Planning Natural Locomotion for Articulated Soft Quadrupeds

Pollayil, Mathew Jose ; Della Santina, C.; Mesesan, George ; Englsberger, Johannes ; Seidel, Daniel ; Garabini, Manolo; Ott, Christian ; Bicchi, Antonio; Albu-Schaffer, Alin

DOI
10.1109/ICRA46639.2022.9812416

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the International Conference on Robotics and Automation (ICRA 2022)

2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

Planning Natural Locomotion for Articulated Soft Quadrupeds

Mathew Jose Pollayil[1], Cosimo Della Santina[2,3], George Mesesan[3], Johannes Englsberger[3], Daniel Seidel[3], Manolo Garabini[1], Christian Ott[3], Antonio Bicchi[1], and Alin Albu-Schäffer[3,4]

***Abstract*— Embedding elastic elements into legged robots through mechanical design enables highly efficient oscillating patterns that resemble natural gaits. However, current trajectory planning techniques miss the opportunity of taking advantage of these natural motions. This work proposes a locomotion planning method that aims to unify traditional trajectory generation with modal oscillations. Our method utilizes task-space linearized modes for generating center of mass trajectories on the sagittal plane. We then use nonlinear optimization to find the gait timings that match these trajectories within the Divergent Component of Motion planning framework. This way, we can robustly translate the modes-aware centroidal motions into joint coordinates. We validate our approach with promising results and insights through experiments on a compliant quadrupedal robot.**

I. Introduction

Compliant legged robots [1]–[3] are being increasingly used as part of the effort to match the mobility skills of living beings [4]. Exploiting their inherent elastic motions promises more natural movements and efficiency [4], [5]. Certainly, by taking inspiration from muscles, tendons, and other soft tissues, introducing elastic elements in the mechanics can promote energy-saving and high-speed locomotion [6], [7].

This notwithstanding, such trend does not tally with a corresponding attention towards locomotion planning techniques for legged robots with elasticity: in particular, efficient motion planning for compliant walkers remains to date an under-explored topic. Few authors have addressed the problem of finding motions to locomote following the natural dynamics of the robotic system and with minimal reliance on feedback: for instance, see [8]–[12]. Data-driven approaches have also been applied successfully with the aim of learning to store and release elastic energy in the legs [13], [14]. However, all these works have the common limitation of tackling the problem only from a control perspective and never actually attempting to generate physically feasible trajectories that comply with the natural system motions of elastic robots. Perhaps, this is due to the mismatch between the complexity of such systems and the simplification commonly required by motion planning.

*This work has received funding under the European Union's Horizon 2020 Research and Innovation Programme from the European Research Council (ERC) (M-Runners - Grant Agreement No. 835284 and NatDyReL - Grant Agreement No. 819358), from Project THING - Grant Agreement No. 780883, from Project NI - Grant Agreement No. 101016970, and from Funded Project DYSTURBANCE - Grant Agreement No. 779963 (EUROBENCH). The authors thank G. J. Pollayil, X. Meng, D. Calzolari, and A. Schmidt for their help and fruitful discussions.

[1]Reasearch Center "E. Piaggio", Università di Pisa, 56122 Pisa, Italy. (email: mathewjose.pollayil@phd.unipi.it) [2]Cognitive Robotics Department, Delft University of Technology (TU Delft), 2628 Delft, The Netherlands. [3]Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Wessling, Germany. [4]Department of Informatics, Technical University of Munich (TUM), 85748 Garching, Germany

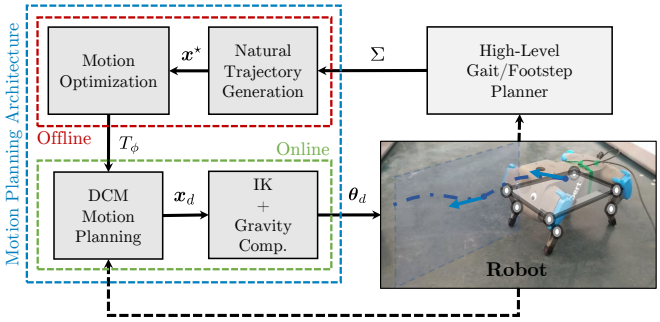


Fig. 1. We propose a motion planning architecture for planar articulated soft quadrupeds that merges traditional motion planning with linearized modal oscillations to generate CoM trajectories coherent with the natural dynamics.

Indeed, traditional motion generation strongly reduces system dynamics through template models [15]. This has the advantage of taming the complexity of planning. Examples of templates are the *Linear Inverted Pendulum Model* (LIPM) [16], or the *Spring-Loaded Inverted Pendulum* (SLIP) [17] - all of which have been successfully used in locomotion planning. Nevertheless, using templates may deprive planners of necessary information about the mechanical system - such as its compliance, its multi-body dynamics, and its nonlinear natural oscillations. These oscillations can be studied with nonlinear modal theory [18]–[21]. Some preliminary applications to locomotion have been proposed (e.g. [22], [23]). But - at least at their current stage - these mathematical descriptions are too complex and not yet sufficiently understood to be directly applied to general quadrupedal locomotion.

A strategy that sits in between these two extremes may potentially combine the capability of the template-based planners to manage complex systems and the power of modal analysis to exploit the robot's natural dynamics. This paper aims to provide an example of such an architecture (Fig. 1). We build upon DCM-based planning framework [24] - of which this work represents the first implementation on a quadrupedal robot - so as to optimally select some free parameters for maximally matching the natural dynamics of the robot. Our planner can generate joint trajectories that produce smooth and natural center of mass (CoM) evolutions, which are then implemented by low level feedforward controllers. We validate our approach on the articulated soft quadruped DLR Bert [1] (Fig. 2).

II. Overall Map of the Proposed Framework

We give a general overview of the proposed architecture, as shown in Fig. 1 and summarized by Algorithm 2. In the following sections, the building blocks will be explained more in detail. First, modal oscillations of the linearized

2022 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9681-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICRA46639.2022.9812416

systems around planned stances equilibria are combined to obtain natural CoM trajectories (Sec. III). We then use non-linear motion optimization to find the optimal gait timings to match the DCM-based CoM trajectory to the natural motion given by the modal oscillations (Sec. V). Such motion is then made physically feasible by using the DCM motion planning framework [25] (Sec. IV) using the optimized timings. The plans returned by our architecture can be implemented with any low level controller. Here, we use feedforward inverse kinematics (IK) augmented with gravity compensation (Sec. VI). Note that the planner's input - a sequence of stances Σ and step heights Δh - can be evaluated by standard foot-step planners (e.g. [26], [27]).

III. Simplified Modal Evolution

This section discusses the automatic generation of a CoM evolution composed by interconnecting linear modal oscillations. This part corresponds to "Natural Trajectory Generation" in Fig. 1 and Find Natural Trajectory in Algorithm 2.

