illustrated in Figure 2, and we explain all components below.

A. Generating Jumping Motion Behaviors

Jumping is a dynamic and coordinated movement that relies on the intricate interplay of muscles and neural circuits. At the heart of this impressive skill are Central Pattern Generators (CPGs), specialized neural networks found within the spinal cord that generate rhythmic motor patterns. These neural circuits play a crucial role in orchestrating the precise sequence of muscle contractions required for a successful jump. By producing the necessary motor commands and coordinating the timings of muscle activations, CPGs enable organisms, from frogs to humans, to execute powerful and well-timed leaps, making jumping an intriguing example of the neural control of complex movements.

To represent the CPG circuits in the spinal cord for generating quadruped locomotion skills, a number of abstract oscillators have been proposed, with some of the most popular including Matsuoka oscillators [44] and phase oscillators [45], [46]. These typically generate rhythm in either joint space [46], [47] or task space (i.e. for each limb) [31], [45], with coupling between different joints and limbs to produce different gaits. In our previous works [34], [35], we used deep reinforcement learning to modulate the CPG, and thus task space positions, in order to dynamically locomote across gaps.

In contrast, in this paper, due to the inherent force interaction during jumping, we choose to optimize force profiles rather than joint or task space positions. Notably, this essentially re-formulates the abstract oscillators that are typically used to model the CPGs into force space. We take inspiration from frogs which have approximate half-sine wave force profiles for both take-off and landing in both horizontal and vertical directions [43].

As in previous works using abstract oscillators, we monitor the system state with the (uncoupled) phase oscillator:

$$\dot{\theta} = 2\pi f_i \ , \quad \text{where } f_i = \begin{cases} f_0 & \text{if } \pi \leq \theta < 2\pi \\ f_1 & \text{if } 0 \leq \theta < \pi \end{cases} \tag{1}$$

where θ is the phase of the oscillator. The frequency f_i changes based on the phase, which dictates whether to apply the impulse force, or to be inactive (for example during flight, or time between jumps). We define the impulse phase frequency as f_0 (to apply the force with the foot in contact with the ground) as $\pi \leq \theta < 2\pi$, and the off-phase frequency as f_1 when $0 \leq \theta < \pi$. Therefore, the impulse time duration T_{impulse} will be $\frac{1}{2f_0}$, and the time between impulses will be $T_{\text{off}} = \frac{1}{2f_1}$. Given these phase relationships, Figure 3 shows

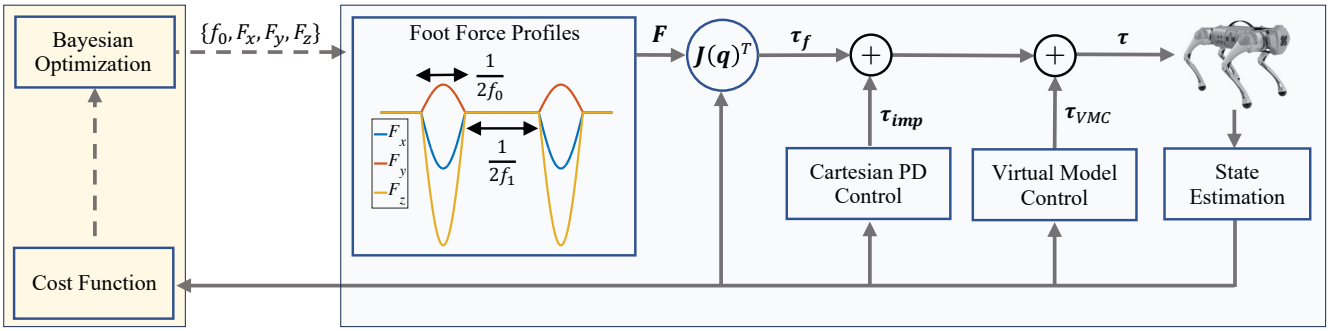

Fig. 2: Control architecture for online jumping optimization. The right (blue) box represents the environment, where the solid arrows operate at 1 kHz. Desired foot force profiles are mapped to torques with the Jacobian. Cartesian PD impedance control helps to regulate the foot at a nominal position below the hips, and Virtual Model Control is added to help stabilize the robot and allow jumping in uneven terrain. The left (yellow) box represents the Bayesian Optimization, which selects new force profile parameters after each jump based on the accumulated cost function specifying the task.

