Autonomous Robots
https://doi.org/10.1007/s10514-022-10068-3

Locomotion generation for quadruped robots on challenging terrains via quadratic programming

Xinyang Jiang[1] · Wanchao Chi[1] · Yu Zheng[1] · Shenghao Zhang[1] · Yonggen Ling[1] · Jiafeng Xu[1] · Zhengyou Zhang[1]

Received: 3 November 2021 / Accepted: 29 September 2022

Abstract
This paper proposes a locomotion generation method for quadruped robots, which first computes an optimal trajectory of the robot's center of mass (CoM) and then its whole-body motion through inverse kinematics. As the core component, the computing of the CoM trajectory, which is parameterized as polynomials, is based on the robot's centroidal dynamics and it is observed that several terms in the centroidal dynamics are minor and can be omitted in the locomotion generation. Then, as a basic form of the proposed method, the CoM trajectory optimization is written as a quadratic programming (QP) problem for the case of given step sequences, timings and footholds. Furthermore, the uncertainty of the robot's CoM, which is described as a convex polyhedron around a nominal CoM position, and the reachability of the robot's feet, which is approximated as another convex polyhedron, can be added to the QP problem as linear inequality constraints. Ultimately, the planning of step sequences, timings, and footholds is all incorporated, leading to a single mixed-integer quadratic programming problem. Numerical and hardware experiments have been conducted and show that the proposed method can generate various walking motions for a quadruped robot to travel over challenging terrains.

Keywords Locomotion generation · Footstep planning · Quadruped robot · Centroidal dynamics · Quadratic programming

List of symbols

$i \in \mathbb{N}^+$	Index of a foot of the robot
$N_C \in \mathbb{N}^+$	Number of feet/contacts
$m \in \mathbb{R}^+$	Mass of the robot
$\boldsymbol{f}_i \in \mathbb{R}^3$	Contact force at contact i
$\boldsymbol{n}_i \in \mathbb{R}^3$	Contact normal at contact i
$\boldsymbol{r}_i \in \mathbb{R}^3$	Position of contact i
$\boldsymbol{p}_G \in \mathbb{R}^3$	Position of the robot's CoM
$\boldsymbol{\mathcal{L}} \in \mathbb{R}^3$	Angular momentum of the robot about its CoM
$\boldsymbol{f} \in \mathbb{R}^{3N_C}$	Total contact force $[\boldsymbol{f}_1^T \ \boldsymbol{f}_2^T \ \dots \ \boldsymbol{f}_{N_C}^T]^T$
$\boldsymbol{R}_0 \in SO(3)$	Orientation of the robot's base
$\boldsymbol{\omega}_0 \in \mathbb{R}^3$	Body angular velocity of the robot's base
$\boldsymbol{\mathcal{I}}_0 \in \mathbb{R}^{3\times3}$	Body inertia tensor of the robot's base
$\boldsymbol{p}_{G0} \in \mathbb{R}^3$	Initial position of the robot's CoM for a motion segment
$\boldsymbol{p}_t \in \mathbb{R}^3$	Change of the robot's CoM position over time t during a motion segment
$n_* \in \mathbb{N}^+$	Order of the polynomial representing x, y, or z coordinate of $\boldsymbol{p}_t$
$\boldsymbol{c}_* \in \mathbb{R}^{n_*+1}$	Coefficients of the polynomial representing x, y, or z coordinate of $\boldsymbol{p}_t$
$\boldsymbol{c} \in \mathbb{R}^{n_\Sigma}$	Total coefficients $[\boldsymbol{c}_x^T \ \boldsymbol{c}_y^T \ \boldsymbol{c}_z^T]^T$, where $n_\Sigma = n_x + n_y + n_z + 3$
$\boldsymbol{f}_k \in \mathbb{R}^{3N_C}$	Total contact force at time t_k
$\boldsymbol{v}_h \in \mathbb{R}^3$	Vertex h of a convex polyhedron centered at $\boldsymbol{p}_{G0}$
$\boldsymbol{f}_{hk} \in \mathbb{R}^{3N_C}$	Total contact force at sample time t_k when the robot's CoM is at $\boldsymbol{v}_h$
$N_i \in \mathbb{R}^3$	Number of candidate footholds for foot i
$\boldsymbol{r}_{ij} \in \mathbb{R}^3$	Candidate foothold j for foot i
$\beta_{ijk} \in \{0, 1\}$	Binary variable associated with $\boldsymbol{r}_{ij}$ to indicate its selection at time t_k
$\boldsymbol{f}_{ijk} \in \mathbb{R}^3$	Contact force at $\boldsymbol{r}_{ij}$ at time t_k

1 Introduction

Quadruped robots possess a balanced locomotion ability and have gained significant attention during the past decade.

✉ Yu Zheng
petezheng@tencent.com

[1] Tencent Robotics X, Tencent Binhai Building, Shenzhen, Guangdong Province, China

Published online: 18 October 2022

Compared with bipedal robots, they have more legs to form larger support areas in different shapes as needed in various locomotion scenarios, and each leg has a simpler structure and lighter weight. All of these facilitate the realization of stable and agile locomotion. Compared with wheeled robots, quadruped robots can traverse a wider diversity of terrains. Thanks to the persistent work by numerous researchers and engineers, several outstanding quadruped robots have been developed, such as BigDog, WildCat and Spot series from Boston Dynamics (Raibert et al., 2008), HyQ series from IIT (Semini et al., 2011), ANYmal from ETH (Hutter et al., 2016), Cheetah series from MIT (Bledt et al., 2018), Jueying series from DeepRobotics (2017), and Laikago and Aliengo from Unitree (2017).

With many legs and degrees of freedom (DoFs) as well as factors to be considered such as equilibrium conditions, footstep sequences, contact constraints, joint limits, and etc., locomotion generation for quadruped robots is a core but hard problem aimed at computing appropriate motion trajectories for every DoF of robots to traverse over various terrains. In this paper, following existing pioneering work, we propose a locomotion generation algorithm for quadruped robots based on quadratic programming (QP) and show its extensions to cover the uncertainty of the robot's center of mass (CoM) and the planning of step sequences, timings, and footholds. A literature review is given first as follows.

1.1 Related work

Both as floating-base robots, quadruped and bipedal robots have many similarities, such as dynamic models, equilibrium conditions, and contact constraints, so their locomotion generation methods often inspire or can be extended to each other. In the following literature review, therefore, we do not distinguish the types of robots that a method is applicable to or originally designed for and focus on the underlying mathematics and physics inherent in the methods. In addition, the whole-body motion of a legged robot is often obtained through two successive steps, namely computing the motion trajectories of its floating base or CoM and feet followed by computing all joint trajectories through inverse kinematics. Our review is focused on the first step, which is the core of locomotion generation.

Early methods for locomotion generation often use a simplified dynamics model, typically an inverted pendulum or a cart-table model, to relate the motion of the robot's CoM to the support and combine a relatively simple stability criterion, usually based on the zero moment point (ZMP) (Vukobratović & Borovac, 2004) or the capture point (CP) (Koolen et al., 2012; Pratt et al., 2006, 2012), to ensure the feasibility of the generated motion. The ZMP-based stability criterion requires the ZMP of a legged robot to always lie in the support area formed by the feet contacting the ground, by which a pair of physically consistent ZMP and CoM trajectories to follow given footholds can be computed (Kajita et al., 2003; Kanehiro et al., 2008; Nagasaka et al., 1999; Sugihara, 2008). The ZMP can also be used to verify the stability of legged robots in following planned body trajectories (Huang et al., 2001; Kuffner et al., 2002; Neo et al., 2007; Park & Youm, 2007) or as constraints in generating CoM trajectories (Bellicoso et al., 2017, 2018; Caron et al., 2017; Kalakrishnan et al., 2011; Mastalli et al., 2017, 2020b; Winkler et al., 2015, 2017a, b). On the contrary, CP-based methods generate foothold sequences for a robot to follow a desired CoM motion (Gehring et al., 2013) or recover from a push (Barasuol et al., 2013; Pratt et al., 2006, 2012).

Recently, the computing of the robot's CoM trajectory, with known or unknown step sequences, timings, and/or footholds, has been written as various optimization problems with different choices of variables, constraints, and objective functions and it is also known as trajectory optimization (TO). As the state of the art, we summarize such methods from three perspectives.

1.1.1 Trajectory representation

The CoM trajectory can be represented as polynomials in time (Bellicoso et al., 2017, 2018; Fernbach et al., 2018; Kalakrishnan et al., 2011; Mordatch et al., 2012; Qiu et al., 2011; Tonneau et al., 2018; Winkler et al., 2014, 2015, 2017a, b, 2018; Zheng et al., 2019) or states containing positions, velocities, accelerations, and/or even jerks of the CoM at discrete time knots (Aceituno-Cabezas et al., 2017, 2018; Carpentier et al., 2016; Dai & Tedrake, 2016; Kuindersma et al., 2016; Orsolino et al., 2018; Perrin et al., 2015; Ponton et al., 2016, 2018). In the former representation, the coefficients (Bellicoso et al., 2017, 2018; Kalakrishnan et al., 2011; Mordatch et al., 2012; Qiu et al., 2011; Winkler et al., 2017a, b, 2018) or control points (Fernbach et al., 2018; Tonneau et al., 2018; Zheng et al., 2019) of polynomials are the primary variables to be determined in TO and the continuity is often imposed as linear equality constraints on the polynomials describing two neighboring motion segments at their conjunction. In the latter one, the states as well as other values, such as contact forces, are unknowns and time integration constraints relating the states at every pair of successive time knots need be added, which are linear equality constraints if time knots or motion phase durations are given (Aceituno-Cabezas et al., 2017, 2018; Carpentier et al., 2016; Dai & Tedrake, 2016; Perrin et al., 2015; Ponton et al., 2016, 2018) or nonlinear if they are also unknowns (Kuindersma et al., 2016; Orsolino et al., 2018) since the time integration requires the multiplication of velocity, acceleration, and jerk by time.

1.1.2 Trajectory feasibility

To check if a motion is physically feasible for a robot, one may reduce the robot to simple models, such as inverted pendulums, based on which the feasibility check can be straightforward. However, simple models do not truly reflect the motion capability of a robot with multiple joints. Instead, one may use the robot's whole-body dynamics (Neunert et al., 2017, 2018; Mastalli et al., 2020a), but it involves more complex computation and can be costly to be incorporated into a TO problem. State-of-the-art methods mostly use the centroidal dynamics of robots with consideration of contact force constraints (Aceituno-Cabezas et al., 2017, 2018; Carpentier et al., 2016; Dai & Tedrake, 2016; Fernbach et al., 2018; Grandia et al., 2019; Kuindersma et al., 2016; Ponton et al., 2016, 2018; Qiu et al., 2011; Tonneau et al., 2018; Winkler et al., 2018; Zheng et al., 2019). The robot's centroidal dynamics relates the translational motion of the CoM and the overall angular momentum change of the robot to the contact forces and the criterion requires that the gravito-inertia wrench of the robot during a motion should be counterbalanced by feasible contact forces (Bretl & Lall, 2008; Hirukawa et al., 2006; Saida et al., 2003). The contact force constraint typically includes the friction constraint, which defines a circular cone or often is approximated by a pyramidal cone limiting the direction of contact force. In some TO methods, contact forces are considered as unknowns and the friction constraint is added as linear or conic inequality constraints (Grandia et al., 2019; Ponton et al., 2016, 2018; Winkler et al., 2018) or each contact force is written as a nonnegative combination of edges of the pyramidal friction cone (Aceituno-Cabezas et al., 2018; Kuindersma et al., 2016). In the case that the contact between a robot's link and the environment is a face rather than a point, such as the sole of a humanoid robot, the resultant wrench (i.e., concatenation of the resultant force and moment) applied to the link can be treated as being generated by the contact forces at the vertices of the face and the set of feasible wrenches subject to the friction constraint form a 6-D convex cone known as the contact wrench cone for the link (Caron et al., 2015a, b; Carpentier et al., 2016; Hirai, 1991; Zheng et al., 2019). In this case, there is an equivalent formulation to limit the 3-D force component and the normal moment component of the resultant wrench by the friction constraint and confine the center of pressure in the contact area of the link (Ponton et al., 2016, 2018). Another way to incorporate the friction constraint is to convert all contact forces to the resultant wrench to counterbalance the gravito-inertia wrench. All such resultant wrenches that can be generated by feasible contact forces also constitute a 6-D cone known as the contact wrench cone for the robot (Aceituno-Cabezas et al., 2017; Dai & Tedrake, 2016; Fernbach et al., 2018; Qiu et al., 2011; Tonneau et al., 2018; Wieber, 2006; Zheng et al., 2019). Linear representations, edge or facet, of the contact wrench cone for a link or the robot could be calculated and used as linear constraints in the TO (Hirai, 1991; Zheng et al., 2019), but the inconvenience is that generally there is no analytical expression of the linear representations for an arbitrary contact area and pre-calculating the constraints may not always be possible in practice. Beyond the friction constraint, some work further takes into account the leg configuration and joint toque limits and formulates the actuation wrench polytope replacing the contact wrench cone (Orsolino et al., 2018). However, its computation is even more complex and can be done only once at the beginning of every TO cycle with a constant leg Jacobian under the quasi-static condition. A more easily-computable motion constraint considering both the robot's static stability and the joint torque limits has also been proposed and added online to the TO (Orsolino et al., 2020).

