Nonlinear Model Predictive Control for Quadrupedal Locomotion Using Second-Order Sensitivity Analysis

Dongho Kang, Flavio De Vincenti, Stelian Coros

***Abstract*— We present a versatile nonlinear model predictive control (NMPC) approach for quadrupedal locomotion. Our formulation jointly optimizes a base trajectory and a set of footholds over a finite time horizon based on simplified dynamics models. We leverage second-order sensitivity analysis and a sparse Gauss-Newton (SGN) method to solve the resulting optimal control problems. We further describe our ongoing effort to verify our approach through simulation and hardware experiments. Finally, we extend our locomotion framework to deal with challenging tasks that comprise gap crossing, movement on stepping stones, and multi-robot control.**

***Paper Type* – Original Work.**

I. Introduction

Model predictive control (MPC) is a powerful tool for enabling agile and robust locomotion skills on legged systems. Its capability of handling flying phases while rejecting disturbances enhances the maneuverability [1, 2] and mobility of quadruped robots [3].

Standard MPC implementations are required to solve finite-horizon optimal control problems (OCPs) at a real-time rate. This process comes with a high computational cost that defies online execution. Current existing MPC methods for quadrupedal locomotion tackle this challenge by thorough software designs and high-performance, parallel implementations [4, 5]; alternatively, they adopt simplified dynamics models to reduce computational burdens [1, 2, 6, 7].

In this paper, we present a versatile nonlinear MPC (NMPC) strategy that jointly optimizes a base trajectory and a set of footholds. We describe the system dynamics as a function of a control input vector evolving over a time horizon and a time-invariant set of footholds. We solve the resulting OCP using a second-order numerical solver [8] that leverages sensitivity analysis (SA) [9, 10, 11] to compute the exact values of the required derivatives efficiently. This approach significantly improves the robustness of the controller while ensuring real-time execution. Moreover, our formulation is easily adaptable to various nonlinear models and quadrupedal locomotion scenarios.

In the following sections, we provide the mathematical formulation of our method. Furthermore, we describe two examples based on different nonlinear dynamics models compatible with our framework. Finally, we present our preliminary results verifying our approach and applying it to generic locomotion control tasks.

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).

The authors are with the Computational Robotics Lab in ETH Zurich, Switzerland. {kangd, dflavio, scoros}@ethz.ch

The first two authors contributed equally to this work.

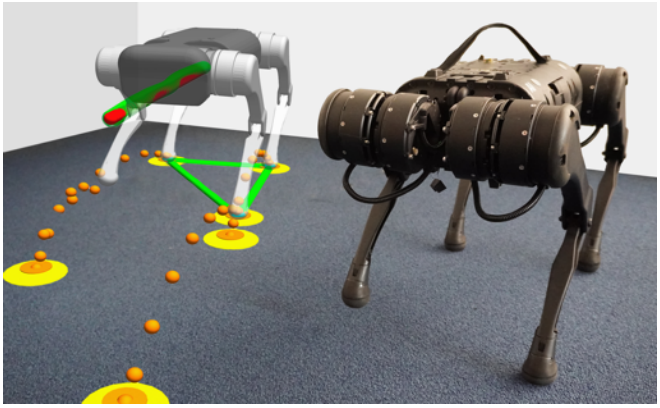

Fig. 1. Our MPC-based locomotion controller in action on a simulated *Unitree A1* robot **(left)** and a real one **(right)**. Given kinematically generated references (**red curve** and **yellow circles**), our planner generates optimal base trajectory (**green curve**) and footholds (**orange circles**) that are dynamically feasible.

II. Nonlinear MPC

In this section, we describe our optimal control framework based on second-order SA. Subsequently, we formulate a model-agnostic OCP for quadrupedal locomotion, where the optimization variables include system states and stepping locations. Finally, we provide two examples applying our formulation to nonlinear dynamics models; namely, the variable-height inverted pendulum and the single rigid body model.