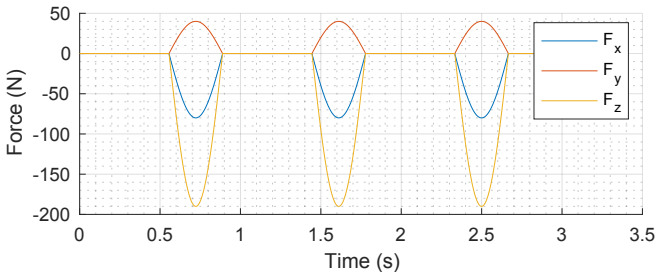

Fig. 3: Force trajectories for hopping forwards and right in the body frame. When the impulse is not active, the system is in the air, or landing. Parameter f_0 determines the frequency of the impulse, and parameter f_1 is the frequency between impulses (and is not optimized).

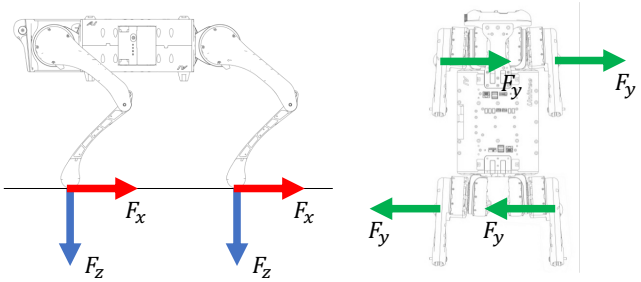

Fig. 4: Jumping force directions visualized at the feet of the Unitree Go1 quadruped. Left: planar XZ forces applied at the feet for jumping forward. Right: top view of lateral forces for performing a counterclockwise twist jump.

the force profiles and directions for a set of parameters for setting the 3D force vector at the foot of each leg:

$$\boldsymbol{F}_i = \begin{cases} [F_x \;\; F_y \;\; F_z]^\top \sin(\theta) & \text{if } \sin(\theta) < 0 \\ \boldsymbol{0}_{3\times 1} & \text{otherwise} \end{cases} \tag{2}$$

where F_x, F_y, F_z are the force amplitudes applied at the foot contact in the local frame. By coordinating the signs of the forces F_x and F_y, jumping can be accomplished in different directions, for example jumping forward by applying forces in the X and Z directions for all feet (Figure 4 left), or a twist turn by changing the direction of the applied Y force for the front and rear feet (Figure 4 right).

B. Leg Controller

The forces from the above force profiles can be applied at each foot by mapping the forces to torques at the joint level with:

$$\boldsymbol{\tau}_{\mathrm{f}} = \boldsymbol{J}(\boldsymbol{q})^\top \boldsymbol{F} \tag{3}$$

where $\boldsymbol{J}(\boldsymbol{q})$ is the foot Jacobian at joint configuration $\boldsymbol{q}$. We also add a Cartesian PD impedance controller to regulate the foot to a nominal position below the hips. The foot position error is mapped to torques and tracked at the joint level with the following controller for each leg i:

$$\boldsymbol{\tau}_{\mathrm{imp}} = \boldsymbol{J}(\boldsymbol{q})^\top \left[\boldsymbol{K}_p(\boldsymbol{p}_d - \boldsymbol{p}) - \boldsymbol{K}_d(\boldsymbol{v}) \right] - \boldsymbol{K}_{d,joint}(\dot{\boldsymbol{q}}) \tag{4}$$

where $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are diagonal matrices of proportional and derivative gains in Cartesian coordinates to track the desired foot positions $(\boldsymbol{p}_d)$ with zero desired foot velocity $(\boldsymbol{v})$ in the leg frame. We add a small joint damping term for stability in the hardware experiments. We use $\boldsymbol{K}_p = 400\boldsymbol{I}_3$, $\boldsymbol{K}_d = 8\boldsymbol{I}_3$, $\boldsymbol{K}_{d,joint} = 0.8\boldsymbol{I}_3$.