To reduce the complexity of TO, some methods consider only the linear motion of the CoM (Caron & Kheddar, 2016; Perrin et al., 2015; Zheng et al., 2010) and neglect the angular momentum, which involves a nonlinear non-convex term of the CoM motion. With zero angular momentum, the feasible CoM acceleration set can be derived from the contact wrench cone for any given CoM position (Caron & Kheddar, 2016). Furthermore, some use a simplified dynamics model and require the ZMP to fall in the support polygon (Bellicoso et al., 2017, 2018; Caron et al., 2017; Kalakrishnan et al., 2011; Mastalli et al., 2017, 2020b; Winkler et al., 2015, 2017a, b) or even the static equilibrium criterion that requires just the vertical projection of the CoM to be within the support polygon (Fankhauser et al., 2018). Some methods use a spring-damper contact model such that the contact forces are no longer treated as independent unknown variables (Neunert et al., 2017), or write the contact force as the multiplication of the vector from the contact point to the CoM by a scalar (Perrin et al., 2015). To avoid the non-convexity induced by the centroidal angular momentum, following the decomposition of the angular momentum into the sum of a convex function and a concave function (Herzog et al., 2016), convex relaxations are further proposed to reformulate the angular momentum as convex quadratic inequality constraints for the TO (Ponton et al., 2016, 2018).

In addition to dynamics, some work also considers the leg's kinematics in the TO. Since computing joint angles for desired CoM and foot positions by inverse kinematics involves nonlinear operations, which are not suited for being directly added to the TO, the existing work checks if a foothold is within an allowable region instead. The region can be defined to have a simple geometric shape, such as a rectangle (Winkler et al., 2017b) or a polygon (Bellicoso et al., 2018) centered at a nominal foothold on the ground, and described as a set of linear inequality constraints on the CoM and foot positions. Instead of describing reachable foot

positions with respect to the CoM, a reachable CoM region with respect to each foothold can be described as a polytope and the overall reachable CoM region can be described as the intersection of such polytopes for all feet (Fernbach et al., 2018; Tonneau et al., 2018). Some work defines the reachable foothold to be within a certain distance from the CoM (Ponton et al., 2016), the hip (Fankhauser et al., 2018), or a nominal foot position (Winkler et al., 2017a), which introduces nonlinear constraints on the CoM and foot positions. To consider the yaw of the robot's feet or base, other work defines the region as the intersection of two circles aligned with the foot (Deits & Tedrake, 2014; Kuindersma et al., 2016) or the biggest square inscribed in the leg's workspace aligned with the robot's body (Aceituno-Cabezas et al., 2017) and approximates the trigonometric functions of yaw by piecewise linear segments with binary variables determining which segment to come into play in the TO (Deits & Tedrake, 2014), by which the final approximated reachability constraints can be convex quadratic or linear.

1.1.3 Trajectory optimization

Given step sequences, timings, and footholds, by writing physical feasibility and kinematic reachability as linear constraints, TO can be formulated as QP problems (Dai & Tedrake, 2016; Bellicoso et al., 2017; Fernbach et al., 2018; Kalakrishnan et al., 2011; Qiu et al., 2011; Tonneau et al., 2018; Winkler et al., 2014, 2015), in which the objective functions are often defined to minimize the velocities, accelerations, and/or jerks of the CoM over the trajectory as well as its deviations from a regularized motion and final states or to maximize the contact wrench cone margin. To obtain a natural walking motion, the centroidal angular momentum, which is originally nonlinear and non-convex with the CoM motion, can be minimized by minimizing a convex upper bound of its L_1 norm, which can be further converted to linear and second-order conic constraints in the case of polytopic and ellipsoidal admissible CoM regions, respectively (Dai & Tedrake, 2016). In the latter case, TO can be written as a second-order conic programming (SOCP) problem, versus a QP problem in the former case (Dai & Tedrake, 2016). In general, the QP and SOCP problems can readily be solved by off-the-shelf solvers (Di Gaspero, 1998; Ferreau et al., 2014; GUROBI, 2014; MOSEK, 2014).

By relaxing the nonlinear centroidal momentum dynamics into convex quadratic constraints, TO can be written as a convex quadratically-constrained QP (QCQP) problem (Ponton et al., 2016). Furthermore, allowing variable time discretization introduces some bilinear constraints, which can similarly be approximated by convex quadratic constraints, and the time-optimized trajectory can still be computed as convex QCQP problems (Ponton et al., 2018). In addition, the selection of step sequences and footholds and the yaw motion of the robot's body and feet can be described by linear/quadratic equalities or inequalities with binary variables, and they can be determined together with the CoM trajectory in a single mixed-integer QP (MIQP) or QCQP (MIQCQP) problem (Deits & Tedrake, 2014; Ponton et al., 2016; Aceituno-Cabezas et al., 2017, 2018). Though there are off-the-shelf solvers for convex QCQP, MIQP, and MIQCQP problems (GLPK, 2000; CPLEX, 2010; MOSEK, 2014; GUROBI, 2014), these problems are more complex than QP and start to be less appropriate for online motion generation.

In fact, while many constraints and objective terms are nonlinear without approximation, TO can be written as nonlinear programming (NLP) problems. When time intervals or motion phase durations are taken to be variables like the robot's states at time knots, the time integration constraints are nonlinear equality constraints (Kuindersma et al., 2016; Mastalli et al., 2017, 2020b; Winkler et al., 2018; Orsolino et al., 2018). In the case of variable footholds (Winkler et al., 2017a, b) or CoM height (Bellicoso et al., 2018), the ZMP-based stability condition gives a set of nonlinear inequality constraints. Some work writes the reachability constraint as a nonlinear function of foot positions and the position and orientation of the robot's base, where considering the robot's orientation as variables without any linearization introduces additional nonlinearity (Fankhauser et al., 2018). The objective function can also be nonlinear due to the uses of inverse dynamics to check the feasibility of contact forces and joint torques (Mordatch et al., 2012), a nonlinear expression of contact forces (Perrin et al., 2015), or nonlinear barrier functions to absorb the robot's kinematic, dynamic, and/or contact force constraints (Carpentier et al., 2016; Grandia et al., 2019). Nonlinear formulations can more precisely model the TO problem or generate a wider variety of motions but significantly increase the level of difficulty in solving the resulting NLP problem, which prevents them from online uses. Available NLP solvers include Ipopt (Wächter & Biegler, 2006) and SNOPT (Gill et al., 2005). In addition, TO can be solved as a multi-level optimization problem (Farshidian et al., 2017).

1.2 Our work

After thoroughly reviewing the existing work, we present some different formulations of TO for quadruped robots in this paper. The core part of our work is to compute an optimal CoM trajectory along given or together with step sequences, timings, and footholds, since the whole-body motion can be easily calculated through inverse kinematics once the trajectories of the CoM and steps are determined. Similarly to some of the state-of-the-art methods (Tonneau et al., 2018; Fernbach et al., 2018), we describe the robot's motion based on its centroidal dynamics and write its CoM trajectory as

polynomials with their coefficients to be optimized. The contributions of this paper include

- We discover that the quadratic terms of the polynomial's coefficients yielded by substituting the polynomials into the robot's centroidal dynamics are negligible for a reasonable motion period (e.g., several steps) of a quadruped robot. Also, we choose to determine the robot's orientation heuristically from pre-specified footholds or pre-planned walking paths such that the angular motion generation is decoupled from the linear motion of the CoM. In this way, the CoM trajectory can be computed as a QP problem for given step sequences, timings, and footholds, which can be easily and efficiently solved online. Compared with the existing work optimizing the CoM and angular motions simultaneously as convex QCQP or MIQCQP problems (Deits & Tedrake, 2014; Ponton et al., 2016, 2018) or completely omitting the angular motion (Caron & Kheddar, 2016; Perrin et al., 2015), this proposed approach gives a compromise considering the angular motion while keeping the TO as simple as QP.
- Considering the difficulty in obtaining the accurate position of the robot's CoM, we allow the CoM to be anywhere in a convex polyhedron at a nominal position and incorporate it as linear constraints into the QP problem. By doing this, the generated motion is feasible for the robot wherever its actual CoM is as long as it is inside the polyhedron. This approach to generating robust CoM trajectories is different from the work (Caron & Kheddar, 2016), which restricts the entire CoM trajectory in a pre-specified convex polyhedron and requires the CoM acceleration at any time to be feasible with respect to given footholds for any CoM position in the polyhedron.
- To embed the automatic selection of footholds, we describe candidate footholds for each foot by a set of discrete points on the ground and assign each point a binary variable to indicate if it is selected as the next foothold. Instead of using a contact region, which leads to quadratic constraints and MIQCQP problems (Aceituno-Cabezas et al., 2017, 2018), we write the selection of footholds from discrete points as linear constraints and reformulate the CoM trajectory optimization as an MIQP problem. Furthermore, we divide the motion period into small time intervals and use binary variables to indicate if a foot is in the air or on the ground during every interval. By doing this, we write another MIQP problem to automatically determine the step sequences and timings with the CoM trajectory.
- We conduct a number of experiments to test the proposed method and its extensions together with a vision system and a motion controller on a real quadruped robot (Fig. 1), on which all of them work seamlessly such that the robot can reliably execute generated motions and traverse challenging terrains.

The rest of this paper is organized as follows. Section 2 introduces the basic dynamics of quadruped robots. The proposed locomotion generation method and its extensions are presented in Sect. 3 followed by experimental results in Sect. 4. Section 5 concludes this paper and discusses some future work.