A. Framework

We express the discrete-time dynamics of a system through an implicit function:

$$\mathbf{G}_k(\mathbf{x}_k,\, \mathbf{x}_{k+1}, \mathbf{u}_k,\, \mathbf{p}) = \mathbf{0}_n\,, \tag{1}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ and $\mathbf{u}_k \in \mathbb{R}^m$ denote the system state and control input vectors at time step k, respectively, $\mathbf{G}_k$ is a differentiable function capturing the system evolution at time step k, and $\mathbf{0}_n \in \mathbb{R}^n$ is the n-dimensional zero vector. In this formulation, the dynamics further depend on a time-invariant vector of parameters $\mathbf{p} \in \mathbb{R}^p$: while $\mathbf{u}_k$ affects the system only at time step k, $\mathbf{p}$ does so at all time steps. In our application to locomotion control, $\mathbf{p}$ represents a set of footholds to be stepped on over a time horizon (see Section II-B).

We define the stacked state and input vectors as

$$\mathbf{X} \coloneqq \begin{bmatrix}\mathbf{x}_1^\top & \mathbf{x}_2^\top & \dots & \mathbf{x}_N^\top\end{bmatrix}^\top,$$
$$\mathbf{U} \coloneqq \begin{bmatrix}\mathbf{u}_0^\top & \mathbf{u}_1^\top & \dots & \mathbf{u}_{N-1}^\top & \mathbf{p}^\top\end{bmatrix}^\top,$$

respectively, where N denotes the time horizon. Additionally, given a measurement $\mathbf{x}_0$ of the current state of the system, we define the stacked dynamics constraint function as:

$$\mathbf{G}(\mathbf{X}, \mathbf{U}) \coloneqq \begin{bmatrix} \mathbf{G}_0^\top & \mathbf{G}_1^\top & \dots & \mathbf{G}_{N-1}^\top \end{bmatrix}^\top . \tag{2}$$

Then, we can define a finite-horizon OCP for the system (1) as:

$$\begin{aligned} \min_{\mathbf{X},\,\mathbf{U}} \quad & \mathcal{J}(\mathbf{X},\,\mathbf{U}) \\ \text{s.t.} \quad & \mathbf{G}(\mathbf{X}, \mathbf{U}) = \mathbf{0}\,, \end{aligned} \tag{3}$$

where $\mathcal{J}(\mathbf{X},\,\mathbf{U})$ is a cost function that depends on the stacked state and input vectors. We implement state and input constraints through penalty terms added to $\mathcal{J}(\mathbf{X},\,\mathbf{U})$.

If an explicit function $\mathbf{g}_k$ such that $\mathbf{x}_{k+1} = \mathbf{g}_k(\mathbf{x}_k,\,\mathbf{u}_k,\,\mathbf{p})$ is available, then we can define $\mathbf{G}_k \coloneqq \mathbf{x}_{k+1} - \mathbf{g}_k(\mathbf{x}_k,\,\mathbf{u}_k,\,\mathbf{p})$ to adapt the system dynamics equation to the form (1). However, we note that (1) is general enough for the cases where an explicit form of the dynamics equations does not exists. For instance, if the backward Euler time integration method is adopted, the system dynamics may not be made explicit. Under mild assumptions, (1) implies that there is a map between $\mathbf{X}$ and $\mathbf{U}$, i.e., $\mathbf{X}(\mathbf{U})$ although the map may not have an analytic form. Therefore, we can convert (3) into the following unconstrained minimization problem:

$$\min_{\mathbf{U}} \quad \mathcal{J}(\mathbf{X}(\mathbf{U}),\,\mathbf{U})\,. \tag{4}$$

We find the optimal control inputs and parameters $\mathbf{U}^*$ minimizing the cost function of $\mathbf{U}$. Even if an analytical expression of $\mathbf{X}(\mathbf{U})$ does not exist, we can perform such optimization using a second-order method through *sensitivity analysis* [10, 9, 11]. SA allows us to compute the exact values of the first and the second derivatives efficiently:

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}\mathbf{U}} = \frac{\partial \mathcal{J}}{\partial \mathbf{X}}\mathbf{S} + \frac{\partial \mathcal{J}}{\partial \mathbf{U}}\,, \tag{5a}$$

$$\frac{\mathrm{d}^2\mathcal{J}}{\mathrm{d}\mathbf{U}^2} = \left(\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{U}}\frac{\partial \mathcal{J}}{\partial \boldsymbol{X}}\right)\mathbf{S} + \frac{\partial \mathcal{J}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\mathbf{S}}{\mathrm{d}\boldsymbol{U}} + \frac{\mathrm{d}}{\mathrm{d}\boldsymbol{U}}\frac{\partial \mathcal{J}}{\partial \boldsymbol{U}} \tag{5b}$$

$$\approx \mathbf{S}^\top \frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{X}^2}\mathbf{S} + \mathbf{S}^\top \frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{U}\partial \boldsymbol{X}} + \frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{X}\partial \boldsymbol{U}}\mathbf{S} + \frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{U}^2}\,, \tag{5c}$$

where *the sensitivity matrix* $\mathbf{S}$ can be evaluated as follows:

$$\mathbf{S} = -\left(\frac{\partial \mathbf{G}}{\partial \boldsymbol{X}}\right)^{-1}\frac{\partial \mathbf{G}}{\partial \boldsymbol{U}}\,. \tag{6}$$

For full derivation, we refer the reader to the technical note [9]. We note that the generalized Gauss–Newton approximation (5c) can be employed in place of the Hessian (5b) to reduce the computational burden and to guarantee the semi-positive definiteness of the second derivative.

B. Quadrupedal Locomotion Control

We formulate an OCP for quadrupedal locomotion using the framework described in Section II-A. The main objective is to track a reference base trajectory generated from a user's commands. Thus, we define the following cost function on the base positions $\mathbf{r}_k$ over a time horizon N:

$$\mathcal{J}(\mathbf{X},\,\mathbf{U}) \coloneqq \sum_{k=0}^{N} \|(\mathbf{r}_{k+1} - \mathbf{r}_k) - (\mathbf{r}_{k+1}^{\mathrm{ref}} - \mathbf{r}_k^{\mathrm{ref}})\|_2^2 \tag{7a}$$

$$+ \sum_{k=0}^{N} \|h_{k+1} - h_{k+1}^{\mathrm{ref}}\|_2^2 \tag{7b}$$

$$+ \sum_{i=1}^{p} \sum_{j=i+1}^{\min(p,i+3)} K_1 \|(\mathbf{s}^i - \mathbf{s}^j) - (\mathbf{s}^{\mathrm{ref},i} - \mathbf{s}^{\mathrm{ref},j})\|_2^2 \tag{7c}$$

$$+ \mathcal{R}_{\mathrm{model}}(\mathbf{X},\,\mathbf{U})\,, \tag{7d}$$

The term (7a) penalizes base velocity tracking errors, (7b) penalizes base height tracking errors, (7c) regularizes the displacements between adjacent stepping locations, and finally (7d) is a model specific cost term. The variable $\mathbf{r}_k$ are parts of the system state vector $\mathbf{x}_k$, and $\mathbf{s}^i$ is a part of the parameter vector $\mathbf{p}$.

Equation (7c) with weighting coefficient K_1 regularizes the foothold optimization towards kinematically feasible solutions. The reference footholds $\mathbf{s}^{\mathrm{ref},i}$ are determined based on a simple *impact-to-impact* method whereby support feet lie below the corresponding hip in the middle of the stance phase [12]. We note that the term (7c) only penalizes relative positions between stepping locations, thus making the corresponding support polygons loosely resemble the reference ones [13].

C. Examples

We briefly describe two nonlinear systems, namely the variable-height inverted pendulum (IPM) and the single rigid body (SRBM) models, and we show how they can be integrated into our framework.