C. Virtual Model Control

To improve the stability of the system, robustness to uneven terrain, and avoid large Cartesian space gains, we add Virtual Model Control (VMC) while in contact with the ground, similar to [48], [49]. We attach virtual springs between a hypothetical XY plane through the center of the trunk, and with another plane horizontal with respect to the world coordinates. These virtual springs naturally generate forces to adjust the attitude (pitch and roll) of the body to be parallel to the ground:

$$\boldsymbol{P} = \boldsymbol{R} \begin{bmatrix} 1 & 1 & -1 & -1 \\ -1 & 1 & -1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5}$$

$$\boldsymbol{F}_{VMC} = \begin{bmatrix} \boldsymbol{0}_{2\times 4} \\ k_{att}(\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \boldsymbol{P}) \end{bmatrix} \tag{6}$$

where $\boldsymbol{R}$ is the robot's rotation matrix with respect to the world coordinates, $\boldsymbol{P}$ are the relative coordinates of the corners of the virtual plane through the body, $k_{att} = 200$ is the gain, and the columns of $\boldsymbol{F}_{VMC}$ are the virtual forces

to be added to each leg i (Front Right, Front Left, Rear Right, Rear Left).

The VMC force contribution column i is added to the feedforward force controller and impedance controller for leg i with:

$$\boldsymbol{\tau}_{\mathrm{VMC},i} = \boldsymbol{J}_i(\boldsymbol{q}_i)^\top \boldsymbol{F}_{VMC_i} \tag{7}$$

making the full torque vector for each leg i:

$$\boldsymbol{\tau}_i = \boldsymbol{\tau}_{\mathrm{f},i} + \boldsymbol{\tau}_{\mathrm{imp},i} + \boldsymbol{\tau}_{\mathrm{VMC},i} \tag{8}$$

D. Bayesian Optimization

Bayesian optimization is a powerful and versatile approach to solving complex optimization problems. It is a probabilistic model-based optimization technique that leverages Bayesian statistics to efficiently explore and exploit the parameter space of a function to find the optimal solution. Unlike more traditional optimization methods which generate jumping maneuvers offline [6], [7], Bayesian optimization is particularly well-suited for online problems with noisy or expensive-to-evaluate objective functions, where it strives to strike a balance between exploration (searching for promising areas) and exploitation (focusing on areas likely to yield the best results).

In Bayesian optimization, a probabilistic model (typically Gaussian Process or Tree-Parzen Estimator (TPE)) is built to represent the unknown objective function. TPE uses a tree-structured approach to model the objective function as a combination of two probability distributions: one for the promising parameter configurations, and another for the less promising ones. These distributions guide the exploration and exploitation of the parameter space. TPE adaptively selects and evaluates candidate configurations that are more likely to improve the objective, which makes it highly sample-efficient. In this paper, we use the TPE implementation of Optuna [50] to optimize several omnidirectional jumping tasks.

1) Forward Jumping: The first task is continuous jumping forward. The cost function is defined as the difference between the final x position in the world frame after the jump, and the initial x position before the jump:

$$J_{\mathrm{fwd}} = x_{\mathrm{final}} - x_{\mathrm{init}} \tag{9}$$

2) Lateral Jumping: The second task is continuous lateral jumping. The cost function is defined as the difference between the final y position in the world frame, and the initial y position. The sign can be changed for jumping left $(+)$ in the body frame, or $(-)$ for jumping right:

$$J_{\mathrm{lat}} = (\pm)\ (y_{\mathrm{final}} - y_{\mathrm{init}}) \tag{10}$$

3) Twist Jumping: The third task is continuous twist jumping (yaw rotational jump). The cost function is defined as the difference between the the final yaw angle and initial yaw angle in the world frame. The sign can be changed for twist jumping counterclockwise $(+)$ in the body frame, or $(-)$ for jumping clockwise:

$$J_{\mathrm{twist}} = (\pm)\ (\psi_{\mathrm{final}} - \psi_{\mathrm{init}}) \tag{11}$$

TABLE I: Optimization parameter ranges for generating forward, lateral, and twist (turn) jumping.

Parameter	Lower Bound	Upper Bound	Units
f_0	0.75	1.75	Hz
F_x	0	150	N
F_y	0	150	N
F_z	150	350	N

In case the robot falls during a jump due to a poor selection of parameters, we set $J = 0$ for that iteration of the optimization. This is to avoid exploiting the dynamics to get a high objective value even though the robot may no longer be standing.