2 Robot dynamics

In this section, we introduce the dynamics based on which the motion controller and generator are derived for our quadruped robot, as shown in Fig. 1. It has been well-known that the whole-body motion of a floating-base robot including quadruped robots can be described as (Wieber, 2006)

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} = \boldsymbol{S}^T\boldsymbol{\tau} + \boldsymbol{J}^T\boldsymbol{f}, \tag{1}$$

where $\boldsymbol{M} \in \mathbb{R}^{N_G \times N_G}$ is the inertia matrix, $\ddot{\boldsymbol{q}} \in \mathbb{R}^{N_G}$ is the generalized acceleration, $\boldsymbol{c} \in \mathbb{R}^{N_G}$ is the sum of Coriolis, centrifugal, and gravity forces, $\boldsymbol{\tau} \in \mathbb{R}^{N_J}$ consists of all joint torques, $\boldsymbol{J} = [\boldsymbol{J}_1^T \ \ \boldsymbol{J}_2^T \ \ \cdots \ \ \boldsymbol{J}_{N_C}^T]^T \in \mathbb{R}^{3N_C \times N_G}$ is the Jacobian of contact points, $\boldsymbol{f} = [\boldsymbol{f}_1^T \ \ \boldsymbol{f}_2^T \cdots \ \ \boldsymbol{f}_{N_C}^T]^T \in \mathbb{R}^{3N_C}$ consists of all contact forces, $N_G = N_J + 6$ is the number of DoFs of the robot including six unactuated DoFs of its floating base, N_J is the number of joints, and N_C is the number of contact points that the robot makes with the environment. Since the robot's base is unactuated, the first six rows of $\boldsymbol{S}^T$ are zero and $\boldsymbol{S} = [\boldsymbol{0}_{N_J \times 6} \ \ \boldsymbol{I}_{N_J \times N_J}] \in \mathbb{R}^{N_J \times N_G}$.

Each contact force $\boldsymbol{f}_i \in \mathbb{R}^3$ should satisfy the friction constraint, which originally defines a circular cone but can be replaced conservatively with a pyramidal cone described by four linear inequalities as

$$\boldsymbol{N}_i^T\boldsymbol{f}_i \le \boldsymbol{0}, \tag{2}$$

where $\boldsymbol{N}_i = -[\mu_i\boldsymbol{n}_i - \boldsymbol{o}_i \ \ \mu_i\boldsymbol{n}_i + \boldsymbol{o}_i \ \ \mu_i\boldsymbol{n}_i - \boldsymbol{t}_i \ \ \mu_i\boldsymbol{n}_i + \boldsymbol{t}_i] \in \mathbb{R}^{3\times 4}$, μ_i is the coefficient of friction, and $\boldsymbol{n}_i, \boldsymbol{o}_i, \boldsymbol{t}_i \in \mathbb{R}^3$ are the unit normal and two orthogonal tangent vectors at contact i described with respect to the global frame, as depicted in Fig. 1. Each inequality in (2) defines a face of the pyramidal cone while each column of $\boldsymbol{N}_i$ is the outward normal of a face. One may also impose a constraint on the contact force magnitude as

$$f_i^L \le \boldsymbol{n}_i^T\boldsymbol{f}_i \le f_i^U, \tag{3}$$

where f_i^L and f_i^U are the nonnegative lower and upper bounds on the normal contact force, respectively.

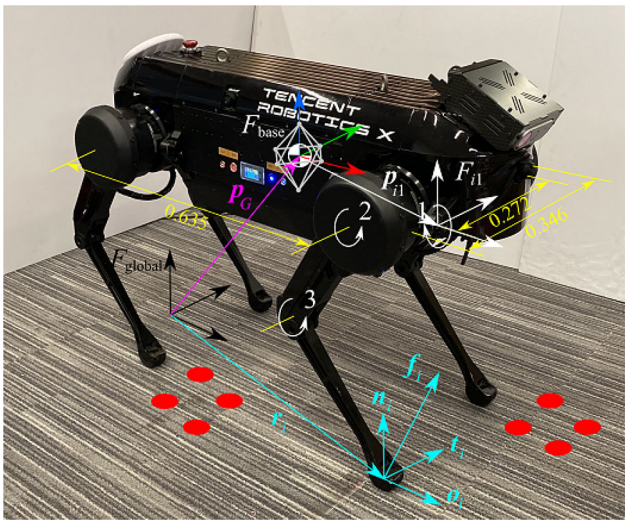


Fig. 1 Our torque-controlled quadruped robot JAMOCA. $\mathcal{F}_{\text{base}}$ is a coordinate frame attached to the robot's CoM and $\boldsymbol{p}_G$ is its position with respect to the global frame $\mathcal{F}_{\text{global}}$. $\boldsymbol{r}_i$ is the contact position of leg i on the ground in frame $\mathcal{F}_{\text{global}}$ and $\boldsymbol{n}_i$, $\boldsymbol{o}_i$, and $\boldsymbol{t}_i$ are normal and tangent vectors at the contact. $\mathcal{F}_{i1}$ is a fixed local frame at the first joint of leg i with a constant offset $\boldsymbol{p}_{i1}$ to frame $\mathcal{F}_{\text{base}}$. The while polyhedron represents a presumptive region of the CoM deviation from its nominal position in the robot model

From the first six rows of (1) we can derive the centroidal dynamics of the robot as (Wieber, 2006)

$$\begin{bmatrix} m(\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) \\ m\hat{\boldsymbol{p}}_G(\ddot{\boldsymbol{p}}_G - \boldsymbol{g}) + \dot{\boldsymbol{\mathcal{L}}} \end{bmatrix} = \sum_{i=1}^{N_C} \begin{bmatrix} \boldsymbol{I}_{3\times3} \\ \hat{\boldsymbol{r}}_i \end{bmatrix} \boldsymbol{f}_i = \boldsymbol{G}\boldsymbol{f}, \tag{4}$$

where m is the total mass of the robot, $\boldsymbol{p}_G \in \mathbb{R}^3$ and $\boldsymbol{r}_i \in \mathbb{R}^3$ are the positions of the robot's CoM and the i-th contact with respect to the global frame, respectively, and $\wedge$ denotes the skew-symmetric matrix to calculate the cross product, $\boldsymbol{\mathcal{L}}$ is the angular momentum of the robot about its CoM, and the matrix $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{I}_{3\times3} & \boldsymbol{I}_{3\times3} & \cdots & \boldsymbol{I}_{3\times3} \\ \hat{\boldsymbol{r}}_1 & \hat{\boldsymbol{r}}_2 & \cdots & \hat{\boldsymbol{r}}_{N_C} \end{bmatrix} \in \mathbb{R}^{6\times3N_C}$. The upper half of (4) is actually Newton's law describing the motion of the robot's CoM, while the lower half is Euler's equation relating the overall angular momentum change of the robot to the moment applied about its CoM.

The angular momentum $\boldsymbol{\mathcal{L}}$ can be written as

$$\boldsymbol{\mathcal{L}} = \sum_{j=0}^{N_L} \left((\boldsymbol{p}_j - \boldsymbol{p}_G) \times m_j \dot{\boldsymbol{p}}_j + \boldsymbol{R}_j \boldsymbol{\mathcal{I}}_j \boldsymbol{\omega}_j \right) \approx \boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0,$$

where $\boldsymbol{p}_j \in \mathbb{R}^3$ and $\boldsymbol{R}_j \in SO(3)$ are the position and orientation of the local coordinate frame of link j of the robot with respect to the global frame, m_j is the mass of link j, $\boldsymbol{\mathcal{I}}_j$ and $\boldsymbol{\omega}_j$ are the inertia tensor and the angular velocity of link j with respect to its local coordinate frame, respectively, and N_L is the number of links. Here, let $j = 0$ represent the base and $j > 0$ the other links of the robot. For a quadruped robot, legs are often made as massless as possible such that $m_j \approx 0$ for $j > 0$ and $\boldsymbol{p}_0 \approx \boldsymbol{p}_G$. Hence, $\boldsymbol{\mathcal{L}}$ is reduced to the angular momentum of the robot's base. Since $\dot{\boldsymbol{R}}_0\boldsymbol{\omega}_0 = \boldsymbol{0}$, we can derive the time derivative of $\boldsymbol{\mathcal{L}}$ as

$$\dot{\boldsymbol{\mathcal{L}}} \approx \boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_0 + \dot{\boldsymbol{R}}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0 = \boldsymbol{\mathcal{I}}_0^S \dot{\boldsymbol{\omega}}_0^S - (\boldsymbol{\mathcal{I}}_0^S \boldsymbol{\omega}_0^S) \times \boldsymbol{\omega}_0^S, \tag{5}$$

where $\boldsymbol{\mathcal{I}}_0^S = \boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{R}_0^T$ is the instantaneous inertia tensor of the robot's base relative to the global frame, and $\boldsymbol{\omega}_0^S = \boldsymbol{R}_0\boldsymbol{\omega}_0$ and $\dot{\boldsymbol{\omega}}_0^S = \boldsymbol{R}_0\dot{\boldsymbol{\omega}}_0$ are the spatial angular velocity and acceleration of the base, respectively.

3 Locomotion generator

The whole control architecture of our quadruped robot is depicted in Fig. 2. Taking a reference motion generated by the motion generator and the current state of the robot acquired by its on-board sensors as inputs, the motion controller determines the required joint torques for the robot to reproduce the reference motion. Since the motion controller implemented on our robot is a traditional one, we briefly explain it in the appendix for the completeness of this paper. In this section, we focus on the locomotion generator based on a new simplification of centroidal dynamics. We first present a basic form aimed at online computing the robot's CoM trajectory for given footholds and timings, while the robot's orientation is determined heuristically from given footholds or walking direction. Further extensions to taking into account the uncertainty of the robot's CoM and the variety of walking patterns in terms of step sequences, timings, and footholds will also be discussed.

3.1 Simplification of the centroidal dynamics

Let $\boldsymbol{p}_{G0} \in \mathbb{R}^3$ be the initial position of the robot's CoM at the beginning of a motion segment and $\boldsymbol{p}_t \in \mathbb{R}^3$ the change of the CoM position from $\boldsymbol{p}_{G0}$ over time t. The value $\boldsymbol{p}_{G0}$ will be updated for but remain constant during every new motion segment to be generated, while $\boldsymbol{p}_t$, starting from zero, describes the motion trajectory of the CoM and needs to be determined. Then, we can write $\boldsymbol{p}_G$ as

$$\boldsymbol{p}_G = \boldsymbol{p}_{G0} + \boldsymbol{p}_t. \tag{6}$$

Substituting (6) into (4) yields

$$\boldsymbol{G}\boldsymbol{f} = \boldsymbol{H}_0\boldsymbol{x}_t - \boldsymbol{w}_c + \boldsymbol{w}_t, \tag{7}$$

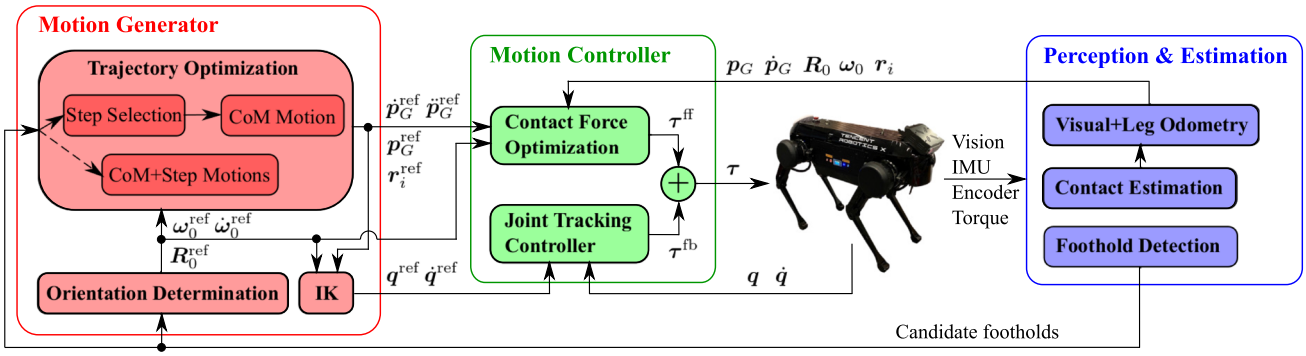


Fig. 2 Control architecture of our quadruped robot. Based on the sensor data including vision, inertia measurement unit (IMU), joint encoder and torque measurement, the position $\boldsymbol{p}_G$ and velocity $\dot{\boldsymbol{p}}_G$ of the robot's CoM and the orientation $\boldsymbol{R}_0$ and angular velocity $\boldsymbol{\omega}_0$ of its floating base as well as its current contacts $\boldsymbol{r}_i$ with the environment and candidate footholds are estimated and detected. The reference angular motion $\boldsymbol{R}_0^{\text{ref}}$ of the robot's base is determined heuristically from candidate footholds, and so can be the reference footholds $\boldsymbol{r}_i^{\text{ref}}$ in the CoM trajectory optimization. Alternatively, the footholds as well as step sequences and timings can be optimized together with the CoM trajectory $\boldsymbol{p}_G^{\text{ref}}$. Finally, reference joint trajectories $\boldsymbol{q}^{\text{ref}}$ are calculated through inverse kinematics with the reference footholds and CoM and orientation trajectories. Based on the robot's centroidal dynamics, the optimal contact forces for realizing the reference motion are calculated and converted to the required joint torque $\boldsymbol{\tau}^{\text{ff}}$, which is added to the joint torque $\boldsymbol{\tau}^{\text{fd}}$ determined by a joint PD tracking controller to form the final control command $\boldsymbol{\tau}$ sent to the robot