When possible, we discretize the continuous dynamics by employing a *semi-implicit Euler method*; given $\mathbf{r}_k$ and $\mathbf{r}_{k-1}$, we approximate the velocity at time steps k and $k+1$, respectively, as $\dot{\mathbf{r}}_k \approx (\mathbf{r}_k - \mathbf{r}_{k-1})/\Delta t$ and $\dot{\mathbf{r}}_{k+1} \approx \dot{\mathbf{r}}_k + \ddot{\mathbf{r}}_k \Delta t$, where $\ddot{\mathbf{r}}_k$ can be computed using a model-specific dynamics equation:

$$\begin{aligned} \mathbf{r}_{k+1} &\approx \mathbf{r}_k + \dot{\mathbf{r}}_{k+1}\Delta t \\ &\approx \mathbf{r}_k + \dot{\mathbf{r}}_k \Delta t + \ddot{\mathbf{r}}_k \Delta t^2 \\ &\approx 2\mathbf{r}_k - \mathbf{r}_{k-1} + \ddot{\mathbf{r}}_k \Delta t^2 \\ &= 2\mathbf{r}_k - \mathbf{r}_{k-1} \\ &\qquad + \boldsymbol{f}_{\mathrm{model}}(\mathbf{r}_k,\,\mathbf{u}_k,\,\mathbf{s}^{i_1},\,\mathbf{s}^{i_2},\,\dots,\,\mathbf{s}^{i_{|\sigma_k|}})\Delta t^2 \\ &=: \mathbf{g}_{\mathrm{model},k}\left(\mathbf{r}_{k-1},\,\mathbf{r}_k,\,\mathbf{u}_k,\,\mathbf{s}\right)\,, \end{aligned} \tag{8}$$

where σ_k denotes the subset of the stance foot positions at time step k, and $\mathbf{s}^{i_j} \in \sigma_k,\, \forall j \in \{1,\,2,\,\dots,\,|\sigma_k|\}$. The makeup of the control input vector $\mathbf{u}_k$ depends on the model and we will introduce it in due time.

In the following subsections, we define an explicit function $\boldsymbol{f}_{\mathrm{model}}$ for the IPM and the SRBM. As mentioned in Section II-A, we define $\mathbf{G}_k \coloneqq \mathbf{x}_{k+1} - \mathbf{g}_k(\mathbf{x}_k,\,\mathbf{u}_k,\,\mathbf{p})$ since an explicit expression of the system dynamics equation exists.

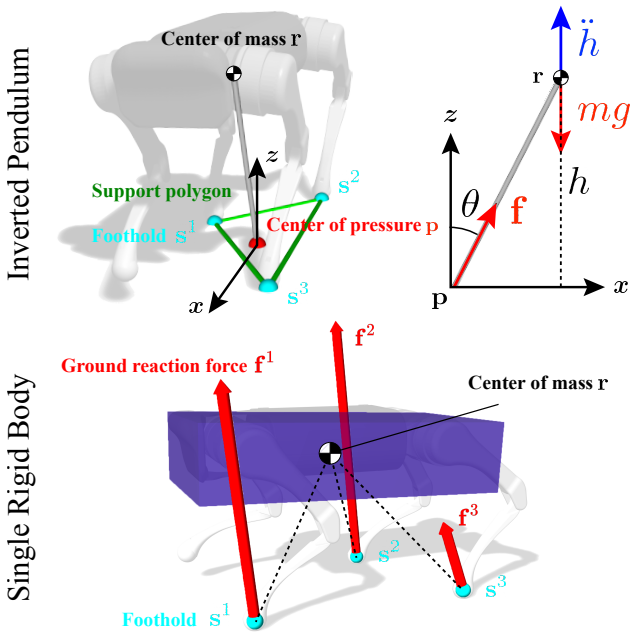

Fig. 2. A quadrupedal robot represented as an inverted pendulum **(top)**, and a single rigid body **(bottom)**.