III. Experimental Results and Discussion

In this section we report and discuss results from the online optimization of our jumping controllers. Sample snapshots of the quadruped jumping in uneven terrain are shown in Figure 1, and the reader is encouraged to watch the supplementary video for clear visualizations of the discussed experiments.

1) Implementation Details: We use the Unitree Go1 quadruped [51], and the Optuna library implementation of the TPE algorithm for the Bayesian optimization [50]. Initial tests and optimizations are carried out in simulation in Gazebo before moving to hardware. The parameters and their ranges we optimize are listed in Table I. The Cartesian PD and Virtual Model Control gains are tuned once on flat terrain so the robot can maintain balance and remain standing. We compute the cost function and run the Bayesian optimization to update the parameters after each jump for single jumps. Each iteration thus corresponds to a single jump.

2) Optimization Convergence: We first investigate the sample-efficiency and optimization time of applying Bayesian optimization to tune the force profiles for our jumping control architecture. For each type of jump, we run 5 different optimizations with different random seeds and observe the robot's ability to successfully complete the desired jumps. Figure 5 shows the mean objective value across the 5 random seeds for each of the three jumping tasks vs. Bayesian Optimization iteration in the Gazebo simulation. In all cases, we can observe the rapid convergence and improvement of the quadruped's ability to jump forward, laterally, and twist-turn. Due to the ideal modeling (i.e. motors are not modeled, the feet do not slip as the coefficient of friction is high), the robot can typically jump farther and better in simulation. For example, the robot is able to optimize a twist-turn jump of 0.67 m in height and over 360 degrees in rotation in simulation while respecting the torque limits of the robot, as shown in Figure 8.

Figure 6 shows the optimization process for optimizing forward jumping directly on the hardware. We observe that the objective value increases during the training, which is the distance that the quadruped jumps forward. Similarly to in simulation, we see rapid convergence to being able to jump 0.5 m forward. In order to accomplish such a jump, the optimizer selects the parameters F_x, F_z, and f, which are

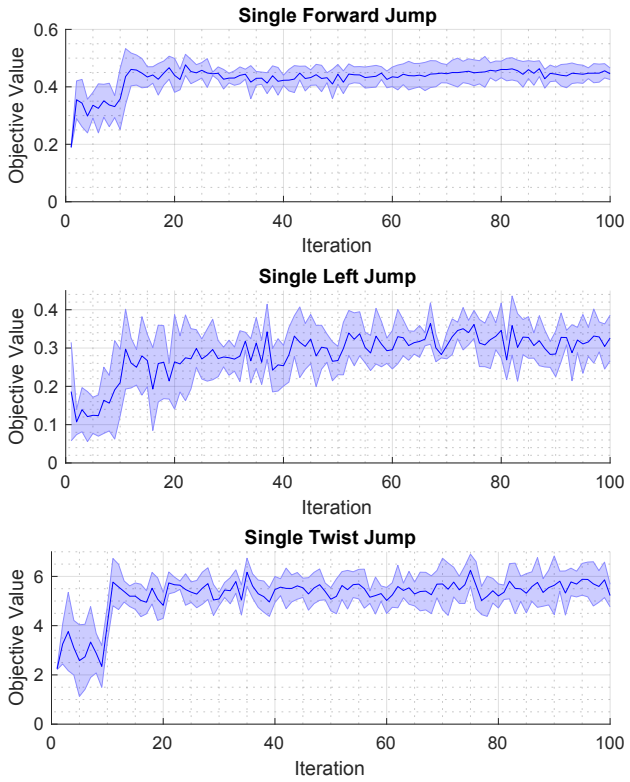

Fig. 5: Training curves averaged across 5 runs with different random seeds for optimizing forward (top), lateral left (middle), and twist-turn (bottom) jumps in Gazebo. All runs result in successful jumping.

