High Accuracy Aerial Maneuvers on Legged Robots using Variational Integrator Discretized Trajectory Optimization

Scott Beck*,[1], Chuong Nguyen*,[1], Thai Duong[2], Nikolay Atanasov[3], and Quan Nguyen[1]

***Abstract*— Performing acrobatic maneuvers involving long aerial phases, such as precise dives or multiple backflips from significant heights, remains an open challenge in legged robot autonomy. Such aggressive motions often require accurate state predictions over long horizons with multiple contacts and extended flight phases. Most existing trajectory optimization (TO) methods rely on Euler or Runge-Kutta integration, which can accumulate significant prediction errors over long planning horizons. In this work, we propose a novel whole-body TO method using variational integration (VI) and full-body nonlinear dynamics for long-flight aggressive maneuvers. Compared to traditional Euler-based TO, our approach using VI preserves energy and momentum properties of the continuous-time system and reduces error between predicted and executed trajectories by factors of between $2-10$ while achieving similar planning time. We successfully demonstrate long-flight triple backflips on a quadruped A1 robot model and backflips on a bipedal HECTOR robot model for various heights and distances, achieving landing angle errors of only a few degrees. In contrast, TO with Euler integration fails to achieve accurate landings in equivalent circumstances, e.g., with landing angle errors greater than 90° for triple backflips. We provide an open-source implementation of our VI-discretized TO to support further research on accurate dynamic maneuvers for multi-rigid-body robot systems with contact: https://github.com/DRCL-USC/VI_discretized_TO**

I. Introduction

Acrobatic maneuvers on legged robots have drawn significant interest due to their notable advantages in traversing difficult terrain, wide gaps, and complex obstacles. These maneuvers have been successfully demonstrated using learning-based control [1]–[9], model-based control [10]–[16], trajectory optimization [17]–[23], and mixed-integer convex programming [24]. However, due to the underactuated flight phase (during which control has little effect on robot motion), achieving precise landing in such motions is still a particular challenge, requiring accurate whole-body coordination across various contact and mid-air phases. While stable landing is a multi-faceted problem and many tools such as dedicated landing controllers [25]–[28], specialized hardware [29], [30], and real-time updates to pre-impact body and joint position [31]–[34] contribute to success, one of the most critical factors is reduction of tracking errors in the body angles and rates. Increasing flight phase duration or aggressiveness of the aerial motion (e.g., multiple backflips) compounds this challenge, as state prediction errors accumulate quickly, causing the terminal state at landing to be highly sensitive to the state at takeoff.

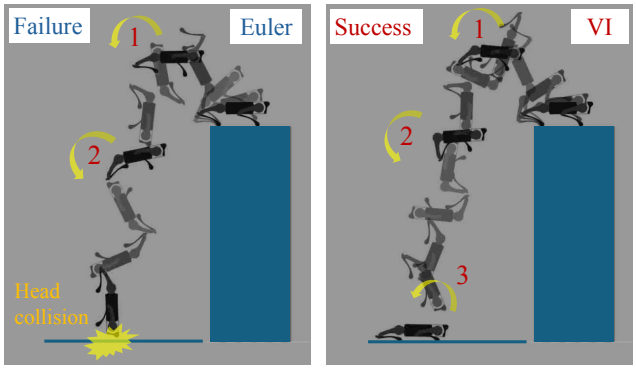

Fig. 1: **Triple backflips on A1 quadruped robot:** Comparison between Euler-based TO (left) and our VI-discretized TO (right). The numbers in the figures indicates how many successful backflips are achieved. Supplemental video: https://youtu.be/z4HzWtGu9RM

Trajectory optimization (TO) is increasingly used for acrobatic maneuvers due to its capability to optimize over multiple contact phases with nonlinear constraints [11], [17], [21], [26], [35]–[38]. For many classes of robot systems, reduced-order models such as spring-loaded inverted pendulum [39], single-rigid-body dynamics (SRBD) [21], [37], [38], or centroidal dynamics with kinematic constraints [40], [41] yield good results for maneuvers with short flight phases. However, a full-body nonlinear dynamics model is critical to successful long-flight motions [20], [22]. Moreover, common TO methods discretize the dynamics using first-order Euler integration for simplicity and computational efficiency. However, as Euler integration does not preserve system momentum or energy [42], it generates trajectories that violate these conservation laws and accumulate large discretization errors. Higher-order explicit Runge-Kutta schemes [43] can mitigate this error but suffer from potentially impractical solve time when used as a dynamics constraint in TO [44], and still inherently introduce artificial energy dissipation [45] arising from a non-physics informed discretization. In our prior work on bipedal [26] and quadruped jumping [20], we observed that energy drift from dynamics discretization can cause large landing angle errors in long and high jumps.

To address the challenges of accuracy in acrobatic motion planning with long aerial phases, we develop a transcription method that directly discretizes the underlying variational

This work was supported by USC Viterbi School of Engineering startup funds.

[1]These authors are with the Department of Aerospace and Mechanical Engineering, University of Southern California, USA:, e-mails: {sdbeck, vanchuong.nguyen, quann}@usc.edu.

[2]This author is with the Department of Computer Science, Rice University, Houston, TX 77005, USA, e-mail: thaiduong@rice.edu.

[3]This author is with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093, USA, e-mail: natanasov@ucsd.edu.

* These two authors contributed equally to this work

mechanics of the robot system. Such *variational integration* (VI) has long been used to improve simulation fidelity of physical systems, leveraging its symplectic form and conservation of physical quantities [46]. VI-based discretization preserves system momentum and, while fixed time step VI does not expressly enforce energy conservation [47], it eliminates the secular energy drift obseved in other integration schemes. This bounded-energy behavior of fixed-time VI enables various acrobatic motions where Euler integration fails. Because these conservation properties are intrinsic to the discretization and hold regardless of time step size, we can achieve a transcription that aligns more closely to the proper mechanics without adding collocation points and incurring commensurate compute time penalties.

In recent years, VI has been used in motion planning [42], [44], [48]–[51], though most applications focus on motions with no or limited flight time, and little acrobatic motion of the robot trunk (walking, horizontal jumping, etc.). Some recent work has approached highly dynamic motion using geometric integration similar to VI, but utilizes non-linear model predictive control (MPC) [52], wherein the planning horizon spans a fraction of the total target motion at a time.

In motions where the contact schedule keeps the robot fully actuated (or under-actuated phases are brief), real-time joint tracking controllers can introduce feedback torque to compensate for discrepancies in feed-forward torques that arise from discretization-induced energy drift during TO. While VI-discretization can be beneficial in such cases—as seen in [48]—we contend that the advantages that VI provides are far *more* impactful in long under-actuated phases, where system coordinates cannot be directly corrected via joint tracking and are therefore highly sensitive to whether the discretized TO predictions for proper lift-off conditions and in-flight joint trajectories align to the continuous system. Compared to [48], we eschew the contact-implicit formulation in lieu of a predetermined contact schedule to limit the size of the optimization problem, allowing more computation time for longer horizon optimization (important for long-flight phase maneuvers) rather than for contact evaluation. Planning with a given contact schedule is commonly used in Euler-based TO to realize aggressive long-flight motions [20], [22], [36], [53]. Our TO method, based on VI and a predetermined contact schedule, allows us to keep solve times low despite the complexity of the maneuvers.