A. Modal Oscillations of Floating-Base Robots

Consider the elastic model of a compliantly actuated floating-base robot

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) &= \boldsymbol{S}^\intercal \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}_j) + \boldsymbol{J}_c^\intercal(\boldsymbol{q})\boldsymbol{\lambda}, \\ \boldsymbol{B}\ddot{\boldsymbol{\theta}} + \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}_j) &= \boldsymbol{\tau}. \end{aligned} \tag{1}$$

Here, $\boldsymbol{q} = \begin{bmatrix}\boldsymbol{q}_b^\intercal & \boldsymbol{q}_j^\intercal\end{bmatrix}^\intercal \in \mathbb{R}^n$ is the vector of generalized coordinates of the robotic system, $\boldsymbol{q}_b \in \mathbb{R}^{n_b}$ is the vector of virtual base coordinates, and $\boldsymbol{q}_j \in \mathbb{R}^{n_j}$ is the vector of link positions of the legs, with $n = n_b + n_j$. We denote by $\boldsymbol{\theta} \in \mathbb{R}^{n_j}$ the vector of motor positions, by $\boldsymbol{\tau} \in \mathbb{R}^{n_j}$ the motor torques vector, and by $\boldsymbol{\lambda} \in \mathbb{R}^{n_c}$ the contact forces vector, where n_c is the number of constraints imposed by contacts. $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ is the inertia matrix, $\boldsymbol{C} \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal matrix, $\boldsymbol{g} \in \mathbb{R}^n$ is the gravity vector, $\boldsymbol{J}_c \in \mathbb{R}^{n_c\times n}$ is the contact Jacobian, $\boldsymbol{B} \in \mathbb{R}^{n_j\times n_j}$ is the projected motor inertia matrix, $\boldsymbol{K} \in \mathbb{R}^{n_j\times n_j}$ is the joint stiffness matrix, and $\boldsymbol{S} = [\boldsymbol{O}_{n_j\times n_b} \;\; \boldsymbol{I}_{n_j\times n_j}] \in \mathbb{R}^{n_j\times n}$ is the selection matrix.

Assuming $n_c \leq n_j$, according to the size of the contact constraint $\boldsymbol{J}_c\dot{\boldsymbol{q}} = \boldsymbol{0}$, we partition the generalized coordinates $\boldsymbol{q}$ into independent and dependent variables, $\boldsymbol{q}_I \in \mathbb{R}^{n-n_c}$ and $\boldsymbol{q}_D \in \mathbb{R}^{n_c}$, respectively. It is noteworthy that, with $n_c \leq n_j$, the independent variables $\boldsymbol{q}_I$ contain the base coordinates $\boldsymbol{q}_b$, which can be chosen to be a parametrization of the CoM position $\boldsymbol{x} = (x_1 \;\; x_2 \;\; x_3)^\intercal$ and orientation without loss of generality.

Subsequently, we use the *Lagrange-D'Alembert Method* to solve the constraint

$$\underbrace{\begin{bmatrix}\boldsymbol{J}_{c,I} & \boldsymbol{J}_{c,D}\end{bmatrix}}_{\boldsymbol{J}_c(\boldsymbol{q})} \underbrace{\begin{bmatrix}\dot{\boldsymbol{q}}_I \\ \dot{\boldsymbol{q}}_D\end{bmatrix}}_{\dot{\boldsymbol{q}}} = \boldsymbol{0} \;\rightarrow\; \dot{\boldsymbol{q}}_D = \underbrace{-\boldsymbol{J}_{c,D}^{-1}\boldsymbol{J}_{c,I}}_{\boldsymbol{T}_{D,I}}\dot{\boldsymbol{q}}_I.$$

We re-write the generalized coordinates as a function of the independent ones:

$$\dot{\boldsymbol{q}} = \underbrace{\begin{bmatrix}\boldsymbol{I} \\ \boldsymbol{T}_{D,I}\end{bmatrix}}_{\boldsymbol{T}}\dot{\boldsymbol{q}}_I. \tag{2}$$

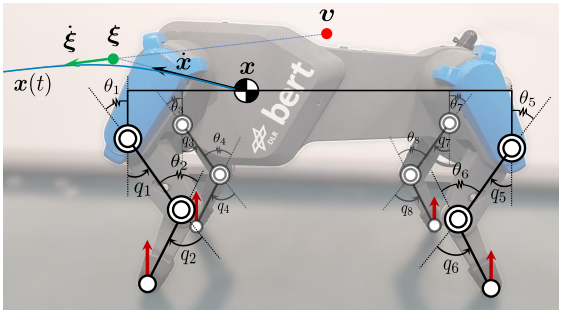


Fig. 2. The articulated soft quadruped DLR Bert with its kinematic structure superimposed. The leg joint positions $\boldsymbol{q}_j = [-\; q_k \;-]^\intercal$ and motor positions $\boldsymbol{\theta} = [-\; \theta_k \;-]^\intercal$ are highlighted. The virtual repellent point $\boldsymbol{v}$ pushes the DCM $\boldsymbol{\xi}$ to which the CoM $\boldsymbol{x}$ converges to produce its trajectory $\boldsymbol{x}(t)$.

Substituting (2) into the first equation of (1) and pre-multiplying $\boldsymbol{T}^\intercal$ lead to

$$\boldsymbol{T}^\intercal\boldsymbol{M}\boldsymbol{T}\ddot{\boldsymbol{q}}_I + \boldsymbol{T}^\intercal(\boldsymbol{M}\dot{\boldsymbol{T}} + \boldsymbol{C}\boldsymbol{T})\dot{\boldsymbol{q}}_I + \boldsymbol{T}^\intercal\left[\boldsymbol{g} + \boldsymbol{S}^\intercal\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}_j)\right] = \boldsymbol{0}.$$

Linearizing this equation around an equilibruim $\boldsymbol{q}_{eq}$ yields

$$\boldsymbol{M}_I(\boldsymbol{q}_{eq})\Delta\ddot{\boldsymbol{q}}_I + \boldsymbol{K}_I(\boldsymbol{q}_{eq})\Delta\boldsymbol{q}_I = \boldsymbol{0}, \tag{3}$$

where $\Delta\boldsymbol{q}_I = \boldsymbol{q}_I - \boldsymbol{q}_{I_{eq}}$, the inertia of the independent coordinates is given by $\boldsymbol{M}_I(\boldsymbol{q}) = \boldsymbol{T}^\intercal\boldsymbol{M}\boldsymbol{T}$, and the Hessian of the potential energy gives the *effective stiffness*

$$\boldsymbol{K}_I(\boldsymbol{q}) = \frac{\partial\left\{\boldsymbol{T}^\intercal\left[\boldsymbol{g} + \boldsymbol{S}^\intercal\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}_j)\right]\right\}}{\partial\boldsymbol{q}_I} \approx \boldsymbol{T}^\intercal\left(\bar{\boldsymbol{K}} + \frac{\partial\boldsymbol{g}}{\partial\boldsymbol{q}}\right)\boldsymbol{T}.$$

The matrix $\bar{\boldsymbol{K}} = \boldsymbol{S}^\intercal\boldsymbol{K}\boldsymbol{S}$ is the equivalent stiffness of the generalized coordinates and $\boldsymbol{T}$ is considered quasi-static.