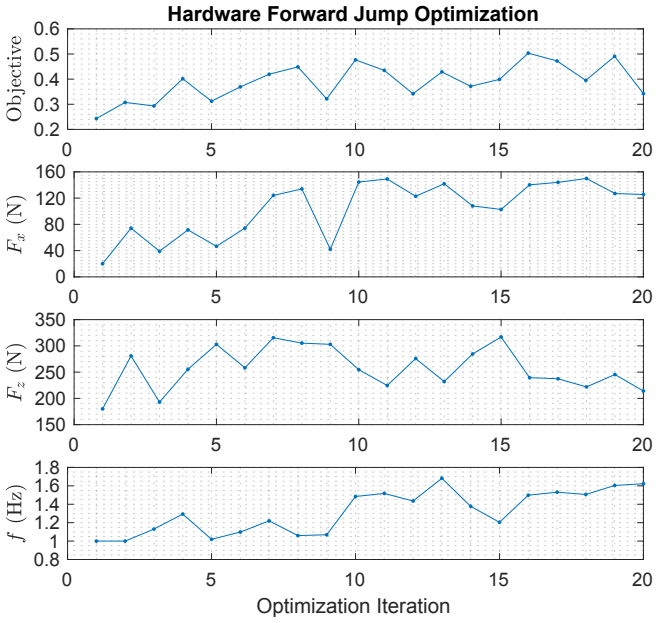

Fig. 6: Forward jumping optimization on the Unitree Go1 hardware in 20 iterations. From top to bottom: objective value (forward distance jumped), force profile parameter F_x, force profile parameter F_z, and frequency f. Notably, there is a trade-off between forward/downward force that must occur when jumping forward.

also shown in the Figure. Notably, there must be a trade-off between these parameters, which must maintain a ratio to jump high enough so that the robot does not slip and fall

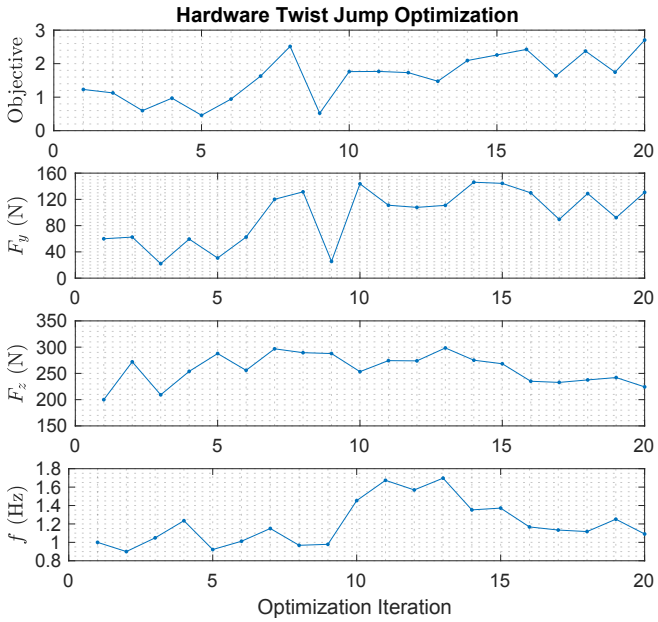

Fig. 7: Twist-turn jumping optimization on the Unitree Go1 hardware in 20 iterations. From top to bottom: objective value (yaw rotation jumped), force profile parameter F_y, force profile parameter F_z, and frequency f. Notably, there is a trade-off between lateral/downward force that must occur when twist-jumping to both jump high enough, as well as generate a large moment.

forward, but also not too high so that the robot does not make any forward progress. The best jumps can be seen when F_z is not at its maximum. Interestingly, the best jump occurs at Trial 17, where $F_x = 140N$ and $F_z = 239N$, corresponding to an overall take-off angle (from the force direction) of approximately 60 degrees. This is in line with observations from biological frogs [43], as well as robotic frogs [52].

Figure 7 shows the optimization process for optimizing a twist-turn jump directly on the hardware. As in the forward case, the robot rapidly improves its jump-turning abilities within 20 trials. There is again the trade-off between applying the forces F_y (in opposite directions for the front/rear feet to generate the moment), and the magnitude of the applied downward force F_z. Due to the motor dynamics and coefficient of friction causing the feet to slip at take-off, the robot cannot rotate as much during the twist jump on the hardware compared to in simulation.

3) Continuous Jumping Optimization: By decreasing the time between successive jumps, we are able to accomplish continuous jumping. This can be done by increasing the frequency f_1, shown in Figure 2. However, with decreased time between jumps, the landing configuration becomes increasingly important in order for the next applied forces to be able to successfully have the same outcome between jumps. Several jumps can also be optimized together, where instead of running the optimization after each individual jump, we can let the robot jump several times and then compute optimal parameters that may work better for successive jumps rather than individual jumps. Another possibility would be adding the frequency of jumps f_1 to the optimization parameters.