The contributions of our work are summarized as follows.

- We develop a TO approach using fixed time step VI for full-body nonlinear robot dynamics, leveraging the conservation properties of VI in long horizon planning to realize long-flight aggressive maneuvers.
- We validate our approach extensively with various aggressive maneuvers such as multiple backflips on quadruped and bipedal robots with significant height and jump distances. Compared to TO using Euler integration, our VI-based method requires similar solving time, but can remarkably reduce the error between predicted and executed trajectories by factors of $2-10$, achieving landing angle errors of only a few degrees.

II. Preliminaries

We begin with a review of Lagrangian mechanics and the formulation of a family of variational integrators that discretize the variational principle of least action.

Consider a system with configuration $\boldsymbol{q} \in \mathcal{Q}$, where $\mathcal{Q}$ is the configuration manifold, and velocity $\dot{\boldsymbol{q}} \in T_{\boldsymbol{q}}\mathcal{Q}$, where $T_{\boldsymbol{q}}\mathcal{Q}$ is the tangent space to $\mathcal{Q}$ at $\boldsymbol{q}$. Let $T\mathcal{Q} = \{(\boldsymbol{q}, \dot{\boldsymbol{q}}) | \boldsymbol{q} \in \mathcal{Q}, \dot{\boldsymbol{q}} \in T_{\boldsymbol{q}}\mathcal{Q}\}$ denote the tangent bundle of $\mathcal{Q}$. Let $\boldsymbol{p} \in T^*_{\boldsymbol{q}}\mathcal{Q}$ denote the generalized momentum of the system, where $T^*_{\boldsymbol{q}}\mathcal{Q}$ is the cotangent manifold of $\mathcal{Q}$, and $T^*\mathcal{Q} = \{(\boldsymbol{q}, \boldsymbol{p}) | \boldsymbol{q} \in \mathcal{Q}, \boldsymbol{p} \in T^*_{\boldsymbol{q}}\mathcal{Q}\}$ is the cotangent bundle of $\mathcal{Q}$.

Lagrangian mechanics describes the system dynamics using a smooth function $\mathcal{L} : T\mathcal{Q} \to \mathbb{R}$, called the Lagrangian, and a Lagrangian force $\boldsymbol{f}_L : T\mathcal{Q} \times \mathcal{U} \to T^*\mathcal{Q}$ acting on the system with control parameter $\boldsymbol{u} \in \mathcal{U}$, which may in general include control torques, contact forces, damping, etc. In this formulation, the generalized momentum is related to the configuration variables and velocity as $\boldsymbol{p} = \frac{\partial \mathcal{L}}{\partial \dot{\boldsymbol{q}}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

The trajectories of a Lagrangian system satisfy the *Lagrange-d'Alembert Principle* [54, Ch. 8]:

$$\delta \int_0^T \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) dt + \int_0^T \boldsymbol{f}_L(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u})\, \delta \boldsymbol{q} dt = 0, \tag{1}$$

where the variation is induced by an infinitesimal variation $\delta\boldsymbol{q}(t)$ that vanishes at the endpoints $t = 0$ and $t = T$. Integration by parts applied to (1) leads to the *forced Euler–Lagrange* (EL) equations of motion:

$$\frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \boldsymbol{q}} - \frac{d}{dt}\left(\frac{\partial \mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}})}{\partial \dot{\boldsymbol{q}}}\right) + \boldsymbol{f}_L(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}) = 0. \tag{2}$$

For many systems, a standard choice of the Lagrangian is $\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathcal{K}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathcal{V}(\boldsymbol{q})$, where $\mathcal{K}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^\top \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}$ is the kinetic energy with generalized mass matrix $\boldsymbol{M}(\boldsymbol{q})$ and $\mathcal{V}(\boldsymbol{q})$ is the potential energy. In this case, the EL equations take the familiar form [55, Ch. 8]:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{f}_L(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}), \tag{3}$$

where $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are Coriolis and other conservative forces.

Traditional integration methods discretize the equations of motion in (3) directly. However, this usually does not preserve conservation laws captured by the continuous-time EL equations. In contrast, a variational integrator [46] discretizes the Lagrange-d'Alembert Principle by replacing the integrals in (1) with finite sums. Dividing the interval $[0, T)$ into N subintervals $[t_k, t_{k+1})$ for $k \in \{0, 1, ..., N-1\}$, such that $0 = t_0 < t_1 < \cdots < t_N = T$, we obtain:

$$\delta \sum_{k=0}^{N-1} \mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) + \sum_{k=0}^{N-1} \boldsymbol{f}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}, \boldsymbol{u}_k, \boldsymbol{u}_{k+1})\, \delta \boldsymbol{q} = 0, \tag{4}$$

where $\boldsymbol{q}_k = \boldsymbol{q}(t_k)$, $\boldsymbol{u}_k = \boldsymbol{u}(t_k)$ and $\mathcal{L}_d$ and $\boldsymbol{f}_d$ are, respectively, the discrete Lagrangian and discrete generalized force, approximating the contribution to the integral over $\Delta t_k = t_{k+1} - t_k$. While many choices are possible for the

discretization of $\mathcal{L}$, and $\boldsymbol{f}_L$, one popular choice—and the one employed in this work—is midpoint quadrature. Using $\underline{\boldsymbol{q}}_k = \frac{\boldsymbol{q}_{k+1}+\boldsymbol{q}_k}{2}$, $\underline{\dot{\boldsymbol{q}}}_k = \frac{\boldsymbol{q}_{k+1}-\boldsymbol{q}_k}{\Delta t_k}$, this yields:

$$\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) \approx \Delta t_k \mathcal{L}\left(\underline{\boldsymbol{q}}_k,\ \underline{\dot{\boldsymbol{q}}}_k\right), \tag{5}$$

$$\boldsymbol{f}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}, \boldsymbol{u}_k, \boldsymbol{u}_{k+1})\,\delta\boldsymbol{q} \approx \boldsymbol{f}^-_{d,k}\,\delta\boldsymbol{q}_k + \boldsymbol{f}^+_{d,k}\,\delta\boldsymbol{q}_{k+1}, \tag{6}$$