where

$$\boldsymbol{H}_0 = \begin{bmatrix} \boldsymbol{0}_{3\times3} & m\boldsymbol{I}_{3\times3} \\ m\hat{\boldsymbol{g}} & m\hat{\boldsymbol{p}}_{G0} \end{bmatrix} \in \mathbb{R}^{6\times6}, \quad \boldsymbol{x}_t = \begin{bmatrix} \boldsymbol{p}_t \\ \ddot{\boldsymbol{p}}_t \end{bmatrix} \in \mathbb{R}^6,$$
$$\boldsymbol{w}_c = \begin{bmatrix} m\boldsymbol{g} \\ m\hat{\boldsymbol{p}}_{G0}\boldsymbol{g} - \dot{\boldsymbol{\mathcal{L}}} \end{bmatrix} \in \mathbb{R}^6, \quad \boldsymbol{w}_t = \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ m\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t \end{bmatrix} \in \mathbb{R}^6.$$

In (7), the first term $\boldsymbol{H}_0\boldsymbol{x}_t$ on the right-hand side of the equal sign is linear to $\boldsymbol{p}_t$ and $\ddot{\boldsymbol{p}}_t$, where the matrix $\boldsymbol{H}_0$ is constant. When the angular motion $\boldsymbol{R}_0$ of the robot over time is predefined (see the next subsection), the second term $\boldsymbol{w}_c$ is time-varying but known.

The last term $\boldsymbol{w}_t$ in (7) is of most interest and key to the validation of the proposed method, so we discuss it in more detail here. First of all, we shall clarify that $\boldsymbol{p}_t$ represents only the change of the CoM position from the initial position $\boldsymbol{p}_{G0}$ and starts from zero for every motion segment to be generated. Thus, for a motion segment where $\hat{\boldsymbol{p}}_t$ is not big, the only nonzero component in $\boldsymbol{w}_t$, namely $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$, is insignificant even if the acceleration of the CoM, namely $\ddot{\boldsymbol{p}}_t$, is large. Moreover, $\boldsymbol{p}_{G0}$ will be updated to the final CoM position in the present motion segment and $\boldsymbol{p}_t$ will restart from zero for generating the next motion segment. Hence, the effect of $\ddot{\boldsymbol{p}}_t$ on the robot's centroidal dynamics will mostly be counted in the term $\boldsymbol{H}_0\boldsymbol{x}_t$ in (7) and $\boldsymbol{p}_t$ will not be accumulated to the next motion segment, which leaves the additional dynamic effect caused by $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$ negligible, especially during a short motion segment.

To further understand the dynamic effect of $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$ in (7), we decompose $\boldsymbol{p}_t = \boldsymbol{p}_t^{xy} + \boldsymbol{p}_t^z$, where $\boldsymbol{p}_t^{xy} = [p_{xt} \;\; p_{yt} \;\; 0]^T$, $\boldsymbol{p}_t^z = [0 \;\; 0 \;\; p_{zt}]^T$, and p_{xt}, p_{yt}, and p_{zt} are the x, y, and z coordinates of $\boldsymbol{p}_t$, respectively. We decompose $\ddot{\boldsymbol{p}}_t$ in the same way to $\ddot{\boldsymbol{p}}_t^{xy}$ and $\ddot{\boldsymbol{p}}_t^z$. Then, $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$ can be expanded as

$$\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t = \hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy} + \hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^z + \hat{\boldsymbol{p}}_t^z\ddot{\boldsymbol{p}}_t^{xy} + \hat{\boldsymbol{p}}_t^z\ddot{\boldsymbol{p}}_t^z. \tag{8}$$

here $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy}$ gives a torque about the z axis, $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^z + \hat{\boldsymbol{p}}_t^z\ddot{\boldsymbol{p}}_t^{xy}$ gives toques about the x and y axes, and $\hat{\boldsymbol{p}}_t^z\ddot{\boldsymbol{p}}_t^z = \boldsymbol{0}$. First, generating the z-direction torque required by $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy}$ needs the help of tangential contact forces. If $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy}$ is in the same direction as the total required torque about z (i.e., the last row of the entire right-hand side of (7)), omitting it in the locomotion generator will leave a risk to the motion controller to face a greater required torque for producing the generated motion, in which additional tangential contact forces may be needed at the risk of bringing the contact forces to the boundary of the friction cone. We presume that the magnitude of $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy}$ is usually not significant and such a risk is very low in real locomotion scenarios. If $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^{xy}$ is opposite to the total required torque, by omitting it we then consider an exaggerated torque in locomotion generation and the generated motion will be easier for tracking control. Moreover, the z-direction motion of the robot during locomotion is often small and steady, which implies that $\ddot{\boldsymbol{p}}_t^z$ and $\boldsymbol{p}_t^z$ are negligible, and so are $\hat{\boldsymbol{p}}_t^{xy}\ddot{\boldsymbol{p}}_t^z$ and $\hat{\boldsymbol{p}}_t^z\ddot{\boldsymbol{p}}_t^{xy}$. Therefore, we exclude the term $\boldsymbol{w}_t$ in (7) from locomotion generation and its real effect will be verified later in experiments (see Sect. 4.2.2).

Remark 1 The method proposed in the work (Fernbach et al., 2018) cleverly expresses the CoM trajectory as a Bezier curve with only one control point such that the quadratic term $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$ automatically disappears without causing any loss in dynamics. Nevertheless, the use of a single free control point may limit the freedom to optimize the CoM trajectory. By contrast, our method allows more free parameters to tune the CoM trajectory but it may not be suited for generating too long motion segments, in which the effect of $\hat{\boldsymbol{p}}_t\ddot{\boldsymbol{p}}_t$ could

increase to a nonnegligible level. By a number of numerical tests as reported in Sect. 4.2.2, we ascertain that $\hat{\boldsymbol{p}}_t \ddot{\boldsymbol{p}}_t$ is indeed insignificant for motion segments in which every leg takes one or two steps or the robot makes one or two walking cycles. We think that generating a long-distance reference motion at once might not be necessary in practice since the robot's perception of terrain can be limited and not able to provide enough information for long-term planning as in our testing scenario (see Sect. 4.1). Also, a robot may have little chance to completely execute a long reference motion due to the deviation accumulated during execution or an environmental change, for which a re-planning is often needed. Hence, we seek an economic and efficient method to generate medium-length motions, with which a robot can quickly plan or re-plan its motion to adapt to the environment or its own state change.

3.2 Determination of the angular motion

It is hard to directly parameterize and optimize the angular motion of a robot like its CoM in the TO. This is due to the fact that the time derivative $\dot{\boldsymbol{\mathcal{L}}}$ of the angular momentum given by (5) involves a cross product and the calculation of the robot's orientation $\boldsymbol{R}_0$ involves trigonometric functions, which induce nonlinear terms of angular parameters in the centroidal dynamics and kinematics constraints. One may use advanced techniques to linearize or relax the nonlinear terms (Deits & Tedrake, 2014; Herzog et al., 2016; Ponton et al., 2016, 2018) but would end up with a more complex optimization problem, such as QCQP or MIQCQP, than QP.

On one hand, simultaneously optimizing the CoM and angular motions of a robot requires a significant computational burden. On the other hand, a robot usually does not require a severe change in its orientation, especially during a short motion segment. As a compromise in this work, we use a heuristics to determine the robot's final orientation based on its footholds on the ground at the end of a motion segment to be generated, say aligning it to the principal axes of the support polygon (Mastalli et al., 2020b), and then interpolate the angular motion between the final orientation and the initial, which is the final orientation in the previous motion segment. In this way, the angular motion can be determined in advance such that $\boldsymbol{R}_0$ and $\dot{\boldsymbol{\mathcal{L}}}$ are known at any time for the TO once the footholds and timings are given. Also, one may take the robot's orientation to follow the walking direction determined by a high-level path planner if there is any. With the orientation being pre-determined, the centroidal dynamics model is actually reduced to a single point mass model to be used in the TO. Compared with some of the existing work (Perrin et al., 2015; Caron & Kheddar, 2016) assuming zero angular momentum, this compromise allows us to consider the angular motion to some extent while keeping the TO problem as a QP problem.

3.3 Representation of the CoM trajectory

We describe each coordinate of $\boldsymbol{p}_t$ by a polynomial such that $\boldsymbol{p}_t$ can be written as

$$\boldsymbol{p}_t = \begin{bmatrix} \boldsymbol{T}_x & \boldsymbol{0}_{1\times(n_y+1)} & \boldsymbol{0}_{1\times(n_z+1)} \\ \boldsymbol{0}_{1\times(n_x+1)} & \boldsymbol{T}_y & \boldsymbol{0}_{1\times(n_z+1)} \\ \boldsymbol{0}_{1\times(n_x+1)} & \boldsymbol{0}_{1\times(n_y+1)} & \boldsymbol{T}_z \end{bmatrix} \begin{bmatrix} \boldsymbol{c}_x \\ \boldsymbol{c}_y \\ \boldsymbol{c}_z \end{bmatrix} = \boldsymbol{T}\boldsymbol{c}, \quad (9)$$

where $\boldsymbol{T}_* = [1 \;\; t \;\; \cdots \;\; t^{n_*}] \in \mathbb{R}^{1\times(n_*+1)}$, $\boldsymbol{c}_* = [c_{*,0} \;\; c_{*,1} \;\; \cdots \;\; c_{*,n_*}] \in \mathbb{R}^{n_*+1}$, n_* is the order of the adopted polynomial, $*$ represents x, y, and z coordinates of $\boldsymbol{p}_t$, and $\boldsymbol{c} = [\boldsymbol{c}_x^T \;\; \boldsymbol{c}_y^T \;\; \boldsymbol{c}_z^T]^T$. Differentiating (9) twice, we derive the acceleration of the CoM as

$$\ddot{\boldsymbol{p}}_t = \ddot{\boldsymbol{T}}\boldsymbol{c}. \quad (10)$$

Substituting (9) and (10) into (7) and omitting $\boldsymbol{w}_t$, we obtain

$$\boldsymbol{G}\boldsymbol{f} = \boldsymbol{H}\boldsymbol{c} - \boldsymbol{w}_c, \quad (11)$$

where $\boldsymbol{H} = \boldsymbol{H}_0[\boldsymbol{T}^T \;\; \ddot{\boldsymbol{T}}^T]^T \in \mathbb{R}^{6\times(n_x+n_y+n_z+3)}$.

Here, we take a specific walking pattern to demonstrate how to determine the value of $\boldsymbol{p}_{G0}$ and the initial and final values of $\boldsymbol{p}_t$, $\dot{\boldsymbol{p}}_t$, and $\ddot{\boldsymbol{p}}_t$, but this is by no means to limit the application scope of the proposed method, which in fact can be applied to different walking patterns. A full cycle of the chosen sample walking pattern comprises the following sequence of motion phases:

- Right-half walking cycle:
  (a) Four-leg support (4 S);
  (b) Hind-right leg swing (HR);
  (c) Front-right leg swing (FR);
  (d) 4 S;
- Left-half walking cycle:
  (e) 4 S;
  (f) Hind-left leg swing (HL);
  (g) Front-left leg swing (FL);
  (h) 4 S.