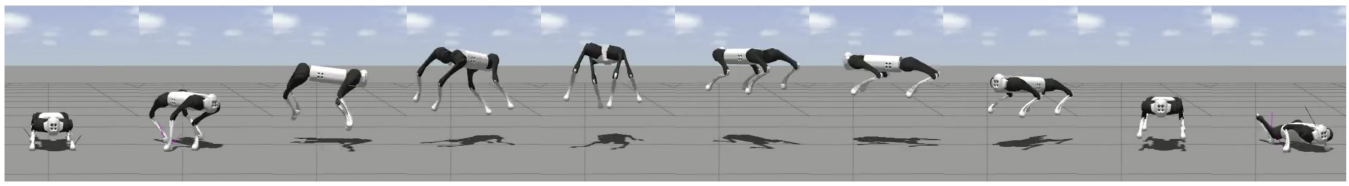

Fig. 8: Optimized twist-turn jump in Gazebo. The Unitree Go1 quadruped jumps 0.67 m high while rotating over 360 degrees.

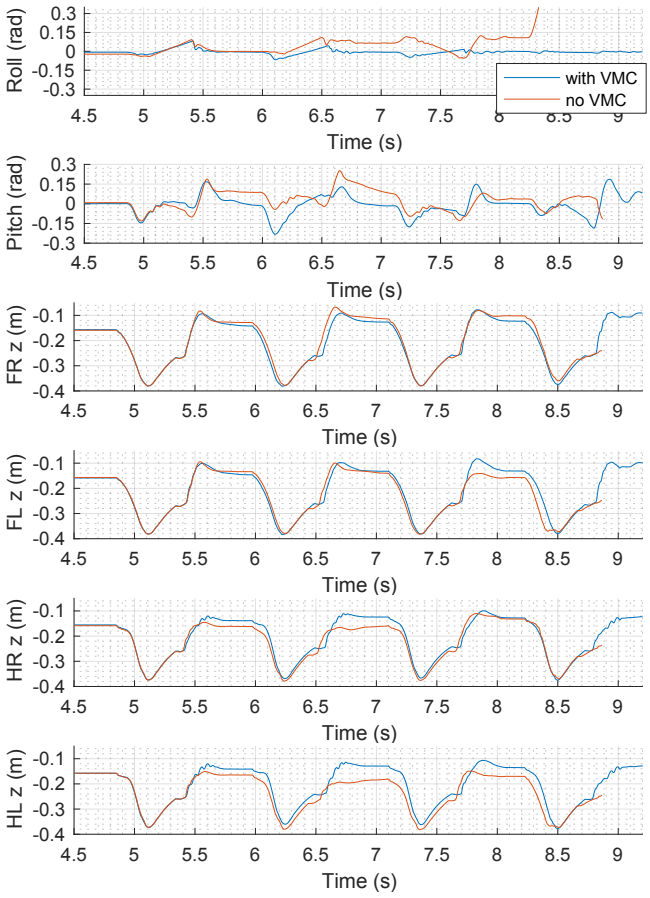

Fig. 9: Rough terrain adaptation with and without Virtual Model Control active. From top: (1) base roll, (2) base pitch, (3)-(6) Front Right (FR), Front Left (FL), Hind Right (HR), Hind Left (HL) foot z positions in the leg frame. The jump at 8.3 (s) without the VMC causes a fall due to the initial roll angle.

4) Rough Terrain Adaptation: We test the robustness of the continuous jumping controller by randomly placing overlapping 0.03 m foam blocks and weights on the floor. The foam blocks are very light and are easily kicked and moved by the robot, causing additional disturbances due to sliding between the robot feet and the floor. Our controller is robust to such disturbances for omnidirectional jumps, though as can be expected, the noise can cause the robot to not jump as far or turn as much as in nominal conditions. Snapshots are shown in Figure 1, and experiments can be seen in the video.

The Virtual Model Control block is especially important in such environments. We tested removing the VMC while jumping forward on the rough terrain, and after a few jumps, keeping the feet at their nominal heights in contact phase despite the uneven terrain caused a large initial roll angle during take-off, causing a fall (see video). This can be observed in Figure 9, where at 8 seconds, we see that there is non-zero roll without the VMC active (red line), and the foot positions do not compensate for this (left feet are still extended). This leads to the large roll take-off and subsequent flip and fall. In contrast, in the same scenario at 8 seconds, with the VMC active (blue line), we see that the left legs are retracted to keep the roll at zero, enabling a successful jump without falling. We could expect even better performance with a more sophisticated whole-body controller, which we plan to add in future work.