The modal oscillations of (3) can be found by studying the generalized eigenvalue problem

$$\lambda_i\boldsymbol{M}_I(\boldsymbol{q}_{eq})\boldsymbol{u}_i = \boldsymbol{K}_I(\boldsymbol{q}_{eq})\boldsymbol{u}_i, \tag{4}$$

and are of the form

$$\Delta\boldsymbol{q}_I(t) = \boldsymbol{u}_i A\sin(\sqrt{\lambda_i}\,t + \psi). \tag{5}$$

These oscillations are commonly referred to as the *eigenmodes* of the linearized system (3). In general, there are $n - n_c$ distinct oscillations $\Delta\boldsymbol{q}_I(t)$ about $\boldsymbol{q}_{eq}$. They evolve on the subspaces (*eigenspaces*) generated by $\boldsymbol{u}_i$, which are the eigenvectors of $\boldsymbol{K}_I^{-\frac{1}{2}}\boldsymbol{M}_I\boldsymbol{K}_I^{\frac{1}{2}}$. The oscillation frequency is the square-root of the associated eigenvalue $\omega_i = \sqrt{\lambda_i}$. Eq. (5) parametrizes the small natural oscillatory motions of the independent variables $\boldsymbol{q}_I$ of (1), constrained by (2), and close enough to $\boldsymbol{q}_{eq}$. The amplitude A and phase ψ depend on the initial conditions $\Delta\boldsymbol{q}_I(0)$ and $\Delta\dot{\boldsymbol{q}}_I(0)$.

Remark 1. *With $n_c \leq n_j$, eq. (5) by itself describes the natural oscillations $\Delta\boldsymbol{x}$ of the CoM around the corresponding equilibrium.*

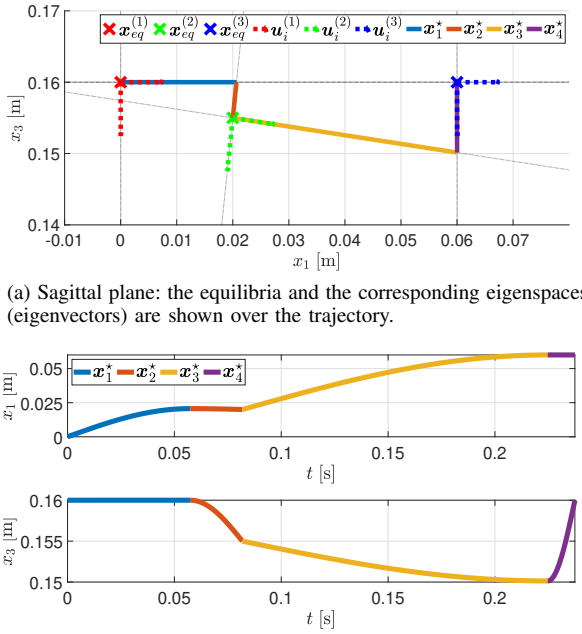
B. Natural Motions of the Center of Mass

In this subsection, we propose the method to generate CoM trajectories that are compliant with the natural dynamics utilizing the theory presented in Sec. III-A

Assumption 1. *We assume to be provided with a sequence of n_s stances $\Sigma = \{\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_{n_s}\}$ that have been planned to move the robot. Here, a stance $\boldsymbol{\sigma}_i = \{c_k | k = 1, \ldots, n_k\}$ is a set of limb contacts [28].*

The generation of natural motions of the CoM is as follows: we choose modal oscillations around the stances equilibria and concatenate them in order to move in the planned direction. As mentioned, we plan these motions only in the sagittal plane of the robot. We reserve to lift this simplification in future works. We define the following procedure, which is also described in a compact way in Algorithm 1:

- Compute a succession of n_s admissible equilibria $Q_{eq} = \{\boldsymbol{q}_{eq}^{(1)}, \dots, \boldsymbol{q}_{eq}^{(n_s)}\}$ of (1) corresponding to the contact positions given by the precomputed stances in Σ. That is, we request that a $\boldsymbol{q}_{eq}$, found by means of IK from a given stance, complies with the static equilibrium condition $\boldsymbol{g}(\boldsymbol{q}_{eq}) = \boldsymbol{S}^\intercal \boldsymbol{K}(\boldsymbol{\theta}_{eq} - \boldsymbol{S}\boldsymbol{q}_{eq}) + \boldsymbol{J}_c^\intercal(\boldsymbol{q})\boldsymbol{\lambda}_{eq}$, while being subject to friction cone constraints on $\boldsymbol{\lambda}_{eq}$ and to bounds on both $\boldsymbol{\theta}_{eq}$ and $\boldsymbol{\lambda}_{eq}$. This problem can be solved effectively using a quadratic program, which can also minimize some cost related to the contact and elastic forces: for instance, $\frac{1}{2}\boldsymbol{\lambda}_{eq}^\intercal\boldsymbol{\lambda}_{eq} + \frac{1}{2}(\boldsymbol{\theta}_{eq} - \boldsymbol{S}\boldsymbol{q}_{eq})^\intercal\boldsymbol{K}(\boldsymbol{\theta}_{eq} - \boldsymbol{S}\boldsymbol{q}_{eq})$. This is performed by ComputeAdmissibleEquilibria.
- Next, for each of the equilibria in Q_{eq}, the function ComputeCoMPositions computes forward kinematics and finds the associated CoM positions $X_{eq} = \{\boldsymbol{x}_{eq}^{(1)}, \dots, \boldsymbol{x}_{eq}^{(n_s)}\}$.
- For each element $\boldsymbol{x}_{eq}^{(i)} \in X_{eq}$, using (4), compute the related *eigenspaces* $ES_{eq}^{(i)} = (\lambda_{1,2}^{(i)}, \boldsymbol{u}_{1,2}^{(i)})$ to get a list $ES_{eq} = \{ES_{eq}^{(1)}, \dots, ES_{eq}^{(n_s)}\}$. This is done by ComputeEigenSpaces.
- Initialize an empty variable for the trajectory $\boldsymbol{x}^\star(t)$ using function EmptyTrajectory.
- Then, iteratively repeat the following for each successive pairs of equilibria $(\boldsymbol{x}_{eq}^{(i)}, \boldsymbol{x}_{eq}^{(i+1)}) \in X_{eq}$:
  - Find all possible intersection points of the corresponding *eigenspaces* in $ES_{eq}^{(i)}$ and $ES_{eq}^{(i+1)}$ (FindIntersections).
  - From these intersections, choose a random $\bar{\boldsymbol{x}}$ through the function ChooseRandom.
  - Then, ComputeOscillation computes the trajectories $\boldsymbol{x}'_{(i)}(t)$ and $\boldsymbol{x}''_{(i)}(t)$, from $\boldsymbol{x}_{eq}^{(i)}$ to $\bar{\boldsymbol{x}}$ and from $\bar{\boldsymbol{x}}$ to $\boldsymbol{x}_{eq}^{(i+1)}$, considering the appropriate time intervals. That is, we find a motion $\boldsymbol{x}'_{(i)}(t)$, which follows modal oscillation with eigenfrequency $\sqrt{\lambda^{(i)}}$ and is along the associated eigenvector $\boldsymbol{u}^{(i)}$, connecting the equilibrium $\boldsymbol{x}_{eq}^{(i)}$ to the state $\bar{\boldsymbol{x}}$, and such that $\forall t' : \boldsymbol{x}'_{(i)}(t) = \bar{\boldsymbol{x}}$ we have $\dot{\boldsymbol{x}}'_{(i)}(t') = \boldsymbol{0}$. Such a motion is given by $\boldsymbol{x}'_{(i)}(t) = \boldsymbol{x}_{eq}^{(i)} + \Delta\boldsymbol{x}'_{(i)}(t)$, where $\Delta\boldsymbol{x}'_{(i)}(t) = \boldsymbol{u}^{(i)}A'\sin(\sqrt{\lambda^{(i)}}\,t)$ with $A' = ||\bar{\boldsymbol{x}} - \boldsymbol{x}_{eq}^{(i)}||$. The trajectory $\boldsymbol{x}'_{(i)}(t)$ goes from $\boldsymbol{x}_{eq}^{(i)}$ to $\bar{\boldsymbol{x}}$ in the time interval $\left[0, \frac{\pi}{2\sqrt{\lambda^{(i)}}}\right]$. A similar procedure can be followed to find the motion from $\bar{\boldsymbol{x}}$ to $\boldsymbol{x}_{eq}^{(i+1)}$ by computing $\boldsymbol{x}''_{(i)}(t) = \boldsymbol{x}_{eq}^{(i+1)} + \Delta\boldsymbol{x}''_{(i)}(t)$, where $\Delta\boldsymbol{x}''_{(i)}(t) = \boldsymbol{u}^{(i+1)}A''\sin(\sqrt{\lambda^{(i+1)}}\,t)$, $A'' = ||\bar{\boldsymbol{x}} - \boldsymbol{x}_{eq}^{(i+1)}||$, and considering the portion of $\boldsymbol{x}''_{(i)}(t)$ in the interval $\left[-\frac{\pi}{2\sqrt{\lambda^{(i+1)}}}, 0\right]$.
  - Finally, append the found $\boldsymbol{x}'_{(i)}(t)$ and $\boldsymbol{x}''_{(i)}(t)$ to the natural one $\boldsymbol{x}^\star(t)$ (AppendTrajectory).



