

# Optimal Control of Walkers with Parallel Actuation

Ludovic De Matteis<sup>1,2,\*</sup>, Virgile Batto<sup>1,3</sup>, Justin Carpentier<sup>2</sup>, Nicolas Mansard<sup>1,4</sup>

**Abstract**—Legged robots with closed-loop kinematic chains are increasingly prevalent due to their increased mobility and efficiency. Yet, most motion generation methods rely on serial-chain approximations, sidestepping their specific constraints and dynamics. This leads to suboptimal motions and limits the adaptability of these methods to diverse kinematic structures. We propose a comprehensive motion generation method that explicitly incorporates closed-loop kinematics and their associated constraints in an optimal control problem (OCP), integrating kinematic closure conditions and their analytical derivatives. This allows the solver to leverage the non-linear transmission effects inherent to closed-chain mechanisms, reducing peak actuator efforts and expanding their effective operating range. Unlike previous methods, our framework does not require serial approximations, enabling more accurate and efficient motion strategies. We also are able to generate the motion of more complex robots for which an approximate serial chain does not exist. We validate our approach through simulations and experiments, demonstrating superior performance in complex tasks such as rapid locomotion and stair negotiation. This method enhances the capabilities of current closed-loop robots and broadens the design space for future kinematic architectures.

## I. INTRODUCTION

Recent progress in biped locomotion result from the sound combination of more mature motion generation techniques and continuous improvements in robot design and hardware [1]. Several advancements have recently demonstrated the advantages of leveraging parallel kinematic chains to boost the dynamic capabilities of robots [2]. This architecture offers benefits like lighter lower limbs and improved impact absorption [3] at the cost of introducing more complex dynamics, [4] eventually making the robot more difficult to simulate and control [5]. Several of the best modern biped robots already rely on such designs (see Fig 1). Yet, the specificity of this complex architecture is mostly ignored when generating their movements [14].

The approach mostly used in the literature is to only model an approximate serial chain in the generation of whole-body locomotion or movement, and to rely on an ad-hoc actuation model, often supported by the robot manufacturer, to transfer the reference joint motion and torques into actuator commands. On Atrias, a Spring Loaded Inverted Pendulum (SLIP) model is used in motion generation and

This work is supported by ROBOTEX 2.0 (ROBOTEX ANR-10-EQPX-0044 and TIRREX ANR-21-ESRE-0015), ANITI (ANR-19-P3IA-0004), by the French government (INEXACT ANR-22-CE33-0007 and "Investissements d'avenir" ANR-19-P3IA-0001) (PRAIRIE 3IA Institute), and by the Louis Vuitton ENS Chair on Artificial Intelligence

<sup>1</sup> Gepetto, LAAS-CNRS, Université de Toulouse, France

<sup>2</sup> Inria, École normale Supérieure, PSL Research University, Paris, France

<sup>3</sup> Auctus, Inria, centre de l'université de Bordeaux, Talence, France

<sup>4</sup> Artificial and Natural Intelligence Toulouse Institute, France

\* Corresponding author: ludovic.de-matteis@laas.fr

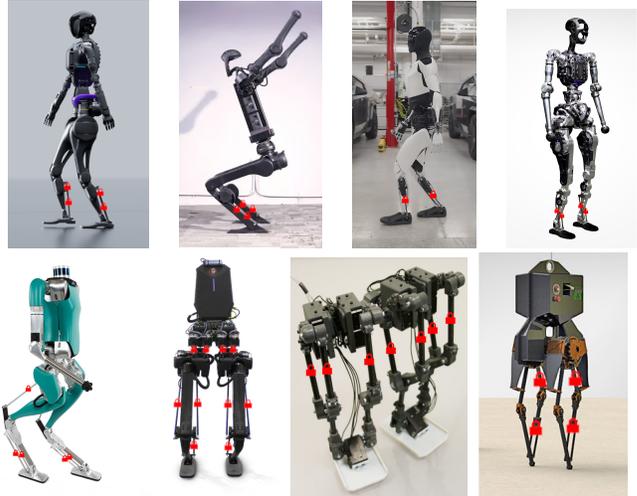


Fig. 1: Examples of legged robots with closed-loop kinematics, ranging (top-left to bottom-right) from robots with a main serial chain (Fourier GR1 [6], Unitree H1 [7], Tesla Optimus [8], Adam [9]), robot with an approximate serial chain (Digit [10]) and robots without an approximate serial chain (Kangaroo [11], Disney bipedal robot [12], Atrias [13]). Each red lock represents a visible closure of the kinematic chain.

computed torques are directly transferred to actuator control with of reduction ratio of 1 [15], [16]. On Talos, Whole-Body controllers [17], such as Whole Body Model Predictive Control [18] have been applied on an approximate serial model, letting the low level control transfer the serial torques to actuator commands [19]. Reinforcement Learning (RL) [20] methods, where the computation of the robot dynamic is delegated to a simulator, have also been applied to several of modern robots such as H1, Digit and Fourier GR1. Yet, modern GPU simulators used in RL, such as PhysicX or MuJoCo XLA (MJX), do not currently support closed kinematics. This yield approaches where the complexity of the closed loop is only accounted for in the low level controller. Only robot specific controllers, alternating the learning process and configuration projection, have yet been applied on such architecture, for instance [21] on Digit.

The main contribution of this paper is to derive a complete and general modeling methodology to enable the simulation of closed-loop kinematics, mostly targeting MPC although these approaches are the same as those used in the simulator for RL. The dynamics of a poly-articulated robot is governed by the unconstrained equations of motion and finds efficient algorithms and their implementations in

the literature [22], [23], [24]. Recent works have proposed general methods to write the dynamics of a system under contact constraints [25], and showed a general form of its derivatives, [4], [26]. It is already accepted that closed-kinematic constraints can be cast under the same scope as contact constraints, to be used in the same framework [25].

In this paper, we first highlight the importance of accurately modeling closed-loop transmissions, as demonstrated in the experimental section. To address this, our main contribution is to propose a comprehensive solution to compute the optimal movement for various walker kinematics. Our method applies efficiently to robots with an underlying serial kinematic chain (such as H1, electric-Atlas, and Adam), to those with approximated serial kinematics (such as Cassie and Digit), and extends to more complex kinematics without serial approximations (such as Kangaroo and the Disney biped). It relies on fully modeling the closed-loop linkages, including their derivatives, and integrating this model into a trajectory optimizer. We experimentally show that our method generates optimal movements in case where neglecting actuator transmissions leads to sub-optimality, to unrealistic behavior, or even to motion generation failure. While simpler methods that decouple actuator models from kinematic models might be suitable for robots with approximated serial kinematics, we demonstrate that our method remains effective for more complex robots where such decoupling is not possible, opening the door to more efficient designs in the future.

After quickly recalling in Sec II the background in optimizing the trajectory of robots subject to dynamic constraints, we provide an efficient computation of the derivatives of the dynamics of a robot involving closed-loop constraints in Sec. III. We then use this new dynamic to formulate and solve an optimal control problem for the walk of a robot with parallel actuation in Sec. IV). The result section (Sec V) highlight the limitations of using an *Approximate Serial Model* of such a robot by comparing it to the *Closed Kinematics Model* over different motions. We also demonstrate the application of our method to control a robot for which no *Approximate Serial Model* can be defined.

## II. BACKGROUND

### A. Optimal control

Optimal control of a multibody system consists in finding the control inputs that minimize a given cost function while satisfying the system dynamics. In this paper, we seek for a solution to this problem by solving a trajectory optimization problem formulated by multiple shooting [27], i.e. by optimizing over both the control inputs  $u[k]$  and the states  $x[k]$  at each discrete time step  $k$ , with the dynamics as an explicit constraint. It is classically written as the following non-linear program (NLP):

$$\begin{aligned} \min_{u, x} \quad & \sum_k^{N-1} l_k(x[k], u[k]) + l_N(x[N]) \\ \text{s.t.} \quad & \forall k \in \llbracket 0, N-1 \rrbracket \quad x[k+1] = f_k(x[k], u[k]) \end{aligned} \quad (1)$$

The cost function is usually defined as the sum of running costs  $l_k$  and a terminal cost  $l_N$ . The functions  $f_k$  are defining the system dynamics at each time step  $k$ . The case where  $f$  models a poly-articulated system in contact is well-known and the derivations are recalled next.

### B. Multibody dynamics

The dynamic of a multibody system is well described by the Lagrangian equation of motion:

$$M(q)\ddot{q} + b(q, \dot{q}) = \tau(u) + J_c(q)^T \lambda \quad (2)$$

where  $M$  represents the generalized inertia matrix,  $b$  the non-linear terms,  $q$  the generalized coordinates of the system,  $\dot{q}$  the generalized velocities<sup>1</sup> and  $\tau$  the torques applied on the system joints, function of the controls  $u$  (usually, the controls correspond to the motor joints torques and all other joints torques are zero). The term  $\lambda$  represents an external force applied on the system, where  $J_c$  is the Jacobian of the contact point. For systems subject to mechanical constraints (such as foot-ground contact or kinematics-closure), these forces  $\lambda$  arise from the satisfaction of the constraints.

### C. Dynamics subject to constraints

From Gauss principle of least action [28], [29], [30], the acceleration  $\ddot{q}$  of the system under contacts should be as close as possible to its free acceleration  $\ddot{q}_f$ , with respect the the kinematic metric, while satisfying the contact constraints. This can be rewritten as the following optimization problem:

$$\begin{aligned} \min_{\ddot{q}} \quad & \|\ddot{q} - \ddot{q}_f\|_M^2 \\ \text{s.t.} \quad & J_c \ddot{q} + a_0 = 0 \end{aligned} \quad (3)$$

The constraint in (3) corresponds to the second-order time derivatives of the constraint of contact, where  $a(q, \dot{q}, \ddot{q}) = J_c \ddot{q} + a_0(q, \dot{q})$  describes the relative acceleration of the contact points, that we consider here - without loss of generality - to be wanted to be zero, and where  $a_0(q, \dot{q}) = a(q, \dot{q}, 0) = \dot{J}_c \dot{q}$  is the acceleration due to the velocity only. Deriving first-order optimality conditions, (3) boils down to:

$$\underbrace{\begin{bmatrix} 0 & J_c \\ J_c^T & M \end{bmatrix}}_K \underbrace{\begin{bmatrix} \lambda \\ \ddot{q} \end{bmatrix}}_y = \underbrace{\begin{bmatrix} -a_0 \\ \tau(u) - b \end{bmatrix}}_k \quad (4)$$

where  $\lambda$ , the dual variables of the optimization problem, correspond to the contact forces applied on the system. This system can be solved to get  $\ddot{q}$  and  $\lambda$  given the state  $x = (q \ \dot{q})$  and the controls  $u$ . The solution  $y = K^{-1}k$  to this problem exists and is unique whenever the matrix  $K$  is invertible, which usually happens when the matrix  $M$  is positive definite and when the constraints are not redundant. Proximal resolution has been proposed to solve the problem in settings where  $J_c$  is not full rank [25]. This formulation has been extensively applied to model ground contact of walking robots. In the next section, we propose to modify the constraint 3 to model internal forces arising in closed-kinematics linkages.

<sup>1</sup>In our implementation,  $q$  typically contains quaternions for representing basis orientation and ball-joints configurations, hence the notation  $\dot{q}$ , although classical, is abusive

#### D. Derivatives of the constrained dynamics

MPC typically uses gradient-based solvers to get the solution of (1), which implies to evaluate the derivatives of  $y$  with respect to  $x$  and  $u$ . In [4] the authors proposed the derivatives of a robot in contact with its environment. More recent work generalize these to arbitrary contacts [26], [23]. Following these works, the gradient of  $y$  with respect to  $z \in \{q, \dot{q}, u\}$  can be reduced to:

$$\frac{\partial y}{\partial z} = K^{-1} \begin{bmatrix} \frac{\partial a_0}{\partial z} \\ \frac{\partial f_D}{\partial z} \end{bmatrix} \quad (5)$$

where the Inverse Dynamics (ID) function outputs the joint torques creating acceleration  $\ddot{q}$  under contact forces  $f$ .

$$ID(q, \dot{q}, \ddot{q}, \lambda) = M\ddot{q} + J_c^T \lambda + b \quad (6)$$

The derivatives of these terms with respect to the controls  $u$  will not be explicit in this paper as the first term is independent on  $u$  and the second depends on the chosen actuation model. The derivatives of ID when  $J_c$  are the joint jacobians have been established [31] and are sufficient for the case of a contact between the robot and its environment, yet we will see that they need to be extended in our setting.

### III. A CONSTRAINT MODELING THE KINEMATIC CLOSURE

#### A. Formulation of the constraint in acceleration

We consider the mechanical linkage between two bodies of the robot, characterized by frames  $\mathcal{F}_1$  and  $\mathcal{F}_2$  rigidly attached to each of them. We choose to write the 6D contact constraint on the relative placement as:

$$\text{Log}({}^1M_2) = 0 \quad (7)$$

where  ${}^1M_2 \in \text{SE}(3)$  is the rigid transformation between the frames and  $\text{Log}$  is the retraction from  $\text{SE}(3)$  to  $\mathbb{R}^6$  [32]. The first-order time derivative of this constraint can be expressed as the difference in spatial velocities expressed in a common frame. We choose as a convention to express all quantities in the frame  $\mathcal{F}_1$ , giving the constraint:

$$\nu_c = {}^1\nu_1 - {}^1X_2{}^2\nu_2 = 0 \quad (8)$$

where, following the notations introduced by Featherstone [22],  ${}^1\nu_1$  and  ${}^2\nu_2$  are the spatial velocities of the two bodies expressed in their respective frames,  ${}^1X_2$  is the Plücker coordinate transform, i.e. the adjoint matrix of  $\text{SE}(3)$ , from  $\mathcal{F}_2$  coordinates to  $\mathcal{F}_1$  coordinates. In the same way, the second-order time derivative of (7) can be expressed as the derivative of the relative spatial velocity  $\nu_c$ .

$$\begin{aligned} a_c &= {}^1a_1 - {}^1X_2{}^2a_2 - [{}^1\nu_2 \ {}^1\nu_1]_{\times} {}^1\nu_2 \\ &= \underbrace{{}^1a_1}_{\gamma_1} - \underbrace{{}^1X_2{}^2a_2}_{\gamma_2} + \underbrace{[{}^1\nu_1]_{\times} {}^1\nu_2}_{\gamma_3} \end{aligned} \quad (9)$$

with the spatial cross-product  $[\nu]_{\times}$  (*small adjoint*) given by

$$[\nu]_{\times} \triangleq \begin{bmatrix} [\omega]_{\times} & [\mathbf{v}]_{\times} \\ \mathbf{0} & [\omega]_{\times} \end{bmatrix} \quad (10)$$

where  $\omega$  and  $\nu$  are the angular and linear velocities.

#### B. Differentiation of the acceleration constraint

As shown in (5), we need the derivatives of the terms  $\gamma_1$ ,  $\gamma_2$  and  $\gamma_3$  with respect to the configuration vector  $q$  and velocity  $\dot{q}$ . The first of these derivatives yields directly:

$$\frac{\partial \gamma_1}{\partial q} = \frac{\partial {}^1a_1}{\partial q} = \frac{{}^1\partial a_1}{\partial q} \quad (11)$$

Note that we make here a clear distinction between the terms  $\frac{\partial {}^A a_B}{\partial q}$  and  $\frac{{}^A \partial a_B}{\partial q}$  as explained in [22] (section 2.10). To compute the derivative of the second term, we can use the method proposed in [33] - i.e. we search the time derivatives of  $\gamma_2$  under the form  $\dot{\gamma}_2 = G\dot{q}$  to deduce  $\frac{\partial \gamma_2}{\partial q} = G$ .

$$\begin{aligned} \frac{\partial \gamma_2}{\partial t} &= \frac{\partial {}^1X_2{}^2a_2}{\partial t} \\ &= \frac{\partial {}^1X_2{}^2a_2}{\partial t} + {}^1X_2 \frac{\partial {}^2a_2}{\partial t} \\ &= [{}^1(\nu_2 - \nu_1)]_{\times} {}^1X_2{}^2a_2 + {}^1X_2 \frac{\partial {}^2a_2}{\partial t} \\ &= -[{}^1X_2{}^2a_2]_{\times} {}^1(\nu_2 - \nu_1) + {}^1X_2 \frac{\partial {}^2a_2}{\partial t} \end{aligned} \quad (12)$$

which yields

$$\frac{\partial \gamma_2}{\partial q} = -[{}^1X_2{}^2a_2]_{\times} ({}^1J_2 - {}^1J_1) + {}^1X_2 \frac{\partial {}^2a_2}{\partial q} \quad (13)$$

We proceed in a similar way for the third and last term:

$$\begin{aligned} \frac{d\gamma_3}{dt} &= \frac{\partial [{}^1\nu_1]_{\times} {}^1X_2{}^2\nu_2}{\partial t} \\ &= [{}^1\dot{\nu}_1]_{\times} {}^1\nu_2 + [{}^1\nu_1]_{\times} \left( \frac{\partial {}^1X_2{}^2\nu_2}{\partial t} \right) + [{}^1\nu_1]_{\times} \left( {}^1X_2 \frac{\partial {}^2\nu_2}{\partial t} \right) \end{aligned} \quad (14)$$

which gives after development

$$\begin{aligned} \frac{\partial \gamma_3}{\partial q} &= -[{}^1\nu_2]_{\times} \frac{\partial {}^1\nu_1}{\partial q} - [{}^1\nu_1]_{\times} \left( [{}^1X_2{}^2\nu_2]_{\times} ({}^1J_2 - {}^1J_1) \right) \\ &\quad + [{}^1\nu_1]_{\times} {}^1X_2 \frac{\partial {}^2\nu_2}{\partial q} \end{aligned} \quad (15)$$

All these terms can be accessed using rigid body dynamics algorithms such as those included in Pinocchio [23].

As the action matrices do not depend on  $\dot{q}$ , the derivatives of  $a_c$  with respect to  $\dot{q}$  are direct and left to the reader.

#### C. Derivatives of Inverse Dynamic

We will now look at the derivatives of ID with respect to  $q$  and  $\dot{q}$ . Let us write the expression of ID as a function of the forces applied on the joints. We will denote these forces by  $\phi_k$  and the corresponding joint Jacobians  $J_k$  for joint  $j_k$ .

$$ID(q, \dot{q}, \ddot{q}, f) = M(q)\ddot{q} + b(q, \dot{q}) + \sum_k J_k(q)^T \phi_k(q, f) \quad (16)$$

where the  $\phi_k$  are typically expressed in the reference frame of joint  $j_k$  denoted by  $\mathcal{F}_{j_k}$ . We will now omit the dependencies to simplify the notation. The derivative with respect to the  $q$  is given by:

$$\frac{\partial ID}{\partial q} = \frac{\partial (M\ddot{q} + b)}{\partial q} + \sum_k \frac{\partial J_k^T}{\partial q} \phi_k + \sum_k J_k^T \frac{\partial \phi_k}{\partial q} \quad (17)$$

The first two terms are the classical Recursive Newton-Euler Algorithm (RNEA) derivatives, which are well established and implemented in the Pinocchio library [31], [23].

To compute the third term, we denote by  $j_1$  and  $j_2$  the parent joints of  $\mathcal{F}_1$  and  $\mathcal{F}_2$ . Following the choice of (9), the force  $f$  arising from the constraint (7) is expressed in  $\mathcal{F}_1$ . Considering that only  $f$  acts on the system, then all forces  $\phi_k$  are null except  $\phi_1$  and  $\phi_2$  which are:

$$\begin{aligned}\phi_1 &= {}^{j_1}X_{c_1}^* f \\ \phi_2 &= -{}^{j_2}X_{c_2}^* {}^{c_2}X_{c_1}^*(q)f\end{aligned}\quad (18)$$

where  $X^*$  is the Plücker transform on forces (dual adjoint),  ${}^{j_1}X_{c_1}$  is the fixed placement of the contact frame  $\mathcal{F}_1$  with respect to the joint frame  $\mathcal{F}_{j_1}$  (respectively  ${}^{j_2}X_{c_2}$ ) and  ${}^{c_2}X_{c_1}$  is function of  $q$ . We can see that  $\phi_1$  is independent of  $q$  while  $\phi_2$  is not. Its derivative with respect to  $q$  is given by:

$$\begin{aligned}\frac{\partial \phi_{j_2}}{\partial q} &= -\frac{\partial {}^{j_2}X_{c_2}^*}{\partial q} f \\ &= [{}^{j_2}X_{c_2}^*]_{\times} (J_2 - {}^{j_2}X_{j_1} J_1)\end{aligned}\quad (19)$$

where,  $J_1, J_2$  are the joint jacobians respectively expressed in  $\mathcal{F}_1$  and  $\mathcal{F}_2$  and the term  $[f]_{\times}^*$  can be defined as follows:

$$[f]_{\times}^* \triangleq \begin{bmatrix} 0 & [f_{linear}]_{\times} \\ [f_{linear}]_{\times} & [f_{angular}]_{\times} \end{bmatrix}\quad (20)$$

With the existing derivatives of RNEA [31], this completes the computation of the derivative of ID with respect to  $q$ .

#### IV. BENCHMARK IMPLEMENTATION

##### A. Kinematic models

We consider two different models of robots with closed-loop kinematics.

1) *Robot with an approximate serial kinematics*: First, we consider a robot that is composed of a main serial kinematic chain, whose actuators are shifted upfront of their serial joints by closed-loop mechanical linkages, similar to H1, Adam or electric Atlas. For these robots, a common approach [34] is to decouple the generation of the motion of the main serial chain from the control of the shifted actuators. We will show that not accounting for the actuator model in the motion generation leads to serious failure cases when the robot is pushed beyond slow walk on a flat terrain. For this first benchmark, we used the model of the robot Bipetto, available in open source [35] with explicit actuator model and depicted in Fig 2. Like H1, Adam, or electric Atlas, it features a knee motor with a four-bar linkage and two ankles motors with intricate four-bar linkages, inspired by Digit. Each closed-loop transmission creates a reduction ratio between the motors and the joints that depends on the robot configuration. We show in Fig. 3 the variation of the reduction ratio of the knee actuation with respect to the knee angle, revealing an highly non-linear relation. Getting an analytical formula for this variable reduction ratio is in general very difficult and poorly generalizes to new models, limiting its possible use in control settings. Our method

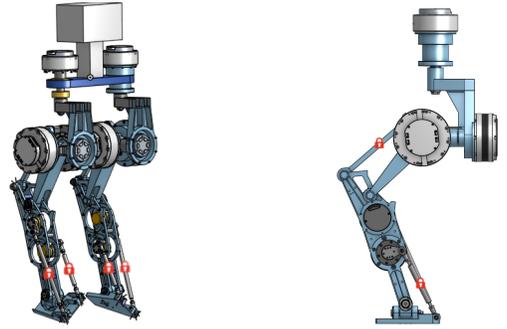


Fig. 2: Robot model used for our benchmark. Each red lock represents a closure of the kinematic chain. In our model, we represent the chain as a tree-like structure with added contact constraints by splitting the bar in two at the lock position and adding 6D contact constraints

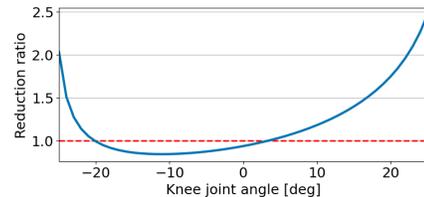


Fig. 3: Variation of the reduction ratio of the knee actuation with respect to the knee angle. The 0 angle corresponds to a nominal configuration of the robot while positive angles correspond to a stretched leg.

overcomes this limitation by providing a general formulation, directly transferable to other robot models.

Our benchmark compares the *Closed-Kinematics* model including closed-loop linkages against an *Approximate Serial* model considering only the serial joints of the robot (hip, knee, and ankle joints) and freezing the other degrees of freedom corresponding to the motor-to-joint transmission. In this simplification, the non-serial motors are fixed and fictive actuation is added on serial joints, yielding a fully serial model with 6 directly actuated revolute joints per leg. The inertia of the closed-loop transmission is modeled to be as realistic as possible, making no assumptions of low inertia. In the *Approximate Serial* model, the linkages joints are blocked in the initial configuration, yielding minimal inertia variations compared to the *Closed-Kinematics* model. We first exhibit the limitation of not accounting for the actuation in the OCP when a *Approximate Serial* model can be defined. We compare the movements generated with our OCP including the full *Closed-Kinematics* model against a classical OCP [19] using the *Approximate Serial* model and neglecting the actuation. In both OCP, the costs are the same, as described in Sec. IV-B. To compare the two movements, we simply lift the movement of the *Approximate Serial* model to the full *Closed-Kinematics* model, using a procedure described in Appendix A. Of course, this procedure may fail if the *Approximate Serial* trajectory corresponds to infeasible configuration or torques.

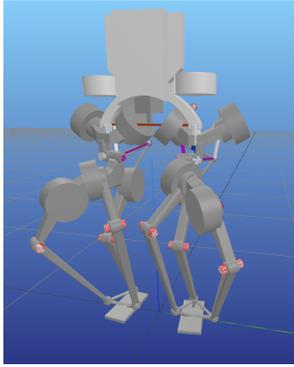


Fig. 4: Model of the Cleobot [36], a fully parallel robot. It uses three parallel chains of equivalent inertia for the knee and ankle actuation and three other parallel chains for hip actuation

2) *Robot without underlying serial kinematics:* We show that our method extends to robots without a simple correspondence to a serial chain, such as the Disney biped [12] or Kangaroo [11]. Note that Digit [10] also formally enter in this category due to its shin joint which is often mistaken as a knee joint, leading to more significant approximation than for example H1 [21]. We used the model of the Cleobot walker [36], presented in Fig 4, also available in open source, whose lower leg is composed of 3 parallel bodies attached to 3 revolute joints in place of the knee and ankle, and for which a serial model would be irrelevant to approximate.

### B. Optimal control problem details and implementation

We implement several variation of a base OCP pattern to achieve squats, walk, stairs and jump tasks. The dynamics under closed-loop constraints and its derivatives have been implemented in C++ in the Crocodyl library [4], used to write and solve the OCPs [37].

a) *Contact patterns:* In our OCP structure, we must define the contact pattern, i.e. when each foot is in contact with the floor, in order to associate to each time step the correct dynamics. For the squat task, foot are always in contact. The OCP for the walk motion consists of 4 steps of alternating double support and single support phases [19]. For the jump OCP, we define a simple sequence composed of double supports and a flying phase of known duration.

b) *Regularization costs:* For all tasks, we regularize the controls and the states around zero and around a reference configuration respectively. The ground contact forces are also regularized around 0 when the foot is in the air, the weight of the robot when the foot is in contact with the ground in single support phases and half of it during double support phases.

c) *Impact costs:* As the dynamics of the robot under contact only constraint the relative accelerations of the contact points, we add soft constraints on the velocity and placement of the foot at the end of single support phases to ensure it lands flat, still, and at the correct height. Not that hard constraints could be used instead using more recent solvers [38], [39] in order to limit the amount of tuning.

d) *Target costs:* The target costs are more specific to each task and, with the contact pattern, define in the OCP the task to be performed. First, the walk motion is defined by a target center of mass (CoM) velocity. It is represented in the OCP as a running cost with a residual corresponding to the CoM velocity error and as a terminal cost with a residual corresponding to the difference, in the forward direction, between the expected displacement given the target velocity and the real terminal position.

The squats motions are defined as a vertical translation of the CoM. Given a target minimal CoM elevation, we define a sinusoidal trajectory for the CoM elevation, starting with 0 velocity, reaching the target then returning to its initial position and ending with 0 velocity. In the OCP, this results in a running cost on the CoM elevation (planar CoM motions are not penalized), with a residual corresponding to the deviation from the target trajectory.

For the jump task, we proceed by defining a flying duration and deducing the expected maximum elevation of the CoM, that should be attained at mid flight. We then only apply in the OCP a cost for the CoM elevation at this time step, with a residual defined as the difference between the expected elevation and the actual one. In opposition to the squat task, this cost at only one time step is sufficient to constrain the entire flight phase.

Finally, the stairs climbing motion is defined by both a CoM velocity and a slope (i.e. a sequence of steps height). It simply adds to the walk OCP, a cost at each impact time (i.e. at the switch between single support to double support), with a residual defined as the difference between the foot elevation and the target step height.

e) *Additional costs:* Other costs are added to improve the realism of the trajectory (foot fly-high, center of pressure, forces in friction cones...). We do not present them here but one can find their definition in the example codes. Lastly, in the walking experiment, a cost penalizes the drift of the COM from its initial height.

f) *Actuator limits:* Our method can directly handle the real actuator constraints (which would not be possible when decoupling the whole-body motion generation from the actuators control). In the following section, we present the results without actuator limits, to better emphasize the comparison and prevent conservative assumptions.

## V. RESULTS

### A. Comparison of Approximate Serial and Closed-Kinematics models

1) *Squats:* Both problems converge to a reasonable squat movement for different target elevations, varying from 0.5 to 1.2 times the initial CoM elevation. Yet, without a proper model of the closed-loop transmission, the *Approximate Serial* trajectory reaches a singular configuration where the reference joint torques cannot be produced by the motors, as shown in Fig 5. On the opposite, the *Closed Kinematics* trajectory anticipates and produces an effective movement. While this effect is expected for such simple scenario, we

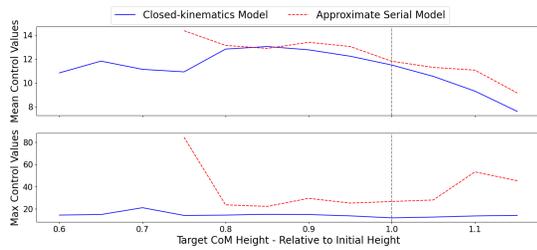


Fig. 5: Evolution of the Maximum and Mean knees controls as functions of the target CoM elevation in the squat motion. The target elevation is noted relative to the initial CoM height. For deeper squats, the required joint torques for the *Approximate Serial* trajectory yield excessive motor torques

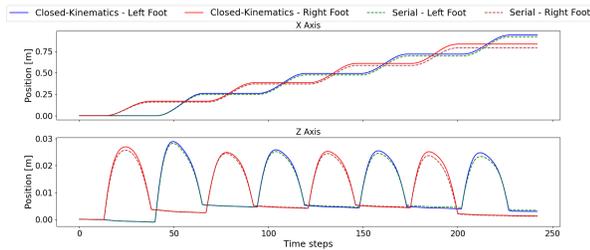


Fig. 6: Feet trajectory for the *Approximate Serial* and *Closed-Kinematics* trajectories during the reference walk motion

will now show that it similarly appears in more complex movements.

2) *Flat ground walk*: We now consider a walking motion on a flat terrain with a constant speed at  $0.5m/s$  (i.e. equivalent to  $6km/h$  walk for a human-sized robot), with an initial robot configuration that places its base at  $0.575 m$  above the ground. Fig 6 emphasizes that the movements obtained with both models are quite similar in appearance. Yet, the trajectory of the CoM, presented in Fig 7, show small deviations, especially in the elevation (i.e. position along the Z-axis). This reveals a convergence toward a slightly different optimal trajectory, yet not yielding any detrimental behavior for the *Approximate Serial* model.

3) *Variation of the velocity command*: Changing the velocity command from  $0.0m/s$  to  $1.2m/s$  (keeping the contact pattern unchanged) tends to accentuate the previous behavior. Fig. 8 shows the maximum knee controls reached over the steady state portion of the walking motion along with the mean CoM elevation. While we observe that the CoM elevation only diverge slightly for both models, a large difference can be observed in the peak controls. The contact sequence for the model being fixed, a higher velocity must be reached by using wider steps. When walking with small steps, the *Approximate Serial* model tend to lead to an higher CoM elevation, allowing it to reduce the joint torques. Yet, accelerating the walk forces the robot to have stretched legs during the steps, lowering the CoM, and eventually converging to a similar CoM elevation for both models. This pushes the closed-loop transmission in its non-linear part and the *Approximate Serial* model then fails to account

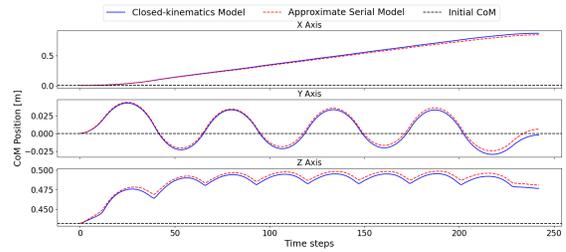


Fig. 7: CoM position trajectory for the *Approximate Serial* and *Closed-Kinematics* trajectories during the reference walk motion. The CoM reaches different steady state positions, revealing differences in the optimal trajectories

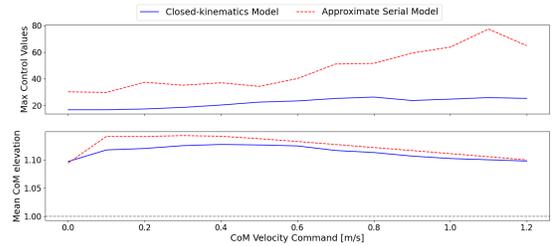


Fig. 8: Elevation of the CoM of the robot and of the mean knees motors controls as functions of the target forward velocity. The trajectory of the *Approximate Serial* Model yields excessive motor torques, eventually leading to convergence failure in the lifting process (cf App. A)

correctly for the reduction ratio, leading to exploding motor control and yielding a trajectory that cannot be transferred to the complete model (this occurs at  $0.9 m/s$  and above for our contact pattern). This demonstrates that the limits of the *Approximate Serial* model observed on the squat motion (Sec V-A.1) also appear on some more complex motions. More generally, the *Closed-Kinematics* model is able to take advantage of the actuator variable reduction created by the parallel linkage, hence leading to reasonable motor effort independently of the walk speed.

4) *Climbing stairs and jumping*: We generalized the results to more general locomotion scenarios: climbing stairs and jumping, with various robots. Like for fast walking, the *Approximate Serial* model fails to properly anticipate the limits of the actuation linkages and results in trajectories with excessive torques or reaching singularities. When jumping, mostly the knee reaches over extension and could be artificially clamped or penalized. Yet our method makes that hyper-parameter tuning useless, providing a better generalization. For stair climbing, clamping the knee would be even more limiting and could completely make some motions unfeasible. Moreover, for stair climbing, the ankle limits are also reached, for which an ad-hoc limitation (with 2 degrees of freedom) is more difficult to decide. The movements are reported in the companion video for the Bipetto walker and Digit.

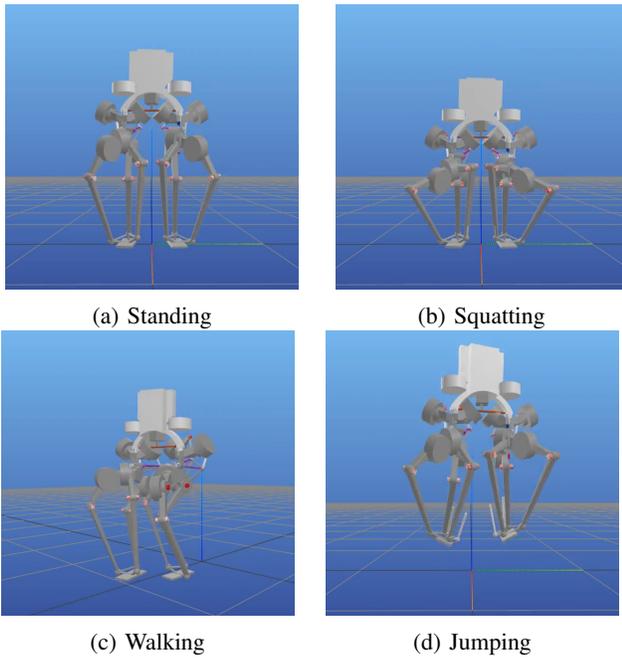


Fig. 9: Snapshots of motions performed on the Cleobot walker [36]

### B. Robot without underlying serial kinematics

Using the same costs as before, we perform squat, walk and jump motions on this architecture, as presented in Fig. 9, validating the generality of the approach. Examples of such motions can be found in the companion video and reproduced using the code provided [40]. It would not be possible to generate similar movements for this robot without our method since no approximate serial model is available and adjusting an equivalent serial kinematics would be very difficult. This result shows that our method enables the design of more complex walking kinematics, opening a possible research direction toward more effective walkers.

### C. Solver timings

Optimizing a walking trajectory consisting in  $T = 250$  time-steps of  $\Delta t = 15ms$  - i.e. a 3.75s trajectory - takes about 500ms per iteration using our implementation in Crocoddyl [4] with an 8 core parallelization. In this time, 43% is spend in the rollout of the problem - i.e. the forward dynamics, 30% in the dynamics derivatives and 25% in the backward pass of the FDDP algorithm. We notice a decrease in the step sizes of the FDDP algorithm, revealing an increase in the problem complexity. Future work should focus on improving the solver behavior and exploring different solvers. We note that, as expected, reducing the number of closed-loop constraints reduces the computation time for both the forward dynamics and its derivatives. This results shows that our additional derivatives computation induces minimal loss in computational efficiency, opening the way for real time implementation.

## VI. CONCLUSION

This paper presents a method to account for closed-loop kinematics in locomotion. Our method relies on establishing an efficient differentiable model of the closed-loop constraints and implementing it in an optimal control solver. We propose a complete implementation based on the open-source solver Crocoddyl. To validate our method, we developed a benchmark for comparing the motions obtained using an *Approximate Serial* model, that relies on an underlying serial chain and ignores the closed-loop transmission, to motions obtained with the *Closed-Kinematics* model. We showed the limitations of the *Approximate Serial* model on squats, walk, and stair climbing motions, revealing issues in joint torques estimation and actuator limits. We then proved the generality of our method by generating motions on a robot for which an *Approximate Serial* model cannot be defined, overcoming the limitations of previous methods. Our method directly improves the efficiency and dynamic range of actual robots and paves the road to more sophisticated actuation.

## REFERENCES

- [1] T. Mikołajczyk, E. Mikołajewska, H. F. Al-Shuka, T. Malinowski, A. Kłodowski, D. Y. Pimenov, T. Paczkowski, F. Hu, K. Giasin, D. Mikołajewski, *et al.*, “Recent advances in bipedal walking robots: Review of gait, drive, sensors and control systems,” *Sensors*, vol. 22, no. 12, p. 4440, 2022.
- [2] J.-P. Merlet, *Parallel robots*, vol. 128. Springer Science & Business Media, 2006.
- [3] Virgile Batto, T. Flayols, N. Mansard, and Margot Vulliez, “Comparative Metrics of Advanced Serial/Parallel Biped Design and Characterization of the Main Contemporary Architectures,” *IEEE-RAS International Conference on Humanoid Robots*, 2023.
- [4] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, “Crocoddyl: An efficient and versatile framework for multi-contact optimal control,” *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.
- [5] G. Liu, Y. Wu, X. Wu, Y. Kuen, and Z. Li, “Analysis and control of redundant parallel manipulators,” *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4, pp. 3748–3754, 2001.
- [6] “Fourier Intelligence.” <https://fourierintelligence.com/>, Accessed on 2024-09-12.
- [7] “Humanoid robot G1 — Unitree Robotics.” Accessed: 2024-09-12.
- [8] “Tesla AI & robotics.” <https://www.tesla.com/AI>, Accessed on 2024-09-12.
- [9] “Adam - PNDbotics.” <https://www.pndbotics.com/humanoid>, Accessed on 2024-09-12.
- [10] “AgilityProducts.” <https://agilityrobotics.com/products>, Accessed on 2024-09-12.
- [11] A. Roig, S. K. Kothakota, N. Miguel, P. Fernbach, E. M. Hoffman, and L. Marchionni, “On the Hardware Design and Control Architecture of the Humanoid Robot Kangaroo,” in *6th Workshop on Legged Robots during the International Conference on Robotics and Automation (ICRA 2022)*, 2022.
- [12] K. G. Gim, J. Kim, and K. Yamane, “Design and fabrication of a bipedal robot using serial-parallel hybrid leg mechanism,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.
- [13] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Sprowitz, A. Abate, and J. Hurst, “Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot,” *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1497–1521, 2016.
- [14] M. Boukheddimi, R. Kumar, S. Kumar, J. Carpentier, and F. Kirchner, “Investigations into exploiting the full capabilities of a series-parallel hybrid humanoid using whole body trajectory optimization,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10433–10439, IEEE, 2023.

- [15] A. T. Peekema, “Template-based control of the bipedal robot atrias,” 2015.
- [16] A. Ramezani, J. W. Hurst, K. Akbari Hamed, and J. W. Grizzle, “Performance analysis and feedback control of atrias, a three-dimensional bipedal robot,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 136, no. 2, p. 021012, 2014.
- [17] H. Dai, A. Valenzuela, and R. Tedrake, “Whole-body motion planning with centroidal dynamics and full kinematics,” *IEEE-RAS International Conference on Humanoid Robots*, pp. 295–302, 2014.
- [18] J. B. Rawlings, D. Q. Mayne, M. Diehl, *et al.*, *Model predictive control: theory, computation, and design*, vol. 2. Nob Hill Publishing Madison, WI, 2017.
- [19] E. Dantec, M. Taix, and N. Mansard, “First order approximation of model predictive control solutions for high frequency feedback,” *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 4448–4455, 2022.
- [20] E. Chane-Sane, P.-A. Leziart, T. Flayols, O. Stasse, P. Souères, and N. Mansard, “Cat: Constraints as terminations for legged locomotion reinforcement learning,” *IEEE/RIS International Conference on Intelligent Robots and Systems*, 2024.
- [21] Guillermo A. Castillo, Bowen Weng, Wei Zhang, and Ayonga Hereid, “Robust Feedback Motion Policy Design Using Reinforcement Learning on a 3D Digit Bipedal Robot,” *IEEE/RIS International Conference on Intelligent Robots and Systems*, 2021.
- [22] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.
- [23] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiroux, O. Stasse, and N. Mansard, “The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives,” *IEEE/SICE International Symposium on System Integration (SII)*, pp. 614–619, 2019.
- [24] M. L. Felis, “RBDL: an efficient rigid-body dynamics library using recursive algorithms,” *Autonomous Robots*, vol. 41, no. 2, pp. 495–511, 2017.
- [25] J. Carpentier, R. Budhiraja, and N. Mansard, “Proximal and sparse resolution of constrained dynamic equations,” *Robotics: Science and Systems*, 2021.
- [26] Shubham Singh, Ryan P. Russell, and Patrick M. Wensing, “Analytical Second-Order Derivatives of Rigid-Body Contact Dynamics: Application to Multi-Shooting DDP,” *IEEE-RAS International Conference on Humanoid Robots*, 2023.
- [27] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, “Fast Direct Multiple Shooting Algorithms for Optimal Robot Control,” *Fast Motions in Biomechanics and Robotics*, 2005.
- [28] S. Redon, A. Kheddar, and S. Coquillart, “Gauss’ least constraints principle and rigid body simulations,” *IEEE international conference on robotics and automation (ICRA)*, 2002.
- [29] D. Baraff, “Fast contact force computation for nonpenetrating rigid bodies,” *International Conference on Computer Graphics and Interactive Techniques*, pp. 23–34, 1994.
- [30] F. E. Udawadia, “Equations of motion for mechanical systems: A unified approach,” *International Journal of Non-linear Mechanics*, vol. 31, no. 6, pp. 951–958, 1996.
- [31] J. Carpentier and N. Mansard, “Analytical derivatives of rigid body dynamics algorithms,” in *Robotics: Science and systems (RSS)*, 2018.
- [32] J. Sola, J. Deray, and D. Atchuthan, “A micro lie theory for state estimation in robotics,” *arXiv:1812.01537*, 2018.
- [33] S. Kleff, J. Carpentier, N. Mansard, and L. Righetti, “On the derivation of the contact dynamics in arbitrary frames: Application to polishing with talos,” *IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*, 2022.
- [34] D. Mronga, S. Kumar, and F. Kirchner, “Whole-body control of series-parallel hybrid robots,” 02 2022.
- [35] “Example parallel robots: a repository containing several urdf models of legged robots with parallel kinematics - on github.” <https://github.com/Gepetto/example-parallel-robots>.
- [36] V. Batto, L. de Matteis, T. Flayols, M. Vulliez, and N. Mansard, “CLEO: Closed-Loop kinematics Evolutionary Optimization of bipedal structures.” working paper or preprint, Oct. 2024.
- [37] “Github - crocodyl - temporary fork.” <https://github.com/LudovicDeMatteis/crocodyl/tree/topic/contact-6D-closed-loop>.
- [38] W. Jallet, A. Bambade, E. Arlaud, S. El-Kazdadi, N. Mansard, and J. Carpentier, “Proxddp: Proximal constrained trajectory optimization,” 2023.
- [39] A. Jordana, S. Kleff, A. Meduri, J. Carpentier, N. Mansard, and L. Righetti, “Stagewise implementations of sequential quadratic programming for model-predictive control,” *Preprint*, 2023.
- [40] “Locomotion generation for parallel robots: a collection of ocp problems used to generate the examples of the paper - on github.” <https://github.com/LudovicDeMatteis/ClosedLoopMotion>.

## APPENDIX A FROM APPROXIMATE SERIAL MODEL TO CLOSED-KINEMATICS

As the leg is fully actuated and outside of singularities of the parallel linkage, every state trajectory of the *Approximate Serial* model can be casted to a trajectory of the *Closed-Kinematics* model with same serial state. A trajectory for the serial model is defined by a sequence of states  $x_s^{[k]} = (q_s^{*[k]}, \dot{q}_s^{*[k]})$  and of controls  $u_s^{[k]}$ . A trajectory for the closed model is similarly defined by the sequences  $x_c^{[k]} = (q_c^{[k]}, \dot{q}_c^{[k]})$  and  $u_c^{[k]}$ , where  $q_c^{[k]} = (q_s^{[k]} \quad q_l^{[k]})$  (respectively  $\dot{q}_c$ ), describing that the *Closed-Kinematics* model state includes the *Approximate Serial* model state. We propose to lift the *Approximate Serial trajectory* into a *Closed-Kinematics trajectory* by solving:

$$\begin{aligned} \min_{u_c^{[k]}, x_c^{[k+1]}} \quad & \frac{1}{2} \|q_s^{[k+1]} - q_s^{*[k+1]}\|^2 + \frac{1}{2} \|\dot{q}_s^{[k+1]} - \dot{q}_s^{*[k]}\|^2 \\ \text{s.t.} \quad & x_c^{[k+1]} = f_k(x_c^{[k]}, u_c^{[k]}) \end{aligned} \tag{21}$$

where the state  $x_c^{[k]}$  is known from the previous iteration (assuming  $x^{[0]}$  is known) and the targets are the expected values for the serial part of the state, given by the *Approximate Serial trajectory*. We can observe that problem (21) takes the form of a 1-step optimal control problem and can therefore be solved using the same solver as before.