IV. Conclusion

In this paper, we have presented a control architecture for rapidly optimizing quadruped jumping skills directly on hardware. We designed and parameterized force profiles with only a few parameters, which can be rapidly optimized within just a few jumps. The force profiles are tracked at the joint level with the foot Jacobian, and Cartesian PD impedance control helps to keep the feet at a nominal position beneath the hips. We also added Virtual Model Control to help stabilize the jumping skills to allow for continuous jumping even on uneven terrain. We demonstrated omnidirectional skills including jumping forward (0.5 m in height, 0.5 m in distance), jumping laterally, and twist-turn jumps (over 2 rad).

There are several directions for future work. While VMC helped to stabilize the jumping motions, a whole-body controller could help with even better robustness. This could be either model-based, or trained with deep reinforcement learning as a feedback control policy on top of the force profiles, similar to recent work learning residual actions on top of a base jumping policy [30]. Another direction could be to integrate a landing controller to dampen the impact, for example as presented in [53] or [54], or study and combine the effects of passive or hybrid compliance components in parallel with the motors [55]. Lastly, it would be interesting to measure the ground reaction forces at the feet with force plates to verify and improve the application of the desired forces.

There is also opportunity for drawing connections to biology, for example comparisons with force profiles from different animal species to improve the parametrization and design. This could also be used to map the force profiles back to muscle activations to study mechanisms of flexor and extensor profiles that could be formed by the CPG. Our framework can also be used to study questions related to front limb damping on landing as shown by the biological study on frogs [43], and effects of sensory feedback on jumping and landing ability (i.e. the importance of the vestibular system (VMC or whole-body control)) [56].

REFERENCES

[1] C. D. Fitzgibbon and J. H. Fanshawe, "Stotting in thomson's gazelles: an honest signal of condition," *Behavioral Ecology and Sociobiology*, vol. 23, pp. 69–74, 1988.

[2] T. Caro, "The functions of stotting: a review of the hypotheses," *Animal Behaviour*, vol. 34, no. 3, pp. 649–662, 1986.

[3] 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.

[4] 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.

[5] 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.

[6] Q. Nguyen, M. J. Powell, B. Katz, J. D. Carlo, and S. Kim, "Optimized jumping on the mit cheetah 3 robot," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7448–7454.

[7] C. V. Nguyen and Q. Nguyen, "Contact-timing and trajectory optimization for 3d jumping on quadruped robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 11 994–11 999.

[8] B. Katz, J. Di 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)*. IEEE, 2019, pp. 6295–6301.

[9] M. Chignoli, S. Morozov, and S. Kim, "Rapid and reliable quadruped motion planning with omnidirectional jumping," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6621–6627.

[10] 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)*. IEEE, 2021, pp. 7693–7699.

[11] C. V. Nguyen, L. Bao, and Q. Nguyen, "Continuous jumping for legged robots on stepping stones via trajectory optimization and model predictive control," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 93–99.

[12] 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)*, 2018, pp. 1–9.

[13] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[14] M. Sombolestan, Y. Chen, and Q. Nguyen, "Adaptive force-based control for legged robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7440–7447.

[15] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2261–2268, 2018.

[16] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[17] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *arXiv preprint arXiv:2205.02824*, 2022.

[18] G. Bellegarda, Y. Chen, Z. Liu, and Q. Nguyen, "Robust high-speed running for quadruped robots via deep reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 364–10 370.

[19] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter, "Advanced skills by learning locomotion and local navigation end-to-end," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 2497–2503.

[20] 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*, 2022.

[21] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Robust and Versatile Bipedal Jumping Control through Reinforcement Learning," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[22] L. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and adapting agile locomotion skills by transferring experience," *arXiv preprint arXiv:2304.09834*, 2023.

[23] G. Bellegarda and K. Byl, "An online training method for augmenting mpc with deep reinforcement learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5453–5459.

[24] G. Bellegarda and K. Byl, "Training in task space to speed up and guide reinforcement learning," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 2693–2699.