(a) Sagittal plane: the equilibria and the corresponding eigenspaces (eigenvectors) are shown over the trajectory.

(b) Forward and vertical components of the CoM trajectory against time.

Fig. 3. An example of natural trajectory $\boldsymbol{x}^\star(t)$ obtained by combining modal oscillations (Algorithm 1) for trotting with DLR Bert. We show the three equilibria ($\boldsymbol{x}_{eq}^{(1)}$, $\boldsymbol{x}_{eq}^{(2)}$, and $\boldsymbol{x}_{eq}^{(3)}$) connected by four piecewise trajectories $\boldsymbol{x}_i^\star$, which build the natural motion.

Remark 2. *Given the periodicity of gaits, there exists a subset $\bar{\Sigma}$ of the stances Σ that contains all information about the chosen gait. Hence, it suffices to apply Algorithm 1 only to $\bar{\Sigma}$ and concatenate the resulting $\boldsymbol{x}^\star(t)$ appropriately to find the natural CoM trajectory associated with Σ.*

An example trajectory, computed using Algorithm 1 and following the natural dynamics, is shown in Fig. 3. Therein, from three equilibria ($\boldsymbol{x}_{eq}^{(1)}$, $\boldsymbol{x}_{eq}^{(2)}$, and $\boldsymbol{x}_{eq}^{(3)}$) of a trot gait for

Algorithm 1 Natural Trajectory of the Center of Mass

Input: Σ **Output:** $\boldsymbol{x}^\star(t)$

procedure FIND NATURAL TRAJECTORY(Σ)
  $Q_{eq} \leftarrow$ ComputeAdmissibleEquilibria(Σ)
  $X_{eq} \leftarrow$ ComputeCoMPositions(Q_{eq})
  $ES_{eq} \leftarrow$ ComputeEigenSpaces(X_{eq})
  $\boldsymbol{x}^\star(t) \leftarrow$ EmptyTrajectory()
  for $\boldsymbol{x}_{eq}^{(i)} \in X_{eq}$ **do**
    $ES_{eq}^{(i)} \leftarrow$ GetEigenSpaces($ES_{eq}, \boldsymbol{x}_{eq}^{(i)}$)
    $ES_{eq}^{(i+1)} \leftarrow$ GetEigenSpaces($ES_{eq}, \boldsymbol{x}_{eq}^{(i+1)}$)
    $\{\boldsymbol{x}_{int}\} \leftarrow$ FindIntersections($ES_{eq}^{(i)}, ES_{eq}^{(i+1)}$)
    $\bar{\boldsymbol{x}} \leftarrow$ ChooseRandom($\{\boldsymbol{x}_{int}\}$)
    $\boldsymbol{x}'_{(i)}(t) \leftarrow$ ComputeOscillation($\boldsymbol{x}_{eq}^{(i)}, \bar{\boldsymbol{x}}$)
    $\boldsymbol{x}''_{(i)}(t) \leftarrow$ ComputeOscillation($\bar{\boldsymbol{x}}, \boldsymbol{x}_{eq}^{(i+1)}$)
    $\boldsymbol{x}^\star(t)$.AppendTrajectory($\boldsymbol{x}'_{(i)}(t)$)
    $\boldsymbol{x}^\star(t)$.AppendTrajectory($\boldsymbol{x}''_{(i)}(t)$)
  return $\boldsymbol{x}^\star(t)$

the quadruped DLR Bert, we concatenate modal oscillations to get a natural motion of the CoM.

Trajectories found by Algorithm 1, even though compliant with the mechanics of the robot, are not guaranteed to be physically feasible, in the sense of locomotion. This means that they disregard the physical constraint that the line of action of the sum of external forces acting on the CoM must intersect the base of support. Traditional motion planning efficiently deals with such a constraint. For this reason, in Sec. IV we quickly review the three-dimensional DCM planning, which will be used to "follow" CoM references generated by Algorithm 1.

IV. Physical Feasibility of CoM Trajectories

We discuss here briefly the *Divergent Component of Motion* layer of our architecture ("DCM Motion Planning" in Fig. 1 and DCMPlanning in Algorithm 2), only to the extent that it is needed to this paper. It is indeed worth stressing out here that - with the exception of two mappings of the quadrupedal gait phases to the conventional *single support* and *double support* phases of bipeds - this block is not a contribution of this work. It is instead an implementation of the theory laid by [25], [29].

The DCM is a point $\boldsymbol{\xi} = (\xi_1 \ \xi_2 \ \xi_3)^\intercal$ defined as $\boldsymbol{\xi} = \boldsymbol{x} + b\dot{\boldsymbol{x}}$, where $\boldsymbol{x}$ is the center of mass (CoM) position, $\dot{\boldsymbol{x}}$ its velocity, $b = \sqrt{\Delta z/g}$ is a constant, Δz is the average height of the CoM above the ground and g is the gravitational acceleration. The second order dynamics of the CoM can be rewritten as follows

$$\dot{\boldsymbol{x}} = -\frac{1}{b}(\boldsymbol{x} - \boldsymbol{\xi}), \quad \dot{\boldsymbol{\xi}} = \frac{1}{b}(\boldsymbol{\xi} - \boldsymbol{v}). \tag{6}$$

Here, $\boldsymbol{v} = (v_1 \ v_2 \ v_3)^\intercal$ is the so called *Virtual Repellent Point* (VRP), which encodes the effects of the total force acting on the CoM. Eq. (6) shows that the CoM has a stable first order dynamics that converges to the DCM and the DCM has an unstable first order dynamics that diverges pushed by the VRP. From a planning perspective, the problem of finding feasible CoM trajectories reduces to planning properly $\boldsymbol{v}$. Solving (6) with suitable boundary conditions and a feasible VRP trajectory $\boldsymbol{v}(t)$ yields closed-form feasible C^2 continuous trajectories of the CoM. This process is briefly summarized below.