1) Inverted Pendulum Model: The inverted pendulum model represents a legged robot as a point mass m concentrated at the center of gravity of the system $\mathbf{r}$ and a massless telescoping rod in contact with a flat ground. We assume that the contact point of the rod is at the center of pressure (CoP) of the robot $\mathbf{p}$, i.e., the location at which the resultant ground reaction force vector $\mathbf{f}$ would act if it were considered to have a single point of application [14]. The CoP always exists inside the support polygon of all stance foot positions $\mathbf{s}^i \in \mathbb{R}^3$. Thus, we can express its position with respect to an inertial reference frame as a convex combination

$$\mathbf{p} = \sum_{\mathbf{s}^i \in \sigma} w^i \, \mathbf{s}^i, \tag{9}$$

where σ is the set of the stance foot positions, and $w^i \in \mathbb{R}_{\geq 0}$ is a non-negative scalar weight corresponding to $\mathbf{s}^i$ that satisfy $\sum_i w^i = 1$.

For brevity, we omit the derivation of the following equation of motion for the IPM[1]:

$$\begin{aligned} \ddot{\mathbf{r}} &= (\mathbf{r} - \sum_{\mathbf{s}^i \in \sigma} w^i \, \mathbf{s}^i) \frac{\ddot{h} + \|\mathbf{g}\|_2}{r_z} + \mathbf{g} \\ &=: \boldsymbol{f}_{\text{IPM}}(\mathbf{r}, \, \mathbf{u}, \, \mathbf{s}^{i_1}, \, \mathbf{s}^{i_2}, \, \ldots, \, \mathbf{s}^{i_{|\sigma|}}) \,, \end{aligned} \tag{10}$$

with control input vector $\mathbf{u} := \begin{bmatrix} \ddot{h} \; w^{i_1} \; w^{i_2} \; \ldots \; w^{i_{|\sigma|}} \end{bmatrix}^\top$, and parameters $\mathbf{s}^{i_j} \in \sigma, \, \forall j \in \{1, \, 2, \, \ldots, \, |\sigma|\}$.

Furthermore, we define the model specific cost term (7d) for the IPM as follows:

$$\mathcal{R}_{\text{IPM}} =: \sum_{k=0}^{N-1} \left(\frac{K_2}{2} \| 1 - \sum_i w_k^i \|_2^2 + \sum_i \mathcal{S}_{\geq 0}(w_k^i) \right), \tag{11}$$

[1]The full derivation of the IPM is available in our recent work [15].

where K_2 is a weighting coefficient and $\mathcal{S}_{\geq r} \colon \mathbb{R} \to \mathbb{R}_{\geq 0}, \, \forall r \in \mathbb{R}$ is a $\mathcal{C}^2$-continuous function following [16, eq. (8)] with unitary stiffness and $\epsilon = 0.1$. This term enforces the constraint $\sum_i w^i = 1$, and the non-negativity of the weights as soft constraints.

2) Single Rigid Body Model: If the limbs of a robot are significantly lightweight compared to its body, then we can neglect their inertial effects and reduce the system to a single rigid body with mass m and body frame moment of inertia ${}^B\mathbf{I} \in \mathbb{R}^{3\times 3}$.

The position $\mathbf{r} \in \mathbb{R}^3$ and unit quaternion $\mathbf{q} \in \mathbb{S}^3$ defines the pose of the lumped rigid body. ${}^B\boldsymbol{\omega}$ denotes its angular velocity vector expressed in body coordinates, and $\mathbf{f}^i \in \mathbb{R}^3$ denotes the ground reaction force associated with the stepping location $\mathbf{s}^i \in \sigma$. Then, we can write the translational and rotational dynamics of the system as:

$$\ddot{\mathbf{r}} = \frac{1}{m} \sum_{\mathbf{s}^i \in \sigma} \mathbf{f}^i + \mathbf{g} =: \boldsymbol{f}_{\text{SRBM,t}}(\mathbf{u}) \,, \tag{12}$$