with "left" and "right" discrete forces, respectively, $\boldsymbol{f}^-_{d,k} = \frac{\Delta t_k}{2}\boldsymbol{f}_L(\boldsymbol{q}_k, \underline{\dot{\boldsymbol{q}}}_k, \boldsymbol{u}_k)$, $\boldsymbol{f}^+_{d,k} = \frac{\Delta t_k}{2}\boldsymbol{f}_L(\boldsymbol{q}_{k+1}, \underline{\dot{\boldsymbol{q}}}_k, \boldsymbol{u}_{k+1})$. Now, carrying through using the chain rule in (4):

$$\sum_{k=0}^{N-1} \left(D_1\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1})\,\delta\boldsymbol{q}_k + D_2\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1})\,\delta\boldsymbol{q}_{k+1}\right) + \sum_{k=0}^{N-1}\left(\boldsymbol{f}^-_{d,k}\,\delta\boldsymbol{q}_k + \boldsymbol{f}^+_{d,k}\,\delta\boldsymbol{q}_{k+1}\right) = 0, \tag{7}$$

where $D_1\mathcal{L}_d$ and $D_2\mathcal{L}_d$ denote the slot derivatives:

$$\begin{aligned} D_1\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) &= \frac{d}{d\boldsymbol{q}_k}\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) \\ &= \left(\frac{\Delta t_k}{2}\frac{\partial\mathcal{L}}{\partial\boldsymbol{q}} - \frac{\partial\mathcal{L}}{\partial\dot{\boldsymbol{q}}}\right)\Big|_{\underline{\boldsymbol{q}}_k,\ \underline{\dot{\boldsymbol{q}}}_k}, \end{aligned} \tag{8}$$

$$\begin{aligned} D_2\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) &= \frac{d}{d\boldsymbol{q}_{k+1}}\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) \\ &= \left(\frac{\Delta t_k}{2}\frac{\partial\mathcal{L}}{\partial\boldsymbol{q}} + \frac{\partial\mathcal{L}}{\partial\dot{\boldsymbol{q}}}\right)\Big|_{\underline{\boldsymbol{q}}_k,\ \underline{\dot{\boldsymbol{q}}}_k}. \end{aligned} \tag{9}$$

Noting that variations $\delta\boldsymbol{q}_k$ must vanish at the endpoints, i.e., $\delta\boldsymbol{q}_0 = \delta\boldsymbol{q}_N = 0$, and grouping the remaining terms ($k \in \{1, N-1\}$) by the index of $\delta\boldsymbol{q}_k$, (7) reduces to:

$$\sum_{k=1}^{N-1}\Big(D_2\mathcal{L}_d(\boldsymbol{q}_{k-1}, \boldsymbol{q}_k) + D_1\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) + \boldsymbol{f}^+_{d,k-1} + \boldsymbol{f}^-_{d,k}\Big)\,\delta\boldsymbol{q}_k = 0. \tag{10}$$

Since this must hold for all $\delta\boldsymbol{q}_k$, we conclude that:

$$D_2\mathcal{L}_d(\boldsymbol{q}_{k-1}, \boldsymbol{q}_k) + D_1\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) + \boldsymbol{f}^+_{d,k-1} + \boldsymbol{f}^-_{d,k} = 0 \tag{11}$$

for each term of the summation. Eq. (11) are known as the *Discrete Euler Lagrange* (DEL) equations. If $\boldsymbol{q}_{k-1}$ and $\boldsymbol{q}_k$ are known, then these equations can be used to propagate the state to $\boldsymbol{q}_{k+1}$. Via discrete Legendre transformation [46], the discrete momentum $\boldsymbol{p}_k$ of the system is defined as:

$$\boldsymbol{p}_k = D_2\mathcal{L}_d(\boldsymbol{q}_{k-1}, \boldsymbol{q}_k) + \boldsymbol{f}^+_{d,k-1}. \tag{12}$$

III. VI-Discretized Trajectory Optimization

In this section, we develop a TO approach using fixed time step VI for full-body nonlinear robot dynamics. The goal is to find a discretized sequence of reference states and controls that achieves the maneuvers under constraints such as robot dynamics, initial and final conditions, a predefined contact schedule, collision avoidance, friction cone, force saturation, and hardware limits. The proposed approach encodes VI for full-body non-linear dynamics constraints and is transcribed as multiple shooting with a predefined contact schedule.

A. Problem Statement

Inconveniently, the multi-rigid-body system dynamics described by the DEL equations in (11) require knowledge of both $\boldsymbol{q}_{k-1}$ and $\boldsymbol{q}_k$ to calculate $\boldsymbol{q}_{k+1}$. Instead, we define the robot state as $\boldsymbol{x}_k = (\boldsymbol{q}_k, \boldsymbol{p}_k) \in T^*\mathcal{Q}$ with the generalized coordinate $\boldsymbol{q}_k$, momentum $\boldsymbol{p}_k$, and control input $\boldsymbol{u}_k$. We derive the robot dynamics in the form of $d\big(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{x}_{k+1}\big) = 0$ in Sec. III-B, allowing us to propagate the state $\boldsymbol{x}_{k+1}$ solely from the current state $\boldsymbol{x}_k$ and control $\boldsymbol{u}_k$. The control input $\boldsymbol{u}_k$ represents both the contact forces and joint torques being applied on the system.

Given an initial condition $\boldsymbol{x}_0 \in T^*\mathcal{Q}$, the goal of trajectory optimization is to find a state-control trajectory $(\boldsymbol{x}_{0:N}, \boldsymbol{u}_{0:N-1})$ for the robot system, that minimizes:

$$\min_{\boldsymbol{x}_{0:N}, \boldsymbol{u}_{0:N-1}} \quad J\big(\boldsymbol{x}_N\big) + \sum_{k=0}^{N-1} w\big(\boldsymbol{x}_k, \boldsymbol{u}_k\big) \tag{13a}$$

$$\text{s.t. } d\big(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{x}_{k+1}\big) = 0, \tag{13b}$$

$$\phi\big(\boldsymbol{x}_k, \boldsymbol{u}_k\big) = 0,\ \psi\big(\boldsymbol{x}_k, \boldsymbol{u}_k\big) \le 0, \tag{13c}$$

$$\phi_N\big(\boldsymbol{x}_N\big) = 0,\ \psi_N\big(\boldsymbol{x}_N\big) \le 0,\ k = 1 \ldots N-1, \tag{13d}$$

where the functions $w(\boldsymbol{x}_k, \boldsymbol{u}_k)$ and $J(\boldsymbol{x}_N)$ specify the stage cost and terminal cost, respectively, of the trajectory. The function ϕ captures equality constraints on the initial and final states, and a predefined contact schedule. The function ψ captures inequality constraints related to hardware limits, friction cones, and collision avoidance constraints. The functions ϕ and ψ are described in details in Sec. III-C.