1) Trajectory Generation: The motion of the CoM is divided into n_ϕ *transition phases*. For each phase $\phi \in \{1, \dots, n_\phi\}$, we design VRP trajectories $\boldsymbol{v}_\phi(t)$. Then, the corresponding $\boldsymbol{\xi}_\phi(t)$ and $\boldsymbol{x}_\phi(t)$ are found by integrating (6). The boundary values of each phase are some $\boldsymbol{v}_{\phi,0}$, $\boldsymbol{\xi}_{\phi,0}$ and $\boldsymbol{x}_{\phi,0}$ for the *start points* and $\boldsymbol{v}_{\phi,T}$, $\boldsymbol{\xi}_{\phi,T}$ and $\boldsymbol{x}_{\phi,T}$ for the *end points*. Continuity of the whole trajectories is guaranteed by connecting *start* and *end points* of successive phases: $\boldsymbol{v}_{\phi,0} = \boldsymbol{v}_{\phi-1,T}$, $\boldsymbol{\xi}_{\phi,0} = \boldsymbol{\xi}_{\phi-1,T}$, and $\boldsymbol{x}_{\phi,0} = \boldsymbol{x}_{\phi-1,T}$. These points can be seen as waypoints of the trajectories. The generation of these waypoints is discussed later on.

For each phase, VRP waypoints are smoothly interpolated in time as

$$\boldsymbol{v}_\phi(t) = (1 - f_\phi(t))\boldsymbol{v}_{\phi,0} + f_\phi(t)\boldsymbol{v}_{\phi,T}, \tag{7}$$

where, $t \in [0, T_\phi]$, T_ϕ is the constant phase duration, and the function $f_\phi(t)$, called *temporal interpolation function*, is a polynomial with the following properties: (i) $f_\phi(0) = 0$, (ii) $f_\phi(T_\phi) = 1$, (iii) $0 \leq f_\phi(t) \leq 1, \ \forall t \in [0, T_\phi]$. Substituting (7) in (6) and solving, for each phase, with boundary condition $\boldsymbol{\xi}_\phi(T_\phi) = \boldsymbol{\xi}_{\phi,T}$, yields

$$\boldsymbol{\xi}_\phi(t) = \alpha_{\phi,\xi}(t)\boldsymbol{v}_{\phi,0} + \beta_{\phi,\xi}(t)\boldsymbol{v}_{\phi,T} + \gamma_{\phi,\xi}(t)\boldsymbol{\xi}_{\phi,T}, \tag{8}$$

with $\alpha_{\phi,\xi}(t)$, $\beta_{\phi,\xi}(t)$, and $\gamma_{\phi,\xi}(t)$ being nonlinear functions of t, b and T_ϕ. Similarly, combining (8) and the first equation in (6), and solving for the boundary condition $\boldsymbol{x}_\phi(0) = \boldsymbol{x}_{\phi,0}$, yields

$$\boldsymbol{x}_\phi(t) = \alpha_{\phi,x}(t)\boldsymbol{v}_{\phi,0} + \beta_{\phi,x}(t)\boldsymbol{v}_{\phi,T} + \gamma_{\phi,x}(t)\boldsymbol{\xi}_{\phi,T} + \delta_{\phi,x}(t)\boldsymbol{x}_{\phi,0}. \tag{9}$$

Analogously to (8), also $\alpha_{\phi,x}(t)$, $\beta_{\phi,x}(t)$, $\gamma_{\phi,x}(t)$, and $\delta_{\phi,x}(t)$ are nonlinear functions of t, b and T_ϕ.

In summary, given the *start* and *end* waypoints for each phase, smooth trajectories of VRP, DCM and CoM can be found using (7), (8), and (9), respectively, and concatenated to find the whole trajectories $\boldsymbol{v}(t)$, $\boldsymbol{\xi}(t)$, and $\boldsymbol{x}(t)$.

2) Waypoint Generation: Waypoints are computed from the given sequence of $n_s = n_\phi + 1$ stances in Σ. We place VRP waypoints $\boldsymbol{V} = [\boldsymbol{v}_1 \dots \boldsymbol{v}_{n_s}]^\intercal \in \mathbb{R}^{n_s \times 3}$ for each $\boldsymbol{\sigma}_i \in \Sigma$. Our strategy is to choose the $\boldsymbol{v}_i$ based on the equilibria $\boldsymbol{x}_{eq}^{(i)}$ computed for the stances $\boldsymbol{\sigma}_i$ in Sec. III-B. We set $\boldsymbol{v}_i$ with the same horizontal position, however we impose the height to be Δz. This is to be consistent with the definition of the time constant b. Notice that these $\boldsymbol{v}_i$ are the same waypoints that were previously denoted as *start* and *end points*. For instance, for the i^{th} phase $\boldsymbol{v}_{\phi,0} = \boldsymbol{v}_i$ and $\boldsymbol{v}_{\phi,T} = \boldsymbol{v}_{i+1}$. DCM waypoints $\boldsymbol{\Xi} = [\boldsymbol{\xi}_1 \dots \boldsymbol{\xi}_{n_s}]^\intercal \in \mathbb{R}^{n_s \times 3}$ and CoM waypoints $\boldsymbol{X} = [\boldsymbol{x}_1 \dots \boldsymbol{x}_{n_s}]^\intercal \in \mathbb{R}^{n_s \times 3}$ can be found from $\boldsymbol{V}$ by evaluating (8) for $t = 0$, and (9) for $t = T_\phi$, for all n_ϕ phases, and writing the resulting relations in matrix form. In this way, waypoints can be efficiently found as follows [29]

$$\boldsymbol{\Xi} = \begin{bmatrix} {}^{\Xi}\boldsymbol{C}_V & {}^{\Xi}\boldsymbol{c}_\xi \end{bmatrix} \begin{bmatrix} \boldsymbol{V} \\ \boldsymbol{\xi}_f^\intercal \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} {}^{X}\boldsymbol{C}_V & {}^{X}\boldsymbol{c}_\xi & {}^{X}\boldsymbol{c}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{V} \\ \boldsymbol{\xi}_f^\intercal \\ \boldsymbol{x}_s^\intercal \end{bmatrix}. \tag{10}$$

Here, $\boldsymbol{\xi}_f = \boldsymbol{\xi}_{n_s} = \boldsymbol{v}_{n_s}$ for stopping the robot and $\boldsymbol{x}_s = \boldsymbol{x}_1 = \boldsymbol{x}(0)$. The matrices ${}^{\Xi}\boldsymbol{C}_V$, ${}^{\Xi}\boldsymbol{c}_\xi$, ${}^{X}\boldsymbol{C}_V$, ${}^{X}\boldsymbol{c}_\xi$, and ${}^{X}\boldsymbol{c}_x$ depend on the coefficients $\alpha_{\phi,\xi}(0)$, $\beta_{\phi,\xi}(0)$, $\gamma_{\phi,\xi}(0)$, $\alpha_{\phi,x}(T_\phi)$, $\beta_{\phi,x}(T_\phi)$, $\gamma_{\phi,x}(T_\phi)$, and $\delta_{\phi,x}(T_\phi)$. Please see [29] for more details.

Remark 3. *The formulation of DCM-based planning adopted in this paper [24], [29] requires that the average height of the CoM does not deviate significantly from Δz. Hence, oscillations in the vertical direction are not present.*

3) Phases Mapping: Here, we propose a simple mapping from quadrupedal to bipedal phases to ease the implementation of the DCM framework on multi-legged robots.