In short, the robot in turn moves its HR, FR, HL, and FL legs and has a 4 S phase between the right and left half walking cycles. The time duration of each phase is also assigned, as depicted in Fig. 3. Let $\boldsymbol{r}_{\mathrm{FL}}^{\mathrm{str}}$, $\boldsymbol{r}_{\mathrm{FR}}^{\mathrm{str}}$, $\boldsymbol{r}_{\mathrm{HL}}^{\mathrm{str}}$, $\boldsymbol{r}_{\mathrm{HR}}^{\mathrm{str}}$, and $\boldsymbol{r}_{\mathrm{FL}}^{\mathrm{end}}$, $\boldsymbol{r}_{\mathrm{FR}}^{\mathrm{end}}$, $\boldsymbol{r}_{\mathrm{HL}}^{\mathrm{end}}$, $\boldsymbol{r}_{\mathrm{HR}}^{\mathrm{end}}$ be the footholds of the robot at the start and end of a full walking cycle, respectively. Then, the centroids of footholds at the start and end of a right-half, left-half, or full walking cycle can be calculated as

– For the right-half walking cycle:

$$\begin{cases} \boldsymbol{r}_{\text{ctr}}^{\text{str}} = (\boldsymbol{r}_{\text{FL}}^{\text{str}} + \boldsymbol{r}_{\text{FR}}^{\text{str}} + \boldsymbol{r}_{\text{HL}}^{\text{str}} + \boldsymbol{r}_{\text{HR}}^{\text{str}})/4 \\ \boldsymbol{r}_{\text{ctr}}^{\text{end}} = (\boldsymbol{r}_{\text{FL}}^{\text{str}} + \boldsymbol{r}_{\text{FR}}^{\text{end}} + \boldsymbol{r}_{\text{HL}}^{\text{str}} + \boldsymbol{r}_{\text{HR}}^{\text{end}})/4; \end{cases}$$

– For the left-half walking cycle:

$$\begin{cases} \boldsymbol{r}_{\text{ctr}}^{\text{str}} = (\boldsymbol{r}_{\text{FL}}^{\text{str}} + \boldsymbol{r}_{\text{FR}}^{\text{end}} + \boldsymbol{r}_{\text{HL}}^{\text{str}} + \boldsymbol{r}_{\text{HR}}^{\text{end}})/4 \\ \boldsymbol{r}_{\text{ctr}}^{\text{end}} = (\boldsymbol{r}_{\text{FL}}^{\text{end}} + \boldsymbol{r}_{\text{FR}}^{\text{end}} + \boldsymbol{r}_{\text{HL}}^{\text{end}} + \boldsymbol{r}_{\text{HR}}^{\text{end}})/4; \end{cases}$$

– For the full walking cycle:

$$\begin{cases} \boldsymbol{r}_{\text{ctr}}^{\text{str}} = (\boldsymbol{r}_{\text{FL}}^{\text{str}} + \boldsymbol{r}_{\text{FR}}^{\text{str}} + \boldsymbol{r}_{\text{HL}}^{\text{str}} + \boldsymbol{r}_{\text{HR}}^{\text{str}})/4 \\ \boldsymbol{r}_{\text{ctr}}^{\text{end}} = (\boldsymbol{r}_{\text{FL}}^{\text{end}} + \boldsymbol{r}_{\text{FR}}^{\text{end}} + \boldsymbol{r}_{\text{HL}}^{\text{end}} + \boldsymbol{r}_{\text{HR}}^{\text{end}})/4. \end{cases}$$

Whenever we generate the robot's motion for a half or full walking cycle, $\boldsymbol{p}_{G0}$ is taken to be the final position of the CoM in the previous planned motion segment or the initial CoM position if the robot just starts to walk. The initial value of the CoM position change $\boldsymbol{p}_t$ during any motion segment is simply zero and its expected final value, denoted by $\boldsymbol{p}_{\text{end}}^{\text{ref}}$, can be set to the change of centroid of footholds, namely $\boldsymbol{r}_{\text{ctr}}^{\text{end}} - \boldsymbol{r}_{\text{ctr}}^{\text{str}}$. The initial values of $\dot{\boldsymbol{p}}_t$ and $\ddot{\boldsymbol{p}}_t$, denoted by $\dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}$ and $\ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}$, are their final values in the previous motion segment or zero at the beginning of walking. The expected final value $\dot{\boldsymbol{p}}_{\text{end}}^{\text{ref}}$ of $\dot{\boldsymbol{p}}_t$ can be equal to $(\boldsymbol{r}_{\text{ctr}}^{\text{end}} - \boldsymbol{r}_{\text{ctr}}^{\text{str}})/t_{\text{total}}$, where t_{total} is the time duration of the current motion segment, and the final value $\ddot{\boldsymbol{p}}_{\text{end}}^{\text{ref}}$ of $\ddot{\boldsymbol{p}}_t$ can be just zero. In this way, the whole generated CoM trajectory is C^2-continuous.

3.4 Formulation of the locomotion generator

To ensure the feasibility of a generated CoM trajectory, we take K sample points along it, as depicted in Fig. 3, and require the existence of contact forces $\boldsymbol{f}_k$ satisfying (2), (3), and (11) at every point t_k, where $k = 1, 2, \ldots, K$. One may take many sample points and evenly spread them along the motion trajectory. However, the number K directly affects the number of variables and constraints as well as the complexity of the final QP problem. To keep it as simple as possible, we set the sample points wherever mostly needed as

(1) Two sample points at the conjunction of two leg swing phases with each belonging to one phase, like the two between the HR and FR phases in Fig. 3. Since the contacts in the two phases are different and do not contain each other, the motion feasibility should be verified separately at the conjunction.
(2) One sample point at the conjunction with a 4 S phase and associate it with the other phase, like the one between the 4 S and HR phases in Fig. 3. If a motion is feasible from the other phase (no matter whether it is a leg swing or 4 S phase), it must be feasible at the conjunction for the 4 S phase with extra or same legs touching the ground.

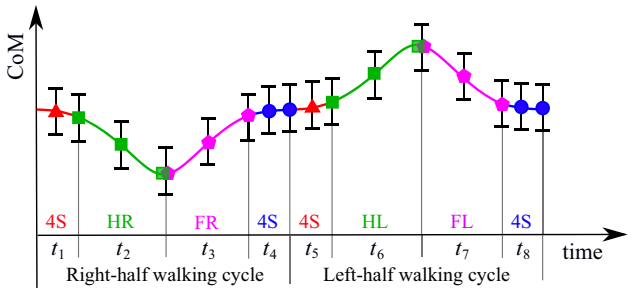


Fig. 3 Illustration of the CoM trajectory and sample points for checking the existence of feasible contact forces. To have a robust trajectory to the uncertainty of the actual CoM position, we further require the trajectory to be feasible for the CoM lying anywhere in a predefined region (depicted by black segments here or the white polyhedron in Fig. 1) surrounding the nominal CoM position (depicted by colorful markers)

(3) Uniform sample points during every motion phase. We set one point in the middle of every phase in Fig. 3 but can use more for a long motion phase or a phase with fewer supporting legs since such a motion phase is more likely to be infeasible.

Combining the above formulations, we finally write the computing of the CoM trajectory as the following QP problem, in which $\boldsymbol{c}$ and all $\boldsymbol{f}_k$'s are unknowns:

$$\begin{cases} \min\limits_{\boldsymbol{c}, \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_K} & J_{\text{grf}} + J_{\text{len}} + J_{\text{tgt}} \\ \text{subject to} & \boldsymbol{G}_k \boldsymbol{f}_k = \boldsymbol{H}_k \boldsymbol{c} - \boldsymbol{w}_{ck} \\ & \boldsymbol{N}_{ik}^T \boldsymbol{f}_{ik} \le \boldsymbol{0} \\ & f_i^L \le \boldsymbol{n}_{ik}^T \boldsymbol{f}_{ik} \le f_i^U \\ & i = 1, 2, \ldots, N_C \\ & k = 1, 2, \ldots, K \\ & \boldsymbol{T}_0 \boldsymbol{c} = \boldsymbol{0}, \ \dot{\boldsymbol{T}}_0 \boldsymbol{c} = \dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}, \ \ddot{\boldsymbol{T}}_0 \boldsymbol{c} = \ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}. \end{cases} \tag{12}$$

It should be noted that the matrix $\boldsymbol{H}$ and the vector $\boldsymbol{w}_c$ in (11) vary continuously along with the time. Hence, we denote them at time t_k by $\boldsymbol{H}_k$ and $\boldsymbol{w}_{ck}$ in (12). Depending on the contact status and locations of the robot at time t_k, however, the contact normal $\boldsymbol{n}_i$, matrices $\boldsymbol{G}$ and $\boldsymbol{N}_i$, and the number of contacts N_C do not change all the time, though we still distinguish them at different t_k's with subscript k. The last line in (12) forces the planned motion segment to continue with the previous one, where $\dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}$ and $\ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}$ are the CoM velocity and acceleration at the end of the previous motion segment.

Here, the objective function of (12) consists of three terms. The objective term J_{grf} is intended to penalize high contact forces and defined as the weighted sum of $\boldsymbol{f}_k^T \boldsymbol{f}_k$ for all k,

$$J_{\text{grf}} = \sum_{k=1}^{K} \boldsymbol{f}_k^T \boldsymbol{W}_{fk} \boldsymbol{f}_k, \tag{13}$$

where $\boldsymbol{W}_{fk} \in \mathbb{R}^{3N_C \times 3N_C}$ is a weight matrix. To shorten the generated trajectory and restrain its oscillation, we define the term J_{len} in (12) as

$$J_{\text{len}} = \sum_{k'=1}^{K'} \Delta \boldsymbol{p}_{k'}^T \boldsymbol{W}_{k'} \Delta \boldsymbol{p}_{k'}, \tag{14}$$

where $\Delta \boldsymbol{p}_{k'} = \boldsymbol{T}(t_{k'})\boldsymbol{c} - \boldsymbol{T}(t_{k'-1})\boldsymbol{c}$, $\boldsymbol{W}_{k'} \in \mathbb{R}^{3\times 3}$ is a weight matrix, and $t_{k'}, k' = 1, \ldots, K'$ are sample times over the trajectory, which may not necessarily be the same as the sample times t_k's for checking the motion feasibility. Here, one may choose to penalize $\dot{\boldsymbol{p}}_{k'} = \dot{\boldsymbol{T}}(t_{k'})\boldsymbol{c}$ instead of $\Delta \boldsymbol{p}_{k'}$ to have a similar effect. Lastly, J_{tgt} is defined as

$$J_{\text{tgt}} = \Delta \boldsymbol{p}^T \boldsymbol{W}_p \Delta \boldsymbol{p} + \Delta \dot{\boldsymbol{p}}^T \boldsymbol{W}_v \Delta \dot{\boldsymbol{p}} + \Delta \ddot{\boldsymbol{p}}^T \boldsymbol{W}_a \Delta \ddot{\boldsymbol{p}}, \tag{15}$$

where $\Delta \boldsymbol{p} = \boldsymbol{p}_{\text{end}}^{\text{ref}} - \boldsymbol{T}(t_{\text{total}})\boldsymbol{c}$, $\Delta \dot{\boldsymbol{p}} = \dot{\boldsymbol{p}}_{\text{end}}^{\text{ref}} - \dot{\boldsymbol{T}}(t_{\text{total}})\boldsymbol{c}$, $\Delta \ddot{\boldsymbol{p}} = \ddot{\boldsymbol{p}}_{\text{end}}^{\text{ref}} - \ddot{\boldsymbol{T}}(t_{\text{total}})\boldsymbol{c}$, $\boldsymbol{W}_p, \boldsymbol{W}_v, \boldsymbol{W}_a \in \mathbb{R}^{3\times 3}$ are weight matrices, and $\boldsymbol{p}_{\text{end}}^{\text{ref}}$, $\dot{\boldsymbol{p}}_{\text{end}}^{\text{ref}}$, and $\ddot{\boldsymbol{p}}_{\text{end}}^{\text{ref}}$ are the reference values of $\boldsymbol{p}_t$, $\dot{\boldsymbol{p}}_t$, and $\ddot{\boldsymbol{p}}_t$ at the end of the generated trajectory, respectively, as discussed in the previous subsection. Besides the three terms, other quadratic terms can be added to the objective function if needed to reflect additional demands on the generated trajectory.