B. Full-body Dynamics using VI

Traditional Euler or Runge-Kutta integration in TO transcription relies directly on (3), enforcing the relation between the position-velocity pair $(\boldsymbol{q}_k, \dot{\boldsymbol{q}}_k)$ at t_k and the corresponding pair $(\boldsymbol{q}_{k+1}, \dot{\boldsymbol{q}}_{k+1})$ at t_{k+1}. Eq. (11), rather, relates generalized coordinates at t_{k-1} and t_k, $(\boldsymbol{q}_{k-1}, \boldsymbol{q}_k)$ to those at t_{k+1}, $\boldsymbol{q}_{k+1}$. Defining VI in position-momentum form makes the two approaches analogous, i.e., forward propagation using only information available at t_k, as follows:

$$\boldsymbol{p}_{k+1} = D_2\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) + \boldsymbol{f}^+_{d,k}, \tag{14a}$$

$$\boldsymbol{p}_k = -D_1\mathcal{L}_d(\boldsymbol{q}_k, \boldsymbol{q}_{k+1}) - \boldsymbol{f}^-_{d,k}, \tag{14b}$$

where Eq. (14b) is derived from (11) and (12).

We define $\boldsymbol{u} = (\boldsymbol{\tau}, \boldsymbol{f}_c)$, where $\boldsymbol{\tau}$ denotes the applied joint torques between the rigid bodies, and $\boldsymbol{f}_c = [\boldsymbol{f}_c^{(1)}, ..., \boldsymbol{f}_c^{(n)}]$ denotes ground reaction forces (GRF) between the rigid bodies and the terrain for n contact points. Then, the discrete forces $\boldsymbol{f}^{\pm}_{d,k}$ in (14a) and (14b) are approximated by (6), with a continuous force:

$$\boldsymbol{f}_L(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}) = \boldsymbol{\tau} - \boldsymbol{K}^j_{fric}\dot{\boldsymbol{q}} + \boldsymbol{J}(\boldsymbol{q})^T\boldsymbol{f}_c, \tag{15}$$

where $\boldsymbol{K}^j_{fric}$ is a vector of damping coefficients for each coordinate, modelling friction torques, and $\boldsymbol{J}(\boldsymbol{q})$ is the concatenated Jacobian for each of the contact points. For floating-base systems (e.g. legged robots), we define $\boldsymbol{q} = (\boldsymbol{q}_b, \boldsymbol{q}_J)$, with body coordinate $\boldsymbol{q}_b$ and joint angles $\boldsymbol{q}_J$.

Eq. (14), evaluated using (6), (8), (9) and midpoint quadrature, defines VI as an implicit relation between the current state $\mathbf{x}_k = (\boldsymbol{q}_k, \boldsymbol{p}_k)$, control input $\boldsymbol{u}_k$ and the subsequent state $\mathbf{x}_{k+1} = (\boldsymbol{q}_{k+1}, \boldsymbol{p}_{k+1})$. This is encoded as the dynamics constraints in Eq. (13b) of the TO problem (13).

C. Trajectory Optimization

The complete trajectory optimization problem is formulated as a NLP of the following form:

$$\min_{\boldsymbol{x}_{0:N}, \boldsymbol{u}_{0:N-1}} \epsilon_N \boldsymbol{e}_N^T \boldsymbol{e}_N + \sum_{k=1}^{N-1} \epsilon_q \boldsymbol{e}_k^T \boldsymbol{e}_k + \epsilon_\tau \boldsymbol{\tau}_k^T \boldsymbol{\tau}_k \quad (16a)$$

s.t. Full-body dynamics constraints (14a), (14b), (16b)

$$\text{Initial condition: } (\boldsymbol{q}, \boldsymbol{p})\big|_{k=0} = (\boldsymbol{q}_0^d, \boldsymbol{p}_0^d), \quad (16c)$$

$$\text{Final condition: } (\boldsymbol{q}, \boldsymbol{p})\big|_{k=N} = (\boldsymbol{q}_N^d, \boldsymbol{p}_N^d), \quad (16d)$$

$$\text{Joint angle: } \boldsymbol{q}_{\boldsymbol{J},min} \le \boldsymbol{q}_{\boldsymbol{J},k} \le \boldsymbol{q}_{\boldsymbol{J},max}, \quad (16e)$$

$$\text{Joint velocity: } \left| \frac{\boldsymbol{q}_{\boldsymbol{J},k+1} - \boldsymbol{q}_{\boldsymbol{J},k}}{\Delta t_k} \right| \le \dot{\boldsymbol{q}}_{\boldsymbol{J},max}, \quad (16f)$$

$$\text{GRF: } \boldsymbol{f}_{c,min}^{(i),z} \cdot \delta_{c,k}^{(i)} \le \boldsymbol{f}_{c,k}^{(i),z} \le \boldsymbol{f}_{c,max}^{(i),z} \cdot \delta_{c,k}^{(i)}, \quad (16g)$$

$$\text{Friction: } |\boldsymbol{f}_{c,k}^{(i),x} / \boldsymbol{f}_{c,k}^{(i),z}| \le \mu, \quad (16h)$$

$$\text{Foot Stationarity: } |\boldsymbol{r}_f^{(i)}(\boldsymbol{q}_k) - \boldsymbol{r}_f^{(i)}(\boldsymbol{q}_0^d)| \cdot \delta_{c,k}^{(i)} = 0, \quad (16i)$$

$$\text{Collision avoidance constraints: } g(\boldsymbol{q}_k) \le 0, \quad (16j)$$

Motor dynamics and torque constraints:

$$\left|V(\boldsymbol{\tau}_k, \dot{\boldsymbol{q}}_{J,k})\right| = \left|\rho \boldsymbol{\tau}_k + \sigma \dot{\boldsymbol{q}}_k\right| \le V_{max}, \quad (16k)$$

$$|\boldsymbol{\tau}_k| \le \boldsymbol{\tau}_{max}, \quad (16l)$$

where $\boldsymbol{\tau}_k$ is joint torque over time interval $[t_k, t_{k+1}]$, $(\boldsymbol{q}_0^d, \boldsymbol{p}_0^d)$ and $(\boldsymbol{q}_N^d, \boldsymbol{p}_N^d)$ are the desired initial and final states, $\boldsymbol{r}_f^{(i)}(\boldsymbol{q})$ is the location of the i^{th} contact point, μ is the friction coefficient, and $\epsilon_q, \epsilon_\tau, \epsilon_N$ are cost function weights. A standing pose for the robot is defined via a nominal joint configuration $\boldsymbol{q}_{\boldsymbol{J},k}^{nom}$, and the error terms $\boldsymbol{e}_k = \boldsymbol{q}_{\boldsymbol{J},k} - \boldsymbol{q}_{\boldsymbol{J},k}^{nom}$ are added to the stage cost to prevent the joint angles from varying arbitrarily during jumping motions. The error $\boldsymbol{e}_N$ denotes this joint angle deviation at $k = N$. The trajectory $\boldsymbol{q}_{\boldsymbol{J},k}^{nom}$ is also used as an initial guess for the NLP solver.