Definition 1 (Quadrupedal Single Support)**.** *We define as* single support *the phases in which the VRP reference is constant. In these phases, one or more legs are swinging.*

Definition 2 (Quadrupedal Double Support)**.** *We define as* double support *the phases in which the VRP reference is interpolated between two waypoints. These correspond to stance phases, in which no legs are swinging.*

4) Feet Trajectories: The trajectories of the limbs $\boldsymbol{r}_k(t)$, with $k \in \{LF, RF, LH, RH\}$, can be computed from the stances Σ as usually done in the literature, i.e. using cubic or quintic splines. In this work, we use third-order polynomials. Note that in our formulation, planning feet motions only requires the step height Δh, since the stances in Σ already specify the step forward-length and side-length.

V. Motion Optimization for Natural Walking

This section aims to enable the DCM-based CoM trajectory (eq. (9) of Sec. IV) to follow CoM motions, compliant with the natural dynamics, which are generated by Algorithm 1 of Sec. III-B. This part corresponds to "Motion Optimization" in Fig. 1 and MotionOptimization in Algorithm 2.

We notice that the trajectories generated by (9) can be shaped by varying either the phase times T_ϕ or the VRP waypoints $\boldsymbol{V}$. Hence, T_ϕ and $\boldsymbol{V}$ can be used to set up an optimization to match the $\boldsymbol{x}(t)$, computed using (9), to the natural CoM motion $\boldsymbol{x}^\star(t)$ over a number of $p \leq n_\phi$ significant phases.

In this work, we choose to optimize over the phase times since we believe that matching the natural frequencies of the system will already produce desirable results (as confirmed by the experiments in Sec. VI). The possibility of finding online "optimal" VRP adjustments is left to a future work.

We choose as cost function the least-squares difference between the two trajectories:

$$w(\boldsymbol{x}(t), \boldsymbol{x}^\star(t)) = ||\boldsymbol{x}(t) - \boldsymbol{x}^\star(t)||_2^2. \tag{11}$$

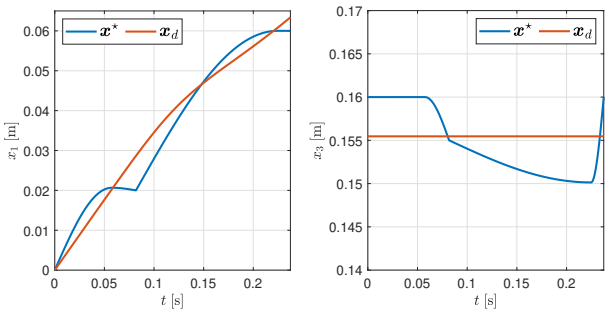
Thereafter, we set up the nonlinear optimization

$$\begin{aligned} \min_{T_1,\dots,T_p} \quad & w(\boldsymbol{x}(t, \boldsymbol{\Xi}, \boldsymbol{X}), \boldsymbol{x}^\star(t)) \\ \text{s.t.} \quad & l_\phi \leq T_\phi \leq u_\phi \end{aligned} \tag{12}$$

Here, l_ϕ and u_ϕ are arbitrary lower and upper bounds on the phase times. Recall that both $\boldsymbol{\Xi}$ and $\boldsymbol{X}$ are nonlinear functions of the phase times T_ϕ; moreover, the coefficients $\alpha_{\phi,\xi}$, $\beta_{\phi,\xi}$, $\gamma_{\phi,\xi}$, $\alpha_{\phi,x}$, $\beta_{\phi,x}$, $\gamma_{\phi,x}$, and $\delta_{\phi,x}$ all depend nonlinearly on T_ϕ. This is the reason why the cost function w is nonlinear with respect to the optimization variables T_ϕ.

In practice, nonlinear optimization problems, such as (12), can be addressed with state-of-the-art numerical methods, such as branch-and-bound and SQP algorithms [30], [31], or other methods as [32]–[35]. Also note that nonlinear optimization (12) is not performed online in Algorithm 2, thus avoiding heavy computations during online planning. Only the DCM-based trajectory generation, which is proven to be very efficient [24], is carried out online.

We applied optimization (12) using as $\boldsymbol{x}^\star$ the trajectory computed in Sec. III-B and shown in Fig. 3. Here, we optimized only over the first two phases (single and double support), which we henceforth call T_{swing} and T_{stance} for the sake of clarity. Assuming a symmetric robot and periodic trot, these two phases fully specify the gait. In line with Remark 3, we used as Δz the average of the vertical component of $\boldsymbol{x}^\star$ (the x_3 of Fig. 3b). The resulting DCM-based optimal CoM trajectory $\boldsymbol{x}_d(t)$, which uses the optimal T_{swing} and T_{stance}, is shown in Fig. 4. It is noteworthy that the forward component of the natural trajectory is followed as close as possible by the second order dynamics of the CoM (see Fig. 4a). However, given our particular choice of VRPs, by virtue of Remark 3, Fig. 4b shows that the vertical oscillations are zeroed to the average height.



(a) Forward axis. (b) Vertical axis.

Fig. 4. The natural CoM trajectory $\boldsymbol{x}^\star(t)$ (previously shown in Fig. 3) against the DCM-based CoM trajectory $\boldsymbol{x}_d(t)$ that uses the optimal phase times given by optimization (12). Notice the vertical oscillations zeroed by the DCM framework.

At this point, having defined all the necessary concepts, we merge our CoM modal evolution method (Sec. III-B) into the DCM planning framework (Sec. IV) using the proposed motion optimization (Sec. V). Thus, we build our complete planning architecture, shown in Algorithm 2.

VI. Experimental Validation

In this section, we validate the proposed approach on the compliant quadruped DLR Bert [1]. The robot weighs 3.2 kg. Its length, width, and height are approximately 0.33 m, 0.21 m, and 0.20 m in zero position. The stiffness of the joint springs is 2.7 Nm/rad.

A. Implementation Details

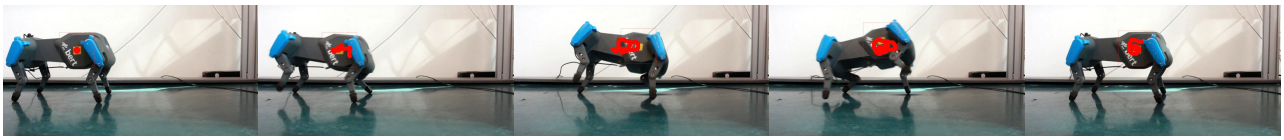
We purposely do not employ any feedback controller for two reasons: 1) for a fair evaluation of the planned trajectories, and 2) because it is known that feedback actions on compliant robots have a stiffening effect [36]. Hence, we perform offline closed-loop inverse kinematics [37]

$$\dot{\boldsymbol{q}}_{k_d} = \boldsymbol{J}_{k_R}^\dagger [\dot{\boldsymbol{s}}_{k_d} + \boldsymbol{K}_P(\boldsymbol{s}_{k_d} - \boldsymbol{s}_k)], \tag{13}$$

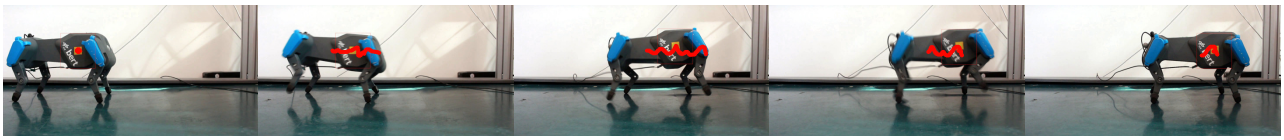
Algorithm 2 Natural Motion Planning. Σ is a pre-planned sequence of stances and Δh the step height. FindSubset finds the subset of stances $\bar{\Sigma}$ that are periodically repeated in Σ. Find Natural Trajectory generates the natural CoM motion through Algorithm 1. MotionOptimization solves the motion optimization (12). DCMPlanning is a DCM-planning layer [25]. ForwardPlan performs IK and sends the trajectories to the robot for execution.