In the above QP formulation, we require the initial CoM position, velocity, and acceleration of the current motion segment to be exactly equal to the final of the previous segment by the last line in (12). Nevertheless, the final CoM state of the current segment is not restricted but made close to the reference value through the objective term J_{tgt} defined by (15). There are many variant formulations. For example, one may add the requirement of reaching the reference final state as linear equality constraints to the QP problem instead of as an objective term. Also, the initial and/or final accelerations could be completely free. We test several variant formulations in Sect. 4.2.2.

3.5 Extensions

Equation (12) provides a basic form of the locomotion generator. Hereinafter, we discuss some extensions.

3.5.1 Considering the CoM uncertainty

Practically, the robot's CoM is not invariant with respect to its body frame during locomotion since the legs are not truly massless and their movements change the CoM position of the robot. Moreover, because of the inevitable modeling error, there is a difference between the CoM computed from the robot's model and the actual one. To consider the uncertainty of the CoM in locomotion generation and enhance the robustness of the generated motion, we take $\boldsymbol{p}_{G0}$ to be the nominal initial CoM position and assume that the actual initial CoM can be anywhere inside a convex polyhedron with vertices $\boldsymbol{v}_h$, $h = 1, 2, \ldots, N_V$ centered at $\boldsymbol{p}_{G0}$, as depicted in Fig. 1. Then, we hope to have a trajectory, still determined by the coefficient $\boldsymbol{c}$, that is feasible for any point inside the polyhedron as the actual CoM.

To do this, we first prove that a trajectory given by $\boldsymbol{c}$ is feasible for any CoM position inside the polyhedron if it is feasible for all the vertices $\boldsymbol{v}_h$'s. Substituting $\boldsymbol{v}_h$ for $\boldsymbol{p}_{G0}$ in (11) and denoting by $\boldsymbol{f}_h$ the corresponding contact forces in this case, we obtain

$$\boldsymbol{G}\boldsymbol{f}_h = \boldsymbol{H}_h\boldsymbol{c} - \boldsymbol{w}_{ch}, \tag{16}$$

where $\boldsymbol{H}_h$ and $\boldsymbol{w}_{ch}$ are the corresponding matrix and vector to $\boldsymbol{v}_h$. For any position $\boldsymbol{p}$ in the convex polyhedron, we can write it as a convex combination of $\boldsymbol{v}_h$, $h = 1, 2, \ldots, N_V$, i.e.,

$$\boldsymbol{p} = \sum_{h=1}^{N_V} \lambda_h \boldsymbol{v}_h, \tag{17}$$

where $\lambda_h \geq 0$ for all h and $\sum_{h=1}^{N_V} \lambda_h = 1$. Substituting $\boldsymbol{p}$ for $\boldsymbol{p}_{G0}$ in (11), from (16) and (17) we can derive

$$\boldsymbol{H}_p\boldsymbol{c} - \boldsymbol{w}_{cp} = \sum_{h=1}^{N_V} \lambda_h(\boldsymbol{H}_h\boldsymbol{c} - \boldsymbol{w}_{ch}) = \boldsymbol{G}\sum_{h=1}^{N_V} \lambda_h \boldsymbol{f}_h, \tag{18}$$

where $\boldsymbol{H}_p$ and $\boldsymbol{w}_{cp}$ are the corresponding matrix and vector to $\boldsymbol{p}$. If we have a solution for $\boldsymbol{c}$ such that (16) holds for all $\boldsymbol{v}_h$'s with all $\boldsymbol{f}_h$'s satisfying the force constraints (2) and (3), from (18) it follows that the trajectory also works for $\boldsymbol{p}$ since $\sum_{h=1}^{N_V} \lambda_h \boldsymbol{f}_h$ as a convex combination of $\boldsymbol{f}_h$, $h = 1, 2, \ldots, N_V$ is the required contact force for the robot with its CoM at $\boldsymbol{p}$ to fulfill the generated motion and satisfies the force constraints. In other words, a motion trajectory feasible for the CoM at any of the vertices $\boldsymbol{v}_h, h = 1, 2, \ldots, N_V$ will be feasible for any CoM position inside the convex polyhedron.

Therefore, combining (16) with (12), we derive another QP formulation to compute such a robust CoM trajectory as

$$\begin{cases} \min\limits_{\boldsymbol{c}, \boldsymbol{f}_{hk}\text{'s}} & J_{\text{grf}} + J_{\text{len}} + J_{\text{tgt}} \\ \text{subject to} & \boldsymbol{G}_k\boldsymbol{f}_{hk} = \boldsymbol{H}_{hk}\boldsymbol{c} - \boldsymbol{w}_{chk} \\ & \boldsymbol{N}_{ik}^T\boldsymbol{f}_{hik} \leq \boldsymbol{0} \\ & f_i^L \leq \boldsymbol{n}_{ik}^T\boldsymbol{f}_{hik} \leq f_i^U \\ & h = 1, 2, \ldots, N_V \\ & i = 1, 2, \ldots, N_C \\ & k = 1, 2, \ldots, K \\ & \boldsymbol{T}_0\boldsymbol{c} = \boldsymbol{0},\ \dot{\boldsymbol{T}}_0\boldsymbol{c} = \dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}},\ \ddot{\boldsymbol{T}}_0\boldsymbol{c} = \ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}. \end{cases} \tag{19}$$

Remark 2 To have a robust CoM trajectory, Caron and Kheddar (2016) restricted the CoM trajectory within a convex polyhedral tube and required the CoM acceleration along the trajectory within the intersection of feasible CoM acceleration sets for the vertices of the tube. Since these vertices are pre-arranged and fixed prior to the generation of the CoM trajectory, the number of constraints can be smaller than that in our case, where the polyhedron around the nominal CoM moves along the generated trajectory, as depicted in Fig. 3, and we would like the trajectory to be feasible no matter where the CoM deviates to from the nominal position as long as it is still inside the polyhedron. To do so, we have to ensure the existence of feasible contact forces at every sample point for each case that the CoM is at a vertex of the polyhedron. It is noted that such a straightforward formulation (19) induces N_V times as many variables and constraints. One possible way to simplify the problem is to pre-compute the contact wrench cone for given footholds (Hirai, 1991; Zheng et al., 2019) and rewrite all the equilibrium and friction constraints as another set of linearly inequality constraints on every $\boldsymbol{H}_{hk}\boldsymbol{c} - \boldsymbol{w}_{chk}$, while all the contact force variables can be eliminated. Furthermore, we may enclose $\boldsymbol{H}_{hk}\boldsymbol{c} - \boldsymbol{w}_{chk}$ for $h = 1, 2, \ldots, N_V$ at every time k with a ball of estimated radius δ centered at $\boldsymbol{H}_k\boldsymbol{c} - \boldsymbol{w}_{ck}$ computed using the nominal CoM position. Then, instead of $\boldsymbol{H}_{hk}\boldsymbol{c} - \boldsymbol{w}_{chk}$'s, we just require the whole ball to be inside the contact wrench cone, which is equivalent to applying those linearly inequality constraints to only $\boldsymbol{H}_k\boldsymbol{c} - \boldsymbol{w}_{ck}$ with δ as a safety margin. In this way, the curse of N_V will be completely eliminated. In the future, we would explore more efficient ways to cover this uncertainty as well as others, such as the errors in contact force tracking, contact normals, and friction coefficients (Del Prete et al., 2016).

3.5.2 Considering the locomotion variety

In the above discussion, the walking pattern including step sequences, timings, and footholds is prespecified. Here, we extend our method to the case where all of these are variable to increase the variety of locomotion.

To facilitate the understanding of the following derivations, we first consider the case where only footholds need to be determined along with the CoM trajectory and then include the variability of step sequences and timings. Assume that there are $N_i \geq 1$ candidate footholds for foot i, denoted by $\boldsymbol{r}_{ij}$, $j = 1, 2, \ldots, N_i$, which together with the normal $\boldsymbol{n}_{ij}$ and tangent vectors $\boldsymbol{o}_{ij}$ and $\boldsymbol{t}_{ij}$ therein can be acquired by the vision system of our robot. We write the selected foothold $\boldsymbol{r}_i$ as

$$\boldsymbol{r}_i = \sum_{j=1}^{N_i} \beta_{ij}\boldsymbol{r}_{ij}, \tag{20}$$

where $\beta_{ij} \in \{0, 1\}$ and $\sum_{j=1}^{N_i} \beta_{ij} = 1$. By using the binary variables β_{ij}'s and the constraint $\sum_{j=1}^{N_i} \beta_{ij} = 1$, only one of binary variables β_{ij}, $j = 1, 2, \ldots, N_i$ is 1, which implies that the only corresponding candidate foothold is selected. Also, let $\boldsymbol{f}_{ij}$ be the contact force at $\boldsymbol{r}_{ij}$. Similar to (2), the friction constraint on $\boldsymbol{f}_{ij}$ can be written as

$$\boldsymbol{N}_{ij}^T\boldsymbol{f}_{ij} \leq \boldsymbol{0}, \tag{21}$$

where $\boldsymbol{N}_{ij} = -[\mu_{ij}\boldsymbol{n}_{ij} - \boldsymbol{o}_{ij} \quad \mu_{ij}\boldsymbol{n}_{ij} + \boldsymbol{o}_{ij} \quad \mu_{ij}\boldsymbol{n}_{ij} - \boldsymbol{t}_{ij} \quad \mu_{ij}\boldsymbol{n}_{ij} + \boldsymbol{t}_{ij}] \in \mathbb{R}^{3\times 4}$. Furthermore, we limit the normal component of $\boldsymbol{f}_{ij}$ as

$$\boldsymbol{n}_{ij}^T\boldsymbol{f}_{ij} \leq f_i^U\beta_{ij}. \tag{22}$$

From (21) and (22) we have $\boldsymbol{f}_{ij} = \boldsymbol{0}$ if $\beta_{ij} = 0$. Hence, only the contact force $\boldsymbol{f}_{ij}$ at the selected foothold can be nonzero and (7) can be rewritten as

$$\sum_{i=1}^{N_C}\sum_{j=1}^{N_i} \boldsymbol{G}_{ij}\boldsymbol{f}_{ij} = \boldsymbol{H}\boldsymbol{c} - \boldsymbol{w}_c, \tag{23}$$

where $\boldsymbol{G}_{ij} = [\boldsymbol{I}_{3\times 3} \quad \hat{\boldsymbol{r}}_{ij}^T]^T \in \mathbb{R}^{6\times 3}$.