The contact force at contact point i in x and z directions are denoted as $\boldsymbol{f}_{c,k}^{(i),x}, \boldsymbol{f}_{c,k}^{(i),z}$, satisfying the friction constraint (16h). The contact schedule for the i^{th} contact point, $\delta_{c,k}^{(i)}$, is defined *a priori* as a mapping from time step k to $\{0, 1\}$—mapping to 1 if the point has scheduled contact with the terrain, and to 0 otherwise.[1] Eq. (16i) enforces that any contact point in ground contact should not move from its initial position during that contact—once it leaves the ground, $\delta_{c,k}^{(i)}$ is zero and this constraint is satisfied regardless of $\boldsymbol{r}_f^{(i)}$. Collision avoidance constraints (16j) are designed to prevent collision between robot links and to guarantee all robot components have clearance to obstacles.

[1]Often $\delta_{c,k}^{(i)}$ may be represented by M phase durations $\{\theta_1, ..., \theta_M\}$, where each θ represents a duration during which $\delta_{c,k}^{(i)}$ is constant, with shifts in contact only happening between phases.

To accomplish aggressive maneuvers, the joint torque and velocity often rapidly reach motor limits and onboard power supply. Thus, we include the motor dynamics constraints (MDC) in TO to represent true system limits (16k). The MDC represents inherent torque-velocity relationships, expressed by the estimated voltage on motors V, with voltage supply V_{max}. In (16k), we have $\rho = r_i/(K_\tau g_r)$, $\sigma = K_v g_r$, where r_i is the resistance of the motor windings, g_r is gear ratio, K_v and K_τ are motor velocity and torque constants.

IV. Evaluation

This section evaluates the effectiveness of our VI TO approach on various robot platforms: 1) a double compound pendulum (Sec. IV-A), validating and illustrating the benefits of VI prediction accuracy and conservation properties; 2) a Unitree A1 quadruped [56] performing triple backflips (Sec. IV-B); and 3) a HECTOR bipedal robot [57] performing backflips (Sec. IV-B). Examples with other systems are provided in our repository[2]. A video of the experiments is included in the supplementary material.

We utilize MATLAB-Simulink-Simscape to build simulation platforms and derive the full-body non-linear dynamics models for both Euler integration and VI. Our algorithm is implemented in MATLAB R2023b, and the NLP is solved using the CasADi toolbox [58] with the IPOPT solver. In all experiments, initial guesses provided to the NLP solver were consistent between Euler and VI trials.

A. Motivating Example: Double Pendulum

While most real multi-rigid-body robots have complex full-body dynamics, we first consider an intuitive system to validate the advantages in energy and prediction behaviors of VI over Euler integration. We consider a double pendulum with $m_1 = m_2 = 1$ kg, $L_1 = L_2 = 1$ m, $I_1 = I_2 = \frac{1}{12}$ kg·m^2 (Fig. 2). TO was performed using both Euler and VI methods to swing the pendulum from the fully downward position to fully upward, with $\Delta t_k = 10$ ms, $T = 2.5$ s.

The control trajectories provided by TO were used as feed-forward torques in a finely discretized simulation ($\Delta t_{k,\,sim} \ll \Delta t_k$). The effectiveness of these torques alone in achieving the motion indicates how well the transcription captures the true dynamics. Fig. 2 shows that, despite some residual error, the VI-discretized TO closely tracks the desired trajectory and flips the pendulum, while the Euler-discretized TO fails to reach the target state.

The accurate tracking performance of our VI-discretized TO, observed in Fig. 2, is attributed to VI's accurate long-term state predictions, enabled by preserving the system energy. We examine the free-response energy behavior ($\tau_j = 0$) of the system under Euler integration and VI. With only gravity acting on the system, the total energy in VI is conserved as expected, while Euler integration rapidly diverges from the initial energy, deviating by 3.1 J (28% of initial energy) after $2.5s$ (Fig. 3).

This excess energy disrupts the behavior of the pendulum system, causing the discrepancy between the reference and

[2]https://github.com/DRCL-USC/VI_discretized_TO

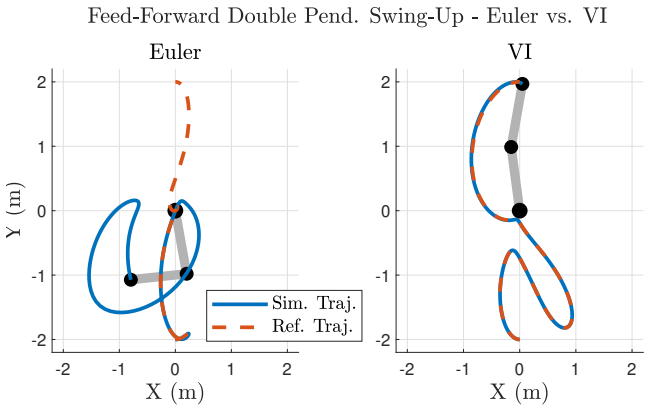

Fig. 2: **Double pendulum swing-up:** Comparison between feed-forward simulations of double pendulum swing-up using Euler-based TO (left) and VI TO (right). Expected TO reference trajectories shown as dotted line.

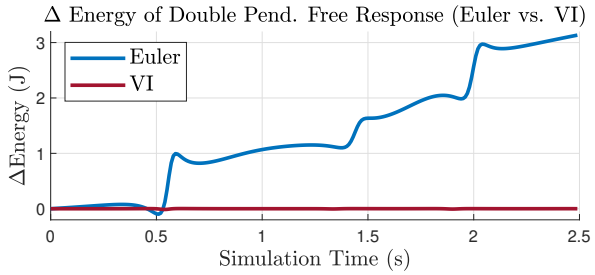

Fig. 3: **Double pendulum energy:** Deviation of system energy for free response of double pendulum for Euler method and VI.

simulated trajectories in Fig. 2. Intuitively, the Euler-based TO expects more energy in the system than the actual amount added by the applied torques. Thus, the required torque to achieve a task is under-estimated, leading to failures. In contrast, the VI approach deviates from the initial energy by only $0.014\ J$ (0.12%) over the planning time $T = 2.5s$, tracking the expected trajectory more successfully. In the next section, we illustrate that similar trends and behaviors are observed in Euler-TO generated trajectories for bipedal and quadruped robots, even when joint tracking controllers are used in addition to feed-forward torques.

B. Maneuvers on Legged Robots

In this section, we demonstrate single and multiple backflip tasks on simulated bipedal HECTOR and quadrupedal Unitree A1 robots, with different jump heights and distances, summarized in Tab. I. For each task, five pairs of target jump distance and platform height were randomly generated. For HECTOR, contact timings denote {heel-toe contact, flight phase}, and, for A1, contact timings denote {all-foot contact, rear-foot contact, flight phase}.