$$\begin{aligned} {}^B\dot{\boldsymbol{\omega}} &= {}^B\mathbf{I}^{-1} \left[\mathbf{R}(\mathbf{q})^\top \sum_{\mathbf{s}^i \in \sigma} (\mathbf{s}^i - \mathbf{r}) \times \mathbf{f}^i - {}^B\boldsymbol{\omega} \times {}^B\mathbf{I}\,{}^B\boldsymbol{\omega} \right] \\ &=: \boldsymbol{f}_{\text{SRBM,r}}(\mathbf{r}, \, \mathbf{q}, \, {}^B\boldsymbol{\omega}, \, \mathbf{u}, \, \mathbf{s}^{i_1}, \, \mathbf{s}^{i_2}, \, \ldots, \, \mathbf{s}^{i_{|\sigma|}}) \,, \end{aligned} \tag{13}$$

where $\mathbf{R}(\mathbf{q})$ is the rotation matrix corresponding to $\mathbf{q}$, and $\mathbf{u} := \begin{bmatrix} \mathbf{f}^{i_1} \; \mathbf{f}^{i_2} \; \ldots \; \mathbf{f}^{i_{|\sigma|}} \end{bmatrix}^\top$ is the control input vector.

We employ a semi-implicit Euler method similar to the one outlined in Section II-C. However, to integrate the orientation dynamics, we employ a forward Lie-group Euler method which allows us to preserve the unitary norm constraint of unit quaternions. Specifically, we approximate the body frame angular velocity at time step k and $k+1$, respectively, as ${}^B\boldsymbol{\omega}_k \approx 2 \, \mathfrak{Im}(\bar{\mathbf{q}}_{k-1} * \mathbf{q}_k)/\Delta t$ and ${}^B\boldsymbol{\omega}_{k+1} \approx {}^B\boldsymbol{\omega}_k + {}^B\dot{\boldsymbol{\omega}}_k \Delta t$, where $\mathfrak{Im}(\mathbf{q})$ extracts the imaginary part of $\mathbf{q}$, $\bar{\mathbf{q}}$ is the conjugate of $\mathbf{q}$, $*$ is the quaternion multiplication operator, and ${}^B\dot{\boldsymbol{\omega}}_k$ can be computed using (13). Then, our integration scheme for unit quaternions translates to $\mathbf{q}_{k+1} \approx \mathbf{q}_k * \exp\left({}^B\boldsymbol{\omega}_{k+1} \Delta t\right)$, where $\exp \colon \mathbb{R}^3 \to \mathbb{S}^3$ is a Lie-group exponential function which, for unit quaternions, has the following closed form:

$$\exp(\mathbf{v}) := \cos\left(\frac{1}{2}\|\mathbf{v}\|\right) + \frac{\mathbf{v}}{\|\mathbf{v}\|} \sin\left(\frac{1}{2}\|\mathbf{v}\|\right).$$

Using the equations above, we can finally write the SRBM dynamics in the form (8) as:

$$\begin{aligned} \begin{bmatrix} \mathbf{r}_{k+1} \\ \mathbf{q}_{k+1} \end{bmatrix} &\approx \begin{bmatrix} 2\mathbf{r}_k - \mathbf{r}_{k-1} + \boldsymbol{f}_{\text{SRBM,t}}(\mathbf{u}_k) \Delta t^2 \\ \mathbf{q}_k * \exp\left({}^B\boldsymbol{\omega}_{k+1} \Delta t\right) \end{bmatrix} \\ &:= \mathbf{g}_{\text{SRBM},k}(\mathbf{r}_{k-1}, \, \mathbf{r}_k, \, \mathbf{q}_{k-1}, \, \mathbf{q}_k, \, \mathbf{u}_k, \, \mathbf{s}) \,. \end{aligned} \tag{14}$$

Following (7d), we can define a cost term for the SRBM penalizing deviations from a reference orientation trajectory $\mathbf{q}_k^{\text{ref}}$ [17] and imposing non-negative vertical components of the GRFs[2]:

$$\mathcal{R}_{\text{SRBM}} =: \sum_{k=0}^{N-1} \left(1 - \left| \mathbf{q}_k^\top \mathbf{q}_k^{\text{ref}} \right| + \sum_i \mathcal{S}_{\geq 0}(\mathbf{f}_{z,k}^i) \right). \tag{15}$$

[2]For our preliminary results, we do not include friction cone soft constraints to (15) to keep our implementation as simple as possible.