While selecting appropriate footholds, we shall make sure that every selected foothold is reachable by the robot. To do this, instead of using an oversimplified region (Winkler et al., 2017b; Bellicoso et al., 2018) or nonlinear functions (Ponton et al., 2016; Fankhauser et al., 2018; Winkler et al., 2017a) to describe the kinematic constraint, we approximate the workspace of each robot's leg as a convex polyhedron, as depicted in Fig. 4. After representing the workspace with discrete points and cutting off the upper non-convex part, we compute the convex polyhedron by the expanding polytope algorithm (Zheng & Yamane, 2015) with the points in the lower part, which is the most useful portion of the leg's workspace for locomotion. The polyhedron is expressed in a fixed local coordinate frame $\mathcal{F}_{i1}$ at the first joint of the leg (see Fig. 1) as a set of linear inequalities with each inequality representing a face of the polyhedron,

$$\boldsymbol{S}_i^T\boldsymbol{x}_i \leq \boldsymbol{d}_i, \tag{24}$$

where $\boldsymbol{x}_i \in \mathbb{R}^3$ is the foot position in frame $\mathcal{F}_{i1}$, $\boldsymbol{S}_i = [\boldsymbol{s}_1 \quad \boldsymbol{s}_2 \quad \cdots \quad \boldsymbol{s}_L] \in \mathbb{R}^{3\times L}$, $\boldsymbol{d}_i = [d_1 \quad d_2 \quad \cdots \quad d_L]^T \in \mathbb{R}^L$, $\boldsymbol{s}_l$ and d_l for $l = 1, 2, \ldots, L$ are the unit normal of a face and its distance from the origin of frame $\mathcal{F}_{i1}$, respectively, and L is the number of faces. For legs with the same configuration, we can define frames $\mathcal{F}_{i1}$ in the same way such that $\boldsymbol{S}_i$ and $\boldsymbol{d}_i$ are identical for them and need be calculated only once. To check if a current or desired foot location $\boldsymbol{r}_i$ in the global frame is inside the workspace by (24), we convert $\boldsymbol{r}_i$ into frame $\mathcal{F}_{i1}$ as

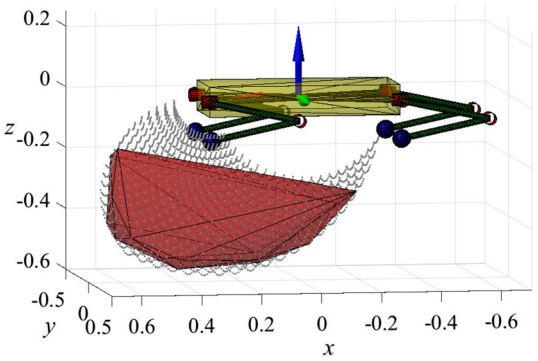


Fig. 4 Convex approximation of the leg's workspace of our quadruped robot

$$\boldsymbol{x}_i = \boldsymbol{R}_{i1}^T \left(\boldsymbol{R}_0^T (\boldsymbol{r}_i - \boldsymbol{p}_G) - \boldsymbol{p}_{i1} \right), \tag{25}$$

where $\boldsymbol{p}_{i1} \in \mathbb{R}^3$ and $\boldsymbol{R}_{i1} \in SO(3)$ are the position and orientation of frame $\mathcal{F}_{i1}$ with respect to frame $\mathcal{F}_{\text{base}}$ and are both constant. Combining (6), (9), (24), and (25), we obtain a set of linear inequality constraints on the CoM position and footholds in the global frame as

$$\boldsymbol{A}_i \boldsymbol{c} + \boldsymbol{B}_i \boldsymbol{r}_i \leq \boldsymbol{b}_i, \tag{26}$$

where $\boldsymbol{A}_i = -\boldsymbol{S}_i^T \boldsymbol{R}_{i1}^T \boldsymbol{R}_0^T \boldsymbol{T}$, $\boldsymbol{B}_i = \boldsymbol{S}_i^T \boldsymbol{R}_{i1}^T \boldsymbol{R}_0^T$, and $\boldsymbol{b}_i = \boldsymbol{d}_i + \boldsymbol{S}_i^T \boldsymbol{R}_{i1}^T \boldsymbol{p}_{i1} + \boldsymbol{S}_i^T \boldsymbol{R}_{i1}^T \boldsymbol{R}_0^T \boldsymbol{p}_{G0}$. Substituting (20) into (26), we finally obtain the reachability constraint on the selected footholds as

$$\boldsymbol{A}_i \boldsymbol{c} + \sum_{j=1}^{N_i} \beta_{ij} \boldsymbol{B}_i \boldsymbol{r}_{ij} \leq \boldsymbol{b}_i. \tag{27}$$

Combining (21)–(23) and (27), therefore, we can simply rewrite (19) as the following MIQP problem to take into account the selection of footholds:

$$\begin{cases} \min\limits_{\boldsymbol{c}, \boldsymbol{f}_{ijk}\text{'s}, \beta_{ij}\text{'s}} & J_{\text{grf}} + J_{\text{len}} + J_{\text{tgt}} \\ \text{subject to} & \sum_{i=1}^{N_C} \sum_{j=1}^{N_i} \boldsymbol{G}_{ijk} \boldsymbol{f}_{ijk} = \boldsymbol{H}_k \boldsymbol{c} - \boldsymbol{w}_{ck} \\ & \boldsymbol{N}_{ijk}^T \boldsymbol{f}_{ijk} \leq \boldsymbol{0} \\ & f_i^L \beta_{ij} \leq \boldsymbol{n}_{ijk}^T \boldsymbol{f}_{ijk} \leq f_i^U \beta_{ij} \\ & \boldsymbol{A}_{ik} \boldsymbol{c} + \boldsymbol{B}_{ik} \sum_{j=1}^{N_i} \beta_{ij} \boldsymbol{r}_{ij} \leq \boldsymbol{b}_{ik} \\ & \sum_{j=1}^{N_i} \beta_{ij} = 1, \quad \beta_{ij} \in \{0, 1\} \\ & i = 1, 2, \ldots, N_C \\ & j = 1, 2, \ldots, N_i \\ & k = 1, 2, \ldots, K \\ & \boldsymbol{T}_0 \boldsymbol{c} = \boldsymbol{0}, \ \dot{\boldsymbol{T}}_0 \boldsymbol{c} = \dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}, \ \ddot{\boldsymbol{T}}_0 \boldsymbol{c} = \ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}. \end{cases} \tag{28}$$

Remark 3 One may describe candidate footholds with a region instead of discrete points and write it as a set of linear inequalities if the region is planar and convex (Deits & Tedrake, 2014) or even a nonlinear continuous function

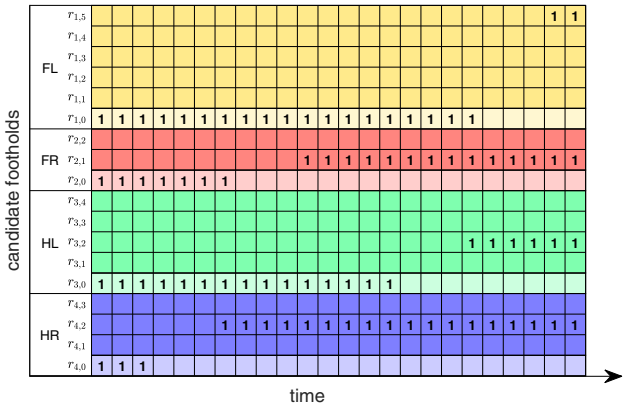


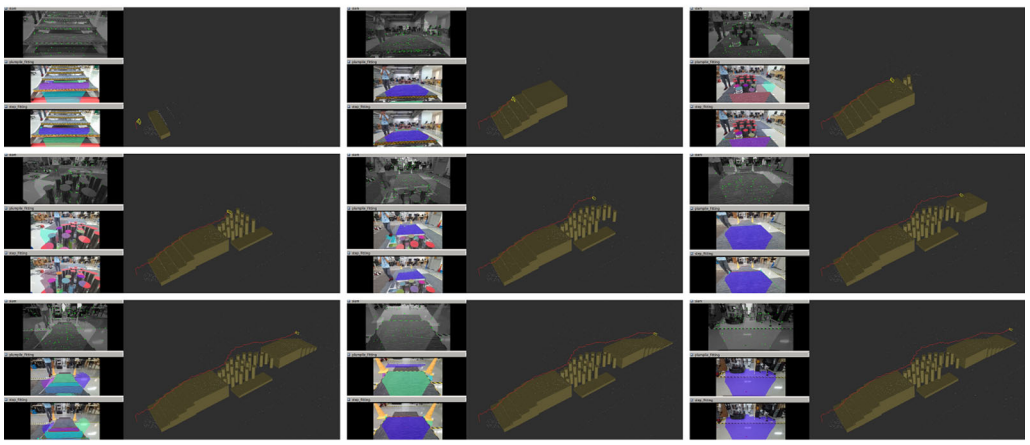
Fig. 5 Representation of foothold occupation during time intervals used in the MIQP problem (29). Each row corresponds to a foothold and each cell is assigned a binary variable with value "1" indicating that the foothold is occupied for the time interval corresponding to the cell. The light and dark colors represent the current and candidate footholds, respectively

if the terrain surface is curved. In this case, however, the contact position $\boldsymbol{r}_i$ and associated contact force $\boldsymbol{f}_i$ are both unknown variables, which causes $\boldsymbol{G}\boldsymbol{f}$ in (7), involving the cross product of $\boldsymbol{r}_i$ and $\boldsymbol{f}_i$, to be quadratic and the final formulation to be a non-convex QCQP or even NLP problem. By contrast, the proposed discrete-point description of candidate footholds enables us to rewrite (7) still in a linear form as (23), where $\boldsymbol{G}_{ij}$'s are all known for given candidate footholds and $\boldsymbol{f}_{ij}$ can be turned on or off by the corresponding binary variable β_{ij} in terms of (21) and (22). Moreover, the MIQP formulation (28) can be applied to any terrain without any change no matter whether the terrain is even, uneven, or rough. Such an MIQP problem (28) can be easily solved by off-the-shelf solvers (GLPK, 2000; CPLEX, 2010; MOSEK, 2014; GUROBI, 2014).

To further allow variable step sequences and timings, we divide the time line of the desired motion segment into small intervals, as depicted in Fig. 5, which together with the candidate footholds on the vertical axis form a 2-D mesh. Each cell of the mesh is associated with a binary variable β_{ijk} and $\beta_{ijk} = 1$ means that foot i is on the ground at position $\boldsymbol{r}_{ij}$ during the k-th time interval. Here, j and k both start with 0 and $\boldsymbol{r}_{i0}$ is the initial contact location for foot i. The binary variables need to satisfy some conditions:

(a) $\beta_{ijk} = 1$ for $\forall i$ while $j = k = 0$: This means that foot i is at $\boldsymbol{r}_{i0}$ when the motion segment begins.
(b) $\beta_{ijk} = 0$ for $\forall i$ while $j = 0$ and $k = K$: The purpose of this condition is to force foot i to leave $\boldsymbol{r}_{i0}$ and take a step in the motion segment.
(c) $\beta_{ijk} \geq \beta_{ijk+1}$ for $\forall i, k$ while $j = 0$: By this condition, once β_{ijk} turns to 0 from 1, it will remain 0 for the rest

(a)



(b)

Fig. 6 Test of the proposed method on our quadruped robot. **a** Testing scenario consisting of platforms, staircases, and uneven piles. **b** Perceived and reconstructed terrain during robot walking

of the motion, which means that foot i will not step back to its initial foothold once leaving.

(d) $\beta_{ijk} \leq \beta_{ijk+1}$ for $\forall i, k$ while $j > 0$: After foot i reaches a new location, it will remain there for the rest of the motion segment.

(e) $\sum_{j=0}^{N_i} \beta_{ijk} \leq 1$ for $\forall i, k$: This reflects that foot i can exist at only one location or be in the air at a time.

(f) $K - \sum_{j=0}^{N_i} \sum_{k=0}^{K} \beta_{ijk} \geq K_{Si}$ for $\forall i, k$: With condition e), $\sum_{j=0}^{N_i} \sum_{k=0}^{K} \beta_{ijk}$ is the total number of time intervals for which foot i is on the ground. Then, $K - \sum_{j=0}^{N_i} \sum_{k=0}^{K} \beta_{ijk}$ is the swing duration of foot i, which should have a lower bound K_{Si} such that foot i has enough time to touch down.