We used a joint PD controller to track reference joint trajectories obtained from TO (16). Given the feedback joint trajectory $\{\boldsymbol{q}_{J,k}, \dot{\boldsymbol{q}}_{J,k}\}$, desired joint angle $\boldsymbol{q}^d_{J,k}$, velocity $\dot{\boldsymbol{q}}^d_{J,k} \triangleq \left|(\boldsymbol{q}^d_{\boldsymbol{J},k+1} - \boldsymbol{q}^d_{\boldsymbol{J},k})/\Delta t_k\right|$, and feed-forward joint torque $\boldsymbol{\tau}^d_{J,k}$, the controller executes a total torque command as:

$$\boldsymbol{\tau}^{cmd}_{J,k} = \boldsymbol{\tau}^d_{J,k} + \boldsymbol{K}_p(\boldsymbol{q}^d_{J,k} - \boldsymbol{q}_{J,k}) + \boldsymbol{K}_d(\dot{\boldsymbol{q}}^d_{J,k} - \dot{\boldsymbol{q}}_{J,k}). \quad (17)$$

1) Accuracy of Trajectory Prediction: The resulting robot trunk center of mass position, trunk angle, and foot position

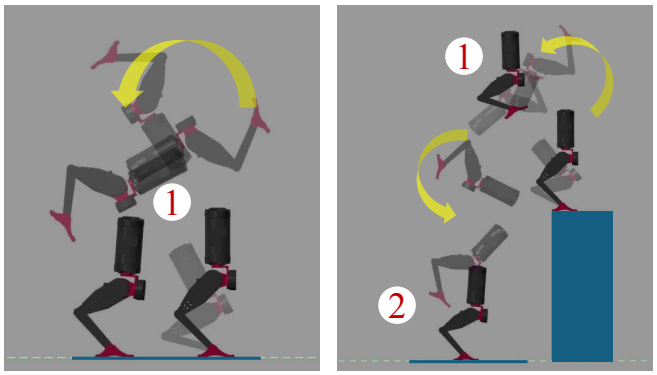

Fig. 4: **Bipedal backflips on HECTOR:** With our approach, the robot performs a backflip on flat ground (left) and double backflips from 1 m platform, with landing angle errors of several degrees, $\Delta t_k = 5$ ms.

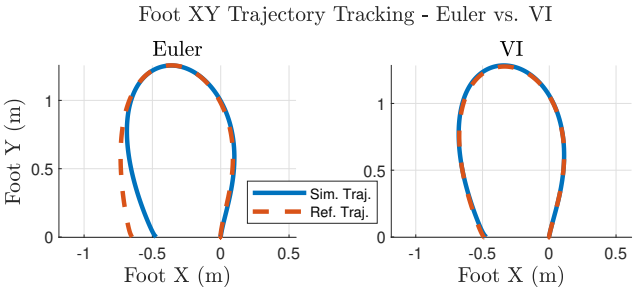

Fig. 5: **Bipedal landing target tracking:** Simulated and reference XY trajectories of toe contact point for HECTOR backflip, showing superior tracking for VI transcription.

(toe location for bipedal robot, front foot for quadruped) at landing were recorded and compared to those predicted by the reference trajectory. The error between predicted and simulated values serves as a metric of how well the robot was able to execute the output trajectory. For each combination of maneuver and discretization, these errors were used to generate a set of root mean square error (RSME) values.

Representative results comparing Euler- and VI-discretized TO in the quadruped triple backflip task are shown in Fig. 1. Results for a single backflip from flat ground and a double backflip from an elevated platform, both using VI-discretized TO, are shown in Fig. 4. Fig. 5 shows a representative comparison of how the foot position of the robot tracks a corresponding reference trajectory.

We observe that the terminal error in the trunk centroid location, foot position, and—most critically for successful landing—trunk angle are reduced for all three maneuvers when using VI in place of Euler integration. For $\Delta t_k = 10$ ms, terminal trunk angle errors are reduced from $60°$ to $23°$ for the single backflip, from $45°$ to $21°$ for the double backflip, and from $98°$ (a head-on collision with the ground) to $12°$ for the quadruped triple backflip. Note that for the bipedal single and double backflip, refining Δt_k to 5 ms results in terminal trunk angle RMSE of $< 5°$ and $< 1°$, respectively, with VI, while remaining at $> 20°$ error when using Euler integration with the refined Δt_k.

Fig. 6 evaluates the solving time and RMSE as a function of the number of knot points N for different integrators. Overall, increasing N (i.e., smaller Δt_k) reduces the RMSE but increases the solving time T_{IPOPT}. Notably, we observe

TABLE I: Comparison of Euler and VI schemes for various biped and quadruped backflip tasks. Five trajectories with different heights and distances for each task were evaluated, and terminal errors in trunk coordinates, foot position, and solve time were recorded.

System *Task*	Platform Height (m)	Jump Length (m)	Contact timings (ms)	**Integrators**	Trunk Errors $\{e_x, e_y, e_\theta\}$	Foot Error (cm)	IPOPT solve time (s)	Avg. comp. time per iter. (ms)
Biped (HECTOR) *Single Backflip*	0	0.0 - 0.5	{650 , 750}	Euler VI	0.7, 11, 60 0.7, 5.8, 23	45 20	5.81 ± 1.23 1.71 ± 0.21	18.8 25.9
Biped (HECTOR) *Double Backflip*	1.25 - 2.0	0.5 - 1.0	{750 , 900}	Euler VI	5.1, 20, 45 0.6, 4.6, 21	31 20	3.38 ± 0.45 2.85 ± 0.53	21.7 31.6
Quadruped (A1) *Triple Backflip*	2 - 2.5	1 - 1.5	{500 , 300 , 900}	Euler VI	20.6, 10.3, 98 9.7, 3.1, 12.3	18.3 9.4	4.51 ± 0.42 5.05 ± 0.63	24.1 31.5

TABLE II: Motor and on-board battery parameters of quadruped A1 and HECTOR bipedal robot

Parameter	Value	Units
(A1) Max Joint Torque Thigh & Calf	33.5, 33.5	Nm
(HECTOR) Max Joint Torque Thigh & Ankle	33.5, 33.5	Nm
(HECTOR) Max Joint Torque Calf	51	Nm
Max Joint Speed	21	Rad/s
Max Battery Voltage	21.5	V

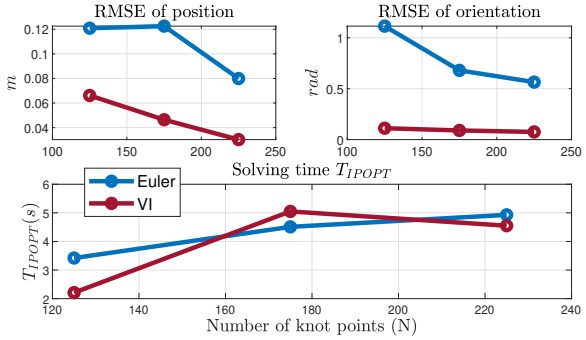

Fig. 6: **Comparison:** Solving time and RMSE with varying numbers of knot points for different integrators, observed from quadruped triple backflips.