Input: Σ, Δh **Output:** $\boldsymbol{x}_d(t)$, $\boldsymbol{r}_{j_d}(t)$

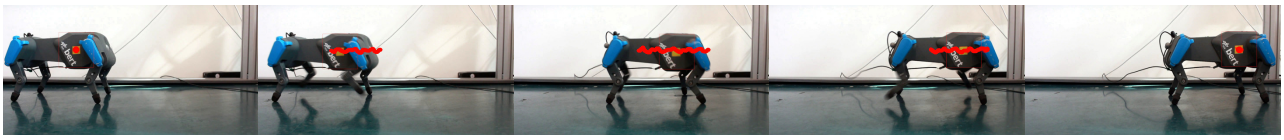
procedure Natural Motion Planning(Σ)
  $\bar{\Sigma} \leftarrow$ FindSubset(Σ)
  $\boldsymbol{x}^\star(t) \leftarrow$ Find Natural Trajectory($\bar{\Sigma}$)
  $(T_1, \dots, T_p) \leftarrow$ MotionOptimization($\boldsymbol{x}^\star(t)$)
  repeat
    $\boldsymbol{x}_d(t) \leftarrow$ DCMPlanning($\Sigma, T_1, \dots, T_p$)
    $\boldsymbol{r}_{j_d}(t) \leftarrow$ FeetTrajectories($\Sigma, \Delta h, T_1, \dots, T_p$)
    ForwardPlan($\boldsymbol{x}_d(t), \boldsymbol{r}_{j_d}(t)$)
  until stopping condition



(a) *First Experiment*: Bert moves using DCM-based planning with $T_{swing} = 0.5\,\mathrm{s}$ and $T_{stance} = 0.4\,\mathrm{s}$.



(b) *Second Experiment*: Bert moves using DCM-based planning with $T_{swing} = 0.2\,\mathrm{s}$ and $T_{stance} = 0.1\,\mathrm{s}$.



(c) *Third Experiment*: Bert moves "naturally" using the optimal timings $T_{swing} = 0.1335\,\mathrm{s}$ and $T_{stance} = 0.1019\,\mathrm{s}$, provided by Algorithm 2.

Fig. 5. Experiments performed on DLR Bert. The photo sequences are in $t \in [0, 5]\,\mathrm{s}$, at intervals of $1.25\,\mathrm{s}$ each. Subfigures (a) and (b) show Bert trotting during the first two experiments with DCM-based planning. The results are not satisfactory: undesired oscillations can be noticed. Our planning architecture (c), instead, confers a robust walking even without control feedback since Algorithm 2 produces trajectories that are compliant with the natural dynamics of the system. The trajectories of the CoM, estimated using video tracking, are shown in red. See also the submission video.

where, $\boldsymbol{s}_k = \boldsymbol{r}_k - \boldsymbol{x}$ is the relative vector between the k^{th} foot and the CoM, $\boldsymbol{s}_{k_d} = \boldsymbol{r}_{k_d} - \boldsymbol{x}_d$ is the corresponding planned vector, and $\boldsymbol{J}_{k_R}$ is the relative Jacobian associated to $\dot{\boldsymbol{s}}_k$. The desired leg link coordinate trajectories would be $\boldsymbol{q}_{j_d} = [\boldsymbol{q}_{LF_d}^\intercal\, \boldsymbol{q}_{RF_d}^\intercal\, \boldsymbol{q}_{LH_d}^\intercal\, \boldsymbol{q}_{RH_d}^\intercal]^\intercal$.

We compute offline the $\boldsymbol{\theta}_d$ that are required to track $\boldsymbol{q}_{j_d}$. Assuming steady-state conditions, we use [38]

$$\boldsymbol{\theta}_d = \boldsymbol{q}_{j_d} + \boldsymbol{K}^{-1}\bar{\boldsymbol{g}}(\boldsymbol{q}_{j_d}). \tag{14}$$

Here, $\bar{\boldsymbol{g}} = (\boldsymbol{N}_c^\intercal \boldsymbol{S}^\intercal)\boldsymbol{N}_c^\intercal \boldsymbol{g}$ is the constraint-consistent gravity compensation torques with $\boldsymbol{N}_c$ being the contact nullspace matrix [39].

Algorithm 2 was implemented in MATLAB/SIMULINK and `fmincon` with the SQP method was used to solve optimization (12). The dynamic and Jacobian matrices were computed using the method presented in [40]. For moving the robot we feedforward the motor position trajectories computed by (14).

B. Experiments: Results and Discussion

We performed three experiments with trot gait. Initially, we planned the CoM motion exclusively based on DCM planning without motion optimization. We stress again that, to the best of the authors' knowledge, this is the first time that this framework is employed for quadrupedal robots. For the sake of comparison, we also let the robot walk with small "non-optimal" timings. Finally, we used our planner (Algorithm 2) to optimize the phase times, which were found to be very close to the eigenfrequencies of the linearizations. The three experiments are shown in the photo sequences of Fig. 5a, 5b, and 5c and in the attached submission video.

In the first experiment (shown in Fig. 5a), we used the following two phase durations: $T_{swing} = 0.5\,\mathrm{s}$ and $T_{stance} = 0.4\,\mathrm{s}$. We noticed that since we do not consider at all the elasticity of the robot, the behaviour is undesirable; the robot bounced back and forth due to the elastic forces and tracking errors. Instead, the second experiment ($T_{swing} = 0.2\,\mathrm{s}$ and $T_{stance} = 0.1\,\mathrm{s}$), although not as bad as the first one, exhibited some small disturbing oscillations. This is probably due to the non-compliance with the system eigenfrequencies. Finally, for the third experiment, we use the optimal phase durations $T_{swing} = 0.1335\,\mathrm{s}$ and $T_{stance} = 0.1019\,\mathrm{s}$ output by Algorithm 2. The corresponding natural and optimized trajectories, $\boldsymbol{x}^\star(t)$ and $\boldsymbol{x}_d(t)$, were already shown in Fig. 3 and Fig. 4, respectively. The trot was visually impressive and very robust notwithstanding the relative IK and the feedforward control. The robot seemed to abide by its intrinsic elasticity. This shows that our planning method complies with the natural dynamics of the system.

Furthermore, we observed that, in the first two experiments, due to the non-conformity to the natural dynamics, the robot movements were considerably more jerky and unsteady when compared to the third experiment (please see the CoM paths, drawn using the video tracking software Kinovea[1], in Fig. 5 and watch the attached video submission). Furthermore, notice also that with Algorithm 2 the robot covered more distance than with the DCM planning with non-optimal timings, despite the same total time of $5\,\mathrm{s}$.

VII. CONCLUSIONS

In this paper, we addressed the challenge of planning natural locomotion with articulated soft quadrupeds. The proposed architecture generates CoM trajectories on the robot's sagittal plane using modal oscillations. By means of nonlinear optimization, phase durations are found so that said natural trajectories are matched by the DCM planning framework. We achieved promising - although preliminary - experimental results with the quadruped DLR Bert. Future work will extend our natural trajectory generation beyond the sagittal plane and use VRP adjustments to optimize the trajectories. We will also employ nonlinear modes to better capture the robot's natural dynamics. Finally, we will conduct a thorough and quantitative evaluation of energy efficiency: this was not included at present due to space constraints.