Incorporating all these conditions and the binary variables, we can expand (28) as

$$\begin{cases} \min\limits_{\boldsymbol{c}, \boldsymbol{f}_{ijk}\text{'s}, \beta_{ijk}\text{'s}} & J_{\text{grf}} + J_{\text{len}} + J_{\text{tgt}} \\ \text{subject to} & \sum_{i=1}^{N_C} \sum_{j=1}^{N_i} \boldsymbol{G}_{ijk} \boldsymbol{f}_{ijk} = \boldsymbol{H}_k \boldsymbol{c} - \boldsymbol{w}_{ck} \\ & \boldsymbol{N}_{ijk}^T \boldsymbol{f}_{ijk} \leq \boldsymbol{0} \\ & f_i^L \beta_{ijk} \leq \boldsymbol{n}_{ijk}^T \boldsymbol{f}_{ijk} \leq f_i^U \beta_{ijk} \\ & \boldsymbol{A}_{ik} \boldsymbol{c} + \boldsymbol{B}_{ik} \sum_{j=1}^{N_i} \beta_{ijk} \boldsymbol{r}_{ij} \leq \boldsymbol{b}_{ik} \\ & \text{Constraints a)–f)}, \quad \beta_{ijk} \in \{0, 1\} \\ & i = 1, 2, \ldots, N_C \\ & j = 0, 1, \ldots, N_i \\ & k = 0, 1, \ldots, K \\ & \boldsymbol{T}_0 \boldsymbol{c} = \boldsymbol{0}, \ \dot{\boldsymbol{T}}_0 \boldsymbol{c} = \dot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}, \ \ddot{\boldsymbol{T}}_0 \boldsymbol{c} = \ddot{\boldsymbol{p}}_{\text{str}}^{\text{ref}}. \end{cases} \tag{29}$$

Till now, the timings of lifting up and touching down of each foot are determined by the binary variables β_{ijk}'s, and so is the step sequence.

4 Experimental results

We have implemented the motion controller and generator in C++ and conducted hardware and numerical experiments to verify the effectiveness and performance of the proposed motion generator in quadruped locomotion. The experimental results are reported as follows.

4.1 Experiment setup

Figure 1 shows the quadruped robot named JAMOCA used in the experiment. The robot is based on a Jueying Pro robot from DeepRobotics with its original vision system replaced by an Azure Kinect camera. The robot is about 70 kg and equipped with a PC having an Intel Core i7–5650U CPU and 8GB RAM, on which the motion generator and controller run in parallel. The perception system runs on a separate onboard computer, which processes and communicates the acquired terrain information with the above PC. We implemented the whole control system including perception, state estimation, and motion generation and control, as depicted in Fig. 2, in which the only open-source libraries used include Eigen3 for linear algebra computation, qpOASES for solving QP problems (Ferreau et al., 2014), and GUROBI for solving MIQP problems (GUROBI, 2014). While solving QP or MIQP problems, we did not implement linear equality constraints but solved them first to attain independent variables instead to reduce the actual numbers of variables and constraints in optimization.

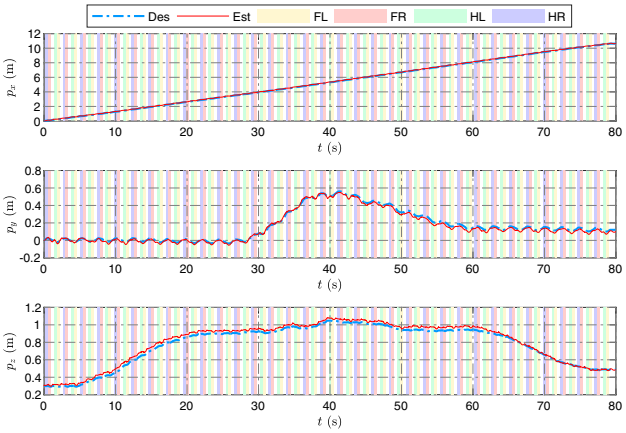


Fig. 7 CoM trajectories generated by the proposed method and realized by the real robot on the terrain shown in Fig. 6a

Figure 6a shows the testing terrain consisting of platforms, staircases, and uneven piles that the robot is required to traverse. The inclination angle of the whole staircases is about 20° and the height of every stair is about 12 cm. The piles have a circular upper face with a radius of 10 cm and are randomly placed with an irregular spacing between 20 and 50 cm. The maximum difference in height between neighboring piles is 16 cm. With the implemented perception system, our robot has the capability of recognizing the upper surfaces of the platforms, stairs, and piles for placing its feet while it is walking over the terrain, as shown in Fig. 6b.

4.2 Experiments with a predefined walking pattern and foothold selection strategy

Here we let the robot take the walking pattern as described in Sect. 3.3 to traverse the terrain, where the duration of every 4 S phase is set to 0.2 s and that of every leg swing phase to 0.4 s. Then, the duration of every half/hull walking cycle is 1.2/2.4 s. We use the proposed method with three sextic polynomials to generate a 3-D CoM trajectory online for every half walking cycle next to the currently executed one. Based on the acquired terrain information, simple foothold selection strategies are used in this experiment, while the optimization of foothold selection is tested in Sect. 4.4. On the platforms

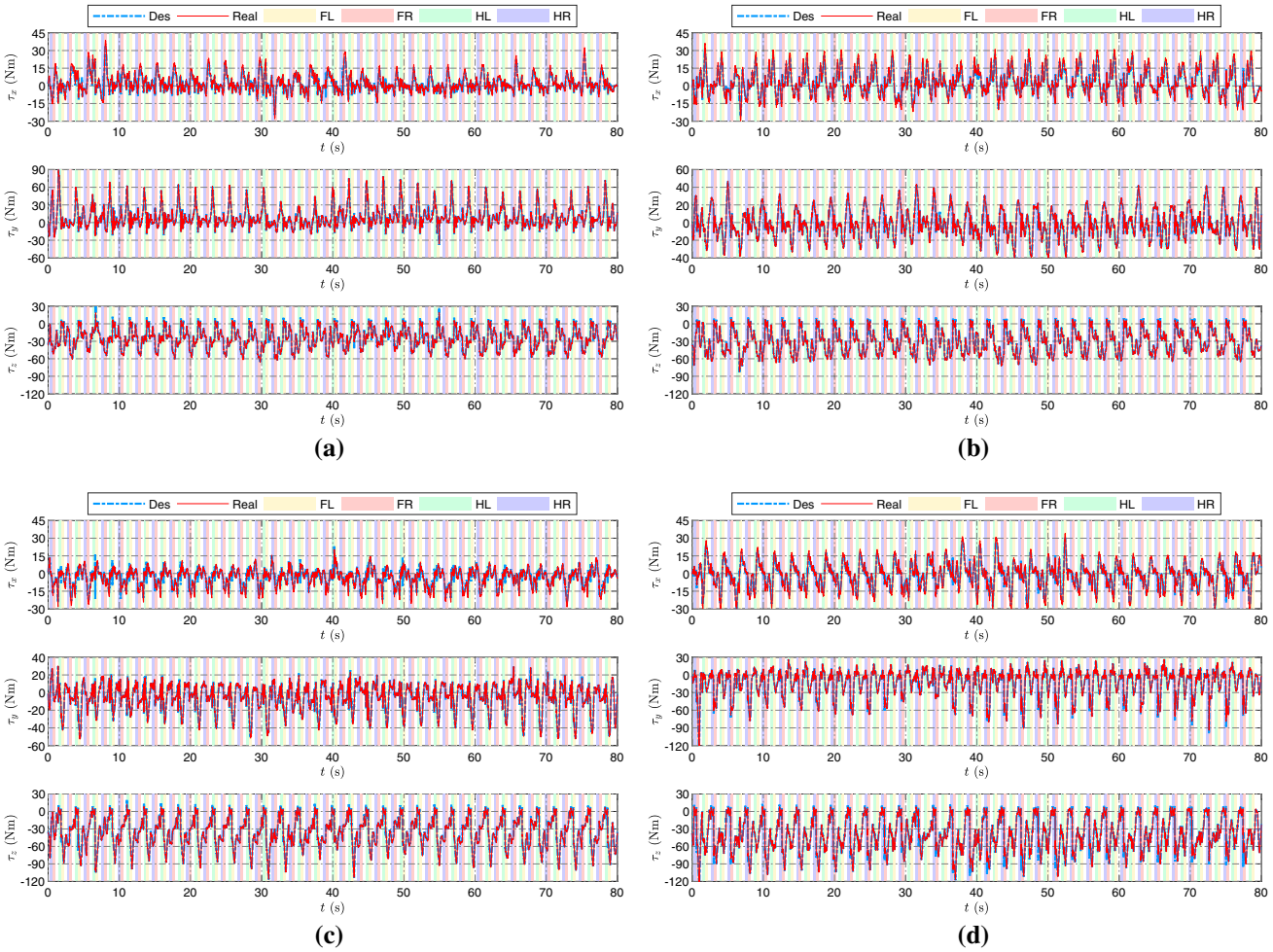


Fig. 8 Desired and real joint torques of each leg on the terrain shown in Fig. 6a. **a** FL. **b** FR. **c** HL. **d** HR. The torque limit for every joint is $[-120, 120]$ Nm

Table 1 Settings and results of the proposed method applied to every Half/Full walking cycle.

Test ID	Order	Continuity conditions						QP size		InF	Computation time (ms)			
		$\boldsymbol{p}_{\text{str}}$	$\dot{\boldsymbol{p}}_{\text{str}}$	$\ddot{\boldsymbol{p}}_{\text{str}}$	$\boldsymbol{p}_{\text{end}}$	$\dot{\boldsymbol{p}}_{\text{end}}$	$\ddot{\boldsymbol{p}}_{\text{end}}$	N_{var}	N_{con}	(%)	Max	Min	Mean	Std
Half–01	6	ccc	ccc	xxx	ccc	ccc	xxx	45	180	0	1.034	0.452	0.551	0.088
Half–02	6	ccc	ccc	xxx	ccc	cxc	xcx	45	180	0	1.300	0.427	0.530	0.122
Half–03	7	ccc	ccc	ccc	ccc	cxc	xcx	45	180	0	2.553	0.441	0.615	0.382
Half–04	7	ccc	ccc	ccc	ooo	oxo	xox	51	180	0	4.149	0.569	0.878	0.559
Full–01	6	ccc	ccc	xxx	ccc	ccc	xxx	81	360	0	2.415	0.927	1.426	0.347
Full–02	6	ccc	ccc	xxx	ccc	cxc	xcx	81	360	0	6.955	0.908	1.575	1.085
Full–03	7	ccc	ccc	ccc	ccc	cxc	xcx	81	360	0	4.690	0.956	1.565	0.775
Full–04	7	ccc	ccc	ccc	ooo	oxo	xox	87	360	0	9.479	1.193	2.684	2.011

$\boldsymbol{p}_{\text{str}}$, $\dot{\boldsymbol{p}}_{\text{str}}$, $\ddot{\boldsymbol{p}}_{\text{str}}$ and $\boldsymbol{p}_{\text{end}}$, $\dot{\boldsymbol{p}}_{\text{end}}$, $\ddot{\boldsymbol{p}}_{\text{end}}$ denote the initial and final values of $\boldsymbol{p}_t$, $\dot{\boldsymbol{p}}_t$, $\ddot{\boldsymbol{p}}_t$ of a motion segment, respectively. "c" and "o" indicate that the continuity of a quantity is considered as a hard constraint and a penalty term in the objective function of the QP problem (12), respectively, and "x" represents no continuity constraint. Three successive letters correspond to the x, y, and z components of a vector in $\mathbb{R}^3$. N_{var} is the number of independent variables after eliminating all the linear equality constraints and N_{con} is the number of linear inequality constraints, which are the constraints on contact forces. "InF" stands for "infeasible"