Fig. 3. We pushed **(left)** and disturbed the *Unitree A1* robot by putting a plate under its feet **(right)** while it was performing a trot gait.

III. Experiments

We present a series of simulation and hardware experiments we conducted to verify the efficacy of our approach. The results we discuss in this section were attained using the IPM described in section II-C.1. The footage of the experiments is available in the supplementary video[2], along with preliminary results achieved with the SRBM. In all our experiments, the optimal base trajectories output by the MPC scheme were tracked by a quadratic programming-based whole-body controller [10].

Firstly, we tested the robustness of our controller for locomotion on flat terrains using the *Unitree A1* robot. We hindered the robot while it was trotting in place, as portrayed in the snapshots in Figure 3. In our tests, the robot was able to withstand unexpected disturbances and successfully recover its stability. We demonstrate in the accompanying video how the foothold optimization improves the capability of the system to resist large lateral pushes.

We show the versatility of our foothold optimization approach to adapt a gait to different terrain types, namely a *gap crossing* and a *stepping stones* scenarios. The former setting consists of a sequence of rifts with different widths; the latter comprises a grid of stepping stones distant $20\,\mathrm{cm}$ from each other the robot must step on – see Figure 4. We add the following terms to the objective function (7) to model each gap and stepping stone, respectively:

$$\mathcal{L}_{\mathrm{gc}}(\mathbf{s}) \coloneqq \sum_{i=1}^{p} \mathcal{S}_{\geq g}(|s_x^i - g_x|)\,, \tag{16}$$

$$\mathcal{L}_{\mathrm{ss}}(\mathbf{s}) \coloneqq \sum_{i=1}^{p} -K_3 \exp\left\{-\frac{1}{2}\frac{\|\mathbf{s}^i - \mathbf{t}\|_2^2}{K_4^2}\right\}, \tag{17}$$

where g and g_x are the gap half width and x-position, respectively, $\mathbf{t}$ is the stepping stone location, and K_3 and K_4 are tuning parameters. To model the stepping stones in (17), we employ a negative Gaussian function centered at the corresponding positions; in this way, we incentivize nearby stepping locations to converge towards the closest footholds. As shown in the supplementary video, these simple penalty terms are sufficient to ensure that the associated constraints are almost never violated. The occasional missteps may be avoided through a careful tuning of (16) and (17), or by designing some fallback control strategies.

[2]The video is available in https://youtu.be/BrJSRlAJaX4.

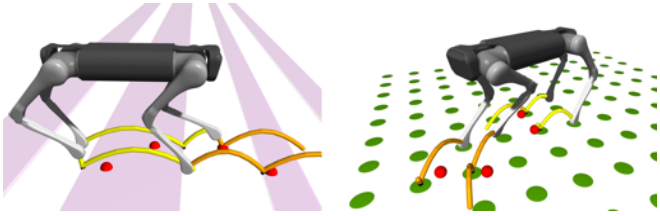

Fig. 4. Snapshots of simulation experiments for the gap crossing and stepping stone scenarios with the *Aliengo* robot. The red spheres denote the reference stepping locations, while the **green** ones represent stepping stones. The black trajectories are the outputs of our MPC controller. The resulting footstep placements deviate considerably from the corresponding references and allow the robot to avoid $32\,\mathrm{cm}$ wide gaps **(left)** and step on isolated footholds **(right)**.

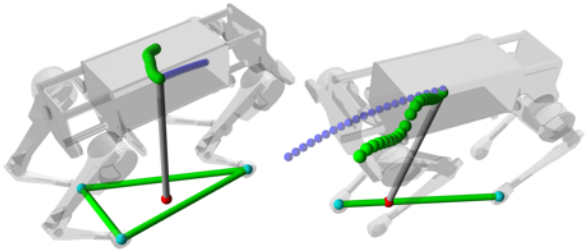

Fig. 5. Two *Laikago* quadruped robots controlled using a centralized MPC strategy in simulation. The robot on the left is executing a walking gait, whereas the one on the right is performing a flying trot. The reference trajectories are denoted by blue spheres, and the optimized ones are represented by green spheres: the latter deviate significantly from the former to prevent the robots from colliding.