[25] G. Bellegarda and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *arXiv preprint arXiv:2011.07089*, 2020.

[26] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. bae Kim, and P. Agrawal, "Learning to jump from pixels," in *5th Annual Conference on Robot Learning*, 2021.

[27] W. Yu, D. Jain, A. Escontrela, A. Iscen, P. Xu, E. Coumans, S. Ha, J. Tan, and T. Zhang, "Visual-locomotion: Learning to walk on complex terrains with vision," in *5th Annual Conference on Robot Learning*, 2021.

[28] K.-H. Lee, O. Nachum, T. Zhang, S. Guadarrama, J. Tan, and W. Yu, "Pi-ars: Accelerating evolution-learned visual-locomotion with predictive information representations," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 1447–1454.

[29] Y. Yang, G. Shi, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Cajun: Continuous adaptive jumping using a learned centroidal controller," *arXiv preprint arXiv:2306.09557*, 2023.

[30] Y. Yang, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Continuous versatile jumping using learned action residuals," in *Proceedings of The 5th Annual Learning for Dynamics and Control Conference*, ser. Proceedings of Machine Learning Research, N. Matni, M. Morari, and G. J. Pappas, Eds., vol. 211. PMLR, 15–16 Jun 2023, pp. 770–782.

[31] 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.

[32] G. Bellegarda and A. Ijspeert, "Visual CPG-RL: Learning central pattern generators for visually-guided quadruped navigation," *arXiv preprint arXiv:2212.14400*, 2022.

[33] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Manyquadrupeds: Learning a single locomotion policy for diverse quadruped robots," *arXiv preprint arXiv:2310.10486*, 2023.

[34] 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.

[35] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Deeptransition: Viability leads to the emergence of gait transitions in learning anticipatory quadrupedal locomotion skills," *arXiv preprint arXiv:2306.07419*, 2023.

[36] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Legged robots that keep on learning: Fine-tuning locomotion policies in the real world," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 1593–1599.

[37] 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.

[38] 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.

[39] 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.

[40] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty: An experimental comparison on a dynamic bipedal walker," *Annals of Mathematics and Artificial Intelligence*, vol. 76, pp. 5–23, 2016.

[41] 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, 2022.

[42] 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.
[43] S. Nauwelaerts and P. Aerts, "Take-off and landing forces in jumping frogs," *Journal of Experimental Biology*, vol. 209, no. 1, pp. 66–77, 2006.
[44] K. Matsuoka, "Mechanisms of frequency and pattern control in the neural rhythm generators," *Biological cybernetics*, vol. 56, no. 5, pp. 345–353, 1987.
[45] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *IEEE International Conference on Robotics and Automation*, 2008, pp. 819–824.
[46] 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.
[47] A. Spröwitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert, "Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot," *The International Journal of Robotics Research*, vol. 32, no. 8, pp. 932–950, 2013.
[48] 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.
[49] M. Ajallooeian, S. Gay, A. Tuleu, A. Spröwitz, and A. J. Ijspeert, "Modular control of limit cycle locomotion over unperceived rough terrain," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 3390–3397.
[50] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.
[51] Unitree Robotics. Go1. https://www.unitree.com/products/go1/.
[52] J. Fan, Q. Du, Z. Dong, J. Zhao, and T. Xu, "Design of the jump mechanism for a biomimetic robotic frog," *Biomimetics*, vol. 7, no. 4, p. 142, 2022.
[53] S. H. Jeon, S. Kim, and D. Kim, "Online optimal landing control of the mit mini cheetah," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 178–184.
[54] F. Roscia, M. Focchi, A. Del Prete, D. G. Caldwell, and C. Semini, "Reactive landing controller for quadruped robots," *arXiv preprint arXiv:2305.07748*, 2023.
[55] M. S. Ashtiani, A. Aghamaleki Sarvestani, and A. Badri-Spröwitz, "Hybrid parallel compliance allows robots to operate with sensorimotor delays and low control frequencies," *Frontiers in Robotics and AI*, vol. 8, p. 645748, 2021.
[56] S. Cox, L. Ekstrom, and G. Gillis, "The influence of visual, vestibular, and hindlimb proprioceptive ablations on landing preparation in cane toads," *Integrative and Comparative Biology*, vol. 58, no. 5, pp. 894–905, 2018.