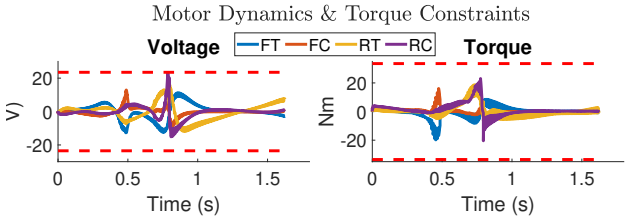

Fig. 7: Voltage and torque estimated from the A1 robot's motors for triple backflips. Dotted red lines denote voltage and torque limits. The notation FT, FC, RT, RC represents front thigh, front calf, rear thigh, rear calf.

that our VI-discretized method can achieve a similar solving time to Euler, while significantly reducing RMSE by 2-3 times in position and 5-10 times in orientation.

To validate that the execution of these trajectories obey hardware constraints (16k), (16l) with values given in Table II, the estimated motor voltage and actual torque during the quadruped triple backflips were plotted in Fig. 7.

2) Optimization Solve Time: The solve time of the TO was recorded for each trial, with averages reported in Table I. Both the time reported by IPOPT at solve termination and the average time per IPOPT iteration are listed.

Intuitively, we would expect increased computation time for VI vs. Euler discretized TO due to the midpoint VI being a second-order implicit integrator. However, while we do observe non-trivial increases in the time to transcribe constraints into the NLP and moderate increases in average compute time per iteration, the average time reported by IPOPT to solve the NLP was found to be similar to or *lower* for VI than Euler for both robots (Tab. I, Fig. 6). While we expected VI properties to mitigate solve time increases resulting from the higher order scheme, this *reduction* in optimization time was surprising and warrants discussion.

It is challenging to isolate the effect of any single factor on compute time of an NLP solver, but we suggest that the VI's properties—adherence to physical conservation laws and second-order accuracy—contribute to tractability. These properties may lead to gradients that more effectively move non-feasible guesses to feasible trajectories or maintain feasibility while minimizing the cost function, reducing the number of required iterations and offseting the increased time per iteration. With VI, it is perhaps less likely than with Euler methods to get "stuck" in non-feasible regimes due to accumulated energy or discretization errors. For tasks sensitive to the energy drift of Euler integration, these factors might reduce NLP solver time despite increased complexity.

C. Open-Source Code

The implementation used to generate the results in this work, along with results for several other physical systems, is provided as an open-source project on GitHub: https://github.com/DRCL-USC/VI_discretized_TO. We provide tutorials on running the examples and creating new systems to support the research community in exploring the practical benefits of VI-discretized TO.

V. Conclusions

This work showed that performing transcription in trajectory optimization for multi-rigid-body robots using variational integration enables accurate trajectory prediction over long planning horizons by preserving the momentum and energy of the continuous-time system. Extensive evaluations on quadruped and biped robot models showed that the ability to performs long-horizon optimization efficiently enables aggressive maneuvers with long flight periods. Future work includes implementation of these trajectories on hardware, extension of the TO horizon to include the landing phase, and assessing the benefits of VI-discretization when paired with MPC — including developing a VI-powered nonlinear MPC controller. Comparisons between VI discretization in TO with other non-Euler integrators such as the Leapfrog integrator or Runge-Kutta methods are also planned.

REFERENCES

[1] C. Nguyen, A. Altawaitan, T. Duong, N. Atanasov, and Q. Nguyen, "Variable-frequency model learning and predictive control for jumping maneuvers on legged robots," *IEEE Robotics and Automation Letters*, vol. 10, no. 2, pp. 1321–1328, 2025.

[2] G. Bellegarda, C. Nguyen, and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *Robotics and Autonomous Systems*, 2024.

[3] V. Atanassov, J. Ding, J. Kober, I. Havoutis, and C. D. Santina, "Curriculum-based reinforcement learning for quadrupedal jumping: A reference-free design," *arXiv:2401.16337v2*, 2024.

[4] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11443–11450, 2024.

[5] Y. Yang, G. Shi, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "CAJun: Continuous adaptive jumping using a learned centroidal controller," *Conference on Robot Learning (CoRL)*, 2023.

[6] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Robust and versatile bipedal jumping control through reinforcement learning," *Robotics: Science and Systems*, 2023.

[7] L. Krishna, N. Sobanbabu, and Q. Nguyen, "OGMP: Oracle guided multimodal policies for agile and versatile robot control," *arXiv preprint arXiv:2403.04205*, 2024.

[8] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Conference on Robot Learning (CoRL)*, 2023.

[9] C. Zhang, W. Xiao, T. He, and G. Shi, "Wococo: Learning whole-body humanoid control with sequential contacts," 2024.

[10] Z. He, J. Wu, J. Zhang, S. Zhang, Y. Shi, H. Liu, L. Sun, Y. Su, and X. Leng, "CDM-MPC: An integrated dynamic planning and control framework for bipedal robots jumping," *IEEE Robotics and Automation Letters*, 2024.

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

[12] H.-W. Park, P. M. Wensing, and S. Kim, "Jumping over obstacles with mit cheetah 2," *Robotics and Autonomous Systems*, vol. 136, p. 103703, 2021.

[13] Y. Ding, A. Pandala, C. Li, Y.-H. Shin, and H.-W. Park, "Representation-free model predictive control for dynamic motions in quadrupeds," *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1154–1171, 2021.

[14] J. Norby, Y. Yang, A. Tajbakhsh, J. Ren, J. K. Yim, A. Stutt, Q. Yu, N. Flowers, and A. M. Johnson, "Quad-sdk: Full stack software framework for agile quadrupedal locomotion," in *2022 ICRA Workshop on Legged Robots*, 2022.

[15] C. Mastalli, W. Merkt, G. Xin, J. Shim, M. Mistry, I. Havoutis, and S. Vijayakumar, "Agile maneuvers in legged robots: a predictive control approach," *arXiv preprint arXiv:2203.07554*, 2022.

[16] C. Nguyen, L. Bao, and Q. Nguyen, "Mastering agile jumping skills from simple practices with iterative learning control," *arXiv preprint arXiv:2408.02619*, 2024.

[17] Z. Zhou, B. Wingo, N. Boyd, S. Hutchinson, and Y. Zhao, "Momentum-aware trajectory optimization and control for agile quadrupedal locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7755–7762, 2022.

[18] F. Ruscelli, A. Laurenzi, N. G. Tsagarakis, and E. M. Hoffman, "Horizon: A trajectory optimization framework for robotic systems," *Frontiers in Robotics and AI*, vol. 9, p. 899025, 2022.