Finally, we extend our framework so that the state vector contains the positions of two robots, and we couple the solutions for the two subsystems by adding the following collision avoidance term to the objective function:

$$\mathcal{L}_{\mathrm{ca}}(\mathbf{X}) \coloneqq \sum_{k=0}^{N} \mathcal{S}_{\geq 1}(\|\mathbf{r}_k^a - \mathbf{r}_k^b\|_2)\,,$$

where $\mathbf{r}_k^a$ and $\mathbf{r}_k^b$ are the states of the two robots at time step k, respectively. This cost term ensures that the robots keep a distance of at least $1\,\mathrm{m}$ from each other – see Figure 5. As shown in the supplementary video, our NMPC framework is able to control the multi-robot system in real time by solving a single OCP.

IV. Conclusion and Future Work

Our NMPC scheme facilitates the implementation of robust controllers for various quadrupedal locomotion tasks. We can easily integrate different nonlinear dynamics models into our framework. We leave a complete demonstration with different models and more comprehensive analysis for future work. Our immediate next step is to verify our formulation of the SRBM and test it on hardware. Furthermore, we intend to compare our method to other state-of-the-art nonlinear control frameworks.

REFERENCES

[1] 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," *ArXiv*, vol. abs/1909.06586, 2019.

[2] Y. Ding, A. Pandala, and H. Park, "Real-time model predictive control for versatile dynamic motions in quadrupedal robots," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8484–8490, 2019.

[3] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Fast, robust quadruped locomotion over challenging terrain," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2665–2670.

[4] F. Farshidian, E. Jelavic, A. Satapathy, M. Giftthaler, and J. Buchli, "Real-time motion planning of legged robots: A model predictive control approach," *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 577–584, 2017.

[5] M. Neunert, M. Stäuble, M. Giftthaler, C. D. Bellicoso, J. Carius, C. Gehring, M. Hutter, and J. Buchli, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1458–1465, 2018.

[6] J. D. Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, 2018.

[7] G. Bledt and S. Kim, "Implementing regularized predictive control for simultaneous real-time footstep and ground reaction force optimization," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6316–6323, 2019.

[8] J. Zehnder, S. Coros, and B. Thomaszewski, "Sgn: Sparse gauss-newton for accelerated sensitivity analysis," *ArXiv*, vol. abs/2107.03285, 2021.

[9] S. Zimmermann, R. Poranne, and S. Coros, "Optimal control via second order sensitivity analysis," *arXiv: Optimization and Control*, 2019.

[10] F. De Vincenti, D. Kang, and S. Coros, "Control-aware design optimization for bio-inspired quadruped robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2021)*. IEEE, 2021.

[11] J. M. Bern, P. Banzet, R. Poranne, and S. Coros, "Trajectory optimization for cable-driven soft robot locomotion." in *Robotics: Science and Systems*, vol. 1, no. 3, 2019.

[12] F. Yin, A. Tang, L. Xu, Y. Cao, Y. Zheng, Z. Zhang, and X. Chen, "Run like a dog: Learning based whole-body control framework for quadruped gait style transfer," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 8508–8514.

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

[14] P. Cavanagh, "A technique for averaging center of pressure paths from a force platform." *Journal of biomechanics*, vol. 11 10-12, pp. 487–91, 1978.

[15] D. Kang, F. De Vincenti, N. C. Adam, 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 2022)*. IEEE, 2022, (under review).

[16] J. M. Bern, K.-H. Chang, and S. Coros, "Interactive design of animated plushies," *ACM Transactions on Graphics (TOG)*, vol. 36, pp. 1 – 11, 2017.

[17] B. E. Jackson, K. Tracy, and Z. Manchester, "Planning with attitude," *IEEE Robotics and Automation Letters*, vol. 6, pp. 5658–5664, 2021.