[1] https://www.kinovea.org/

REFERENCES

[1] D. Lakatos, K. Ploeger, F. Loeffl, D. Seidel, F. Schmidt, T. Gumpert, F. John, T. Bertram, and A. Albu-Schäffer, "Dynamic locomotion gaits of a compliantly actuated quadruped with slip-like articulated legs embodied in the mechanical design," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3908–3915, 2018.

[2] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2016, pp. 38–44.

[3] C. Semini, N. G. Tsagarakis, B. Vanderborght, Y. Yang, and D. G. Caldwell, "Hyq-hydraulically actuated quadruped robot: Hopping leg prototype," in *2008 2nd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*. IEEE, 2008, pp. 593–599.

[4] X. Zhou and S. Bi, "A survey of bio-inspired compliant legged robot designs," *Bioinspiration & biomimetics*, vol. 7, no. 4, p. 041001, 2012.

[5] C. Della Santina, M. G. Catalano, and A. Bicchi, "Soft robots," *Encyclopedia of Robotics*, 2020.

[6] R. Alexander *et al.*, "Three uses for springs in legged locomotion," *International Journal of Robotics Research*, vol. 9, no. 2, pp. 53–61, 1990.

[7] J. Ackerman and J. Seipel, "Energy efficiency of legged robot locomotion with elastically suspended loads," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 321–330, 2013.

[8] D. Lakatos, C. Rode, A. Seyfarth, and A. Albu-Schäffer, "Design and control of compliantly actuated bipedal running robots: Concepts to exploit natural system dynamics," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 930–937.

[9] M. de Lasa and M. Buehler, "Dynamic compliant quadruped walking," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, vol. 3. IEEE, 2001, pp. 3153–3158.

[10] M. Hutter, C. D. Remy, M. A. Hoepflinger, and R. Siegwart, "Efficient and versatile locomotion with highly compliant legs," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 2, pp. 449–458, 2012.

[11] O.-S. Kwon, R. Choi, and D.-H. Lee, "Locomotion control of a compliant legged robot from slow walking to fast running," *International Journal of Advanced Robotic Systems*, vol. 9, no. 6, p. 240, 2012.

[12] G. M. Gasparri, S. Manara, D. Caporale, G. Averta, M. Bonilla, H. Marino, M. Catalano, G. Grioli, M. Bianchi, A. Bicchi *et al.*, "Efficient walking gait generation via principal component representation of optimal trajectories: application to a planar biped robot with elastic joints," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2299–2306, 2018.

[13] P. Fankhauser, M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, and R. Siegwart, "Reinforcement learning of single legged locomotion," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 188–193.

[14] D. Seidel, D. Lakatos, and A. Albu-Schäffer, "Data-driven discrete planning for targeted hopping of compliantly actuated robotic legs," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2261–2266.

[15] R. J. Full and D. E. Koditschek, "Templates and anchors: neuromechanical hypotheses of legged locomotion on land," *Journal of experimental biology*, vol. 202, no. 23, pp. 3325–3332, 1999.

[16] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," 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. IEEE, 2001, pp. 239–246.

[17] W. J. Schwind, *Spring loaded inverted pendulum running: A plant model*. University of Michigan, 1998.

[18] R. Rosenberg, "On nonlinear vibrations of systems with many degrees of freedom," in *Advances in applied mechanics*. Elsevier, 1966, vol. 9, pp. 155–242.

[19] A. F. Vakakis, L. I. Manevitch, Y. V. Mikhlin, V. N. Pilipchuk, and A. A. Zevin, *Normal modes and localization in nonlinear systems*. Springer, 2001.

[20] F. Wang, A. K. Bajaj, and K. Kamiya, "Nonlinear normal modes and their bifurcations for an inertially coupled nonlinear conservative system," *Nonlinear Dynamics*, vol. 42, no. 3, pp. 233–265, 2005.

[21] A. Albu-Schaeffer and C. Della Santina, "A review on nonlinear modes in conservative mechanical systems," *Annual Reviews in Control*, 2020.

[22] D. Lakatos, W. Friedl, and A. Albu-Schäffer, "Eigenmodes of nonlinear dynamics: Definition, existence, and embodiment into legged robots with elastic elements," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1062–1069, 2017.

[23] C. Della Santina, D. Lakatos, A. Bicchi, and A. Albu-Schaeffer, "Using nonlinear normal modes for execution of efficient cyclic motions in articulated soft robots," in *International Symposium on Experimental Robotics*. Springer, 2020, pp. 566–575.

[24] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *Ieee transactions on robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[25] ——, "Three-dimensional bipedal walking control using divergent component of motion," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2600–2607.

[26] A. Werner, B. Henze, D. A. Rodriguez, J. Gabaret, O. Porges, and M. A. Roa, "Multi-contact planning and control for a torque-controlled humanoid robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5708–5715.

[27] J. Bednarek, N. Maalouf, M. J. Pollayil, M. Garabini, M. G. Catalano, G. Grioli, and D. Belter, "Cnn-based foothold selection for mechanically adaptive soft foot," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 225–10 232.

[28] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[29] G. Mesesan, J. Englsberger, C. Ott, and A. Albu-Schäffer, "Convex properties of center-of-mass trajectories for locomotion based on divergent component of motion," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3449–3456, 2018.

[30] D. Kennedy, "Some branch and bound techniques for nonlinear optimization," *Mathematical Programming*, vol. 42, no. 1, pp. 147–157, 1988.

[31] N. I. Gould and P. L. Toint, "Sqp methods for large-scale nonlinear programming," in *IFIP Conference on System Modeling and Optimization*. Springer, 1999, pp. 149–178.

[32] J. B. Rosen, "The gradient projection method for nonlinear programming. part i. linear constraints," *Journal of the society for industrial and applied mathematics*, vol. 8, no. 1, pp. 181–217, 1960.

[33] ——, "The gradient projection method for nonlinear programming. part ii. nonlinear constraints," *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, no. 4, pp. 514–532, 1961.

[34] I. Pólik and T. Terlaky, "Interior point methods for nonlinear optimization," in *Nonlinear optimization*. Springer, 2010, pp. 215–276.

[35] A. Ruszczynski, *Nonlinear optimization*. Princeton university press, 2011.

[36] C. Della Santina, M. Bianchi, G. Grioli, F. Angelini, M. Catalano, M. Garabini, and A. Bicchi, "Controlling soft robots: balancing feedback and feedforward elements," *IEEE Robotics & Automation Magazine*, vol. 24, no. 3, pp. 75–83, 2017.

[37] B. Siciliano, "A closed-loop inverse kinematic scheme for on-line joint-based robot control," *Robotica*, vol. 8, no. 3, pp. 231–243, 1990.

[38] A. De Luca and F. Flacco, "Dynamic gravity cancellation in robots with flexible transmissions," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 288–295.

[39] M. Mistry and L. Righetti, "Operational space control of constrained and underactuated systems," in *Robotics: Science and systems*, vol. 7, 2012, pp. 225–232.

[40] G. Garofalo, C. Ott, and A. Albu-Schäffer, "On the closed form computation of the dynamic matrices and their differentiations," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 2364–2359.