[19] L. Tiziani, Y. Zhang, F. Dellaert, and F. L. Hammond, "Factor graph-based trajectory optimization for a pneumatically-actuated jumping robot," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9870–9876, 2021.

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

[21] M. Chignoli and S. Kim, "Online trajectory optimization for dynamic aerial motions of a quadruped robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7693–7699, 2021.

[22] C. Nguyen and Q. Nguyen, "Contact-timing and trajectory optimization for 3D jumping on quadruped robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[23] G. Bellegarda, M. Shafiee, M. E. Ozberk, and A. Ijspeert, "Quadruped-frog: Rapid online optimization of continuous quadruped jumping," *arXiv preprint arXiv:2403.06954v1*, 2024.

[24] Y. Ding, C. Li, and H.-W. Park, "Kinodynamic motion planning for multi-legged robot jumping via mixed-integer convex program," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3998–4005, IEEE, 2020.

[25] S. Wang, W. Chi, J. Zuo, Q. Zhou, K. Chen, S. Yang, L. Xiang, and Y. Zheng, "Online multi-phase trajectory generation for compliant landing control of quadruped robots," in *2023 21st International Conference on Advanced Robotics (ICAR)*, pp. 168–175, 2023.

[26] J. Li, O. Kolt, and Q. Nguyen, "Continuous dynamic bipedal jumping via adaptive-model optimization," *arXiv preprint arXiv:2404.11807*, 2024.

[27] F. Roscia, M. Focchi, A. D. Prete, D. G. Caldwell, and C. Semini, "Reactive landing controller for quadruped robots," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7210–7217, 2023.

[28] S. H. Jeon, S. Kim, and D. Kim, "Online optimal landing control of the mit mini cheetah," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 178–184, 2022.

[29] Y. Tang, J. An, X. Chu, S. Wang, C. Y. Wong, and K. W. Samuel Au, "Towards safe landing of falling quadruped robots using a 3-dof morphable inertial tail," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, p. 1141–1147, IEEE, May 2023.

[30] F. Roscia, A. Cumerlotti, A. Del Prete, C. Semini, and M. Focchi, "Orientation control system: Enhancing aerial maneuvers for quadruped robots," *Sensors*, vol. 23, p. 1234, Jan. 2023.

[31] J. Jo, G. Park, and Y. Oh, "Robust landing stabilization of humanoid robot on uneven terrain via admittance control and heel strike motion," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2994–3000, 2021.

[32] V. Kurtz, H. Li, P. M. Wensing, and H. Lin, "Mini cheetah, the falling cat: A case study in machine learning and trajectory optimization for robot acrobatics," 2022.

[33] N. Rudin, H. Kolvenbach, V. Tsounis, and M. Hutter, "Cat-like jumping and landing of legged robots in low gravity using deep reinforcement learning," *IEEE Transactions on Robotics*, vol. 38, p. 317–328, Feb. 2022.

[34] J. Zhang, J. Shen, Y. Liu, and D. Hong, "Design of a jumping control framework with heuristic landing for bipedal robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8502–8509, 2023.

[35] K. Wang, G. Xin, S. Xin, M. Mistry, S. Vijayakumar, and P. Kormushev, "A unified model with inertia shaping for highly dynamic jumps of legged robots," *Mechatronics*, vol. 95, 2023.

[36] M. Chignoli, "Trajectory optimization for dynamic aerial motions of legged robots," *MIT Libraries*, 2021.

[37] Z. Song, L. Yue, G. Sun, Y. Ling, H. Wei, L. Gui, and Y.-H. Liu, "An optimal motion planning framework for quadruped jumping," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Japan*, pp. 11366–11373, IEEE, 2022.

[38] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1560–1567, 2018.

[39] P. M. Wensing and D. E. Orin, "Development of high-span running long jumps for humanoids," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 222–227, IEEE, 2014.

[40] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 295–302, 2014.

[41] C. Li, Y. Ding, and H.-W. Park, "Centroidal-momentum-based trajectory generation for legged locomotion," *Mechatronics*, vol. 68, p. 102364, 2020.

[42] V. Duruisseaux, T. P. Duong, M. Leok, and N. Atanasov, "Lie group forced variational integrator networks for learning and control of robot systems," in *Learning for Dynamics and Control Conference*, 2023.

[43] V. Arendt, "Numerical methods, energy conservation, and a new method for particle motion in magnetic fields," *Mathematics and Computers in Simulation*, vol. 205, pp. 142–185, 2023.

[44] S. Ober-Bloebaum, O. Junge, and J. E. Marsden, "Discrete mechanics and optimal control: an analysis," 2008.

[45] F. Hu, M. Hussaini, and J. Manthey, "Low-dissipation and low-dispersion runge–kutta schemes for computational acoustics," *Journal of Computational Physics*, vol. 124, no. 1, pp. 177–191, 1996.

[46] J. E. Marsden and M. West, "Discrete mechanics and variational integrators," *Acta numerica*, vol. 10, pp. 357–514, 2001.
[47] H. Sharma, M. Patil, and C. Woolsey, "Energy-preserving variational integrators for forced lagrangian systems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 64, p. 159–177, Nov. 2018.
[48] Z. Manchester, N. Doshi, R. J. Wood, and S. Kuindersma, "Contact-implicit trajectory optimization using variational integrators," *The International Journal of Robotics Research*, vol. 38, no. 12-13, pp. 1463–1476, 2019.
[49] T. Fan, J. Schultz, and T. Murphey, "Efficient computation of higher-order variational integrators in robotic simulation and trajectory optimization," 2019.
[50] A. Havens and G. Chowdhary, "Forced variational integrator networks for prediction and control of mechanical systems," in *Learning for Dynamics and Control*, pp. 1142–1153, PMLR, 2021.
[51] T. Lee, M. Leok, and N. H. McClamroch, "Computational geometric optimal control of connected rigid bodies in a perfect fluid," in *Proceedings of the 2010 American Control Conference*, pp. 5985–5990, 2010.
[52] N. Csomay-Shanklin, V. D. Dorobantu, and A. D. Ames, "Nonlinear model predictive control of a 3D hopping robot: Leveraging Lie group integrators for dynamically stable behaviors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.
[53] M. Chignoli, D. Kim, E. Stanger-Jones, and S. Kim, "The mit humanoid robot: Design, motion planning, and control for acrobatic behaviors," in *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*, pp. 1–8, 2021.
[54] T. Lee, M. Leok, and N. H. McClamroch, "Global formulations of Lagrangian and Hamiltonian dynamics on manifolds. interaction of mechanics and mathematics," 2018.
[55] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, 2017.
[56] "Unitree a1 robot," https://www.unitree.com/products/a1/.
[57] J. Li, J. Ma, O. Kolt, M. Shah, and Q. Nguyen, "Dynamic loco-manipulation on hector: Humanoid for enhanced control and open-source research," 2023.
[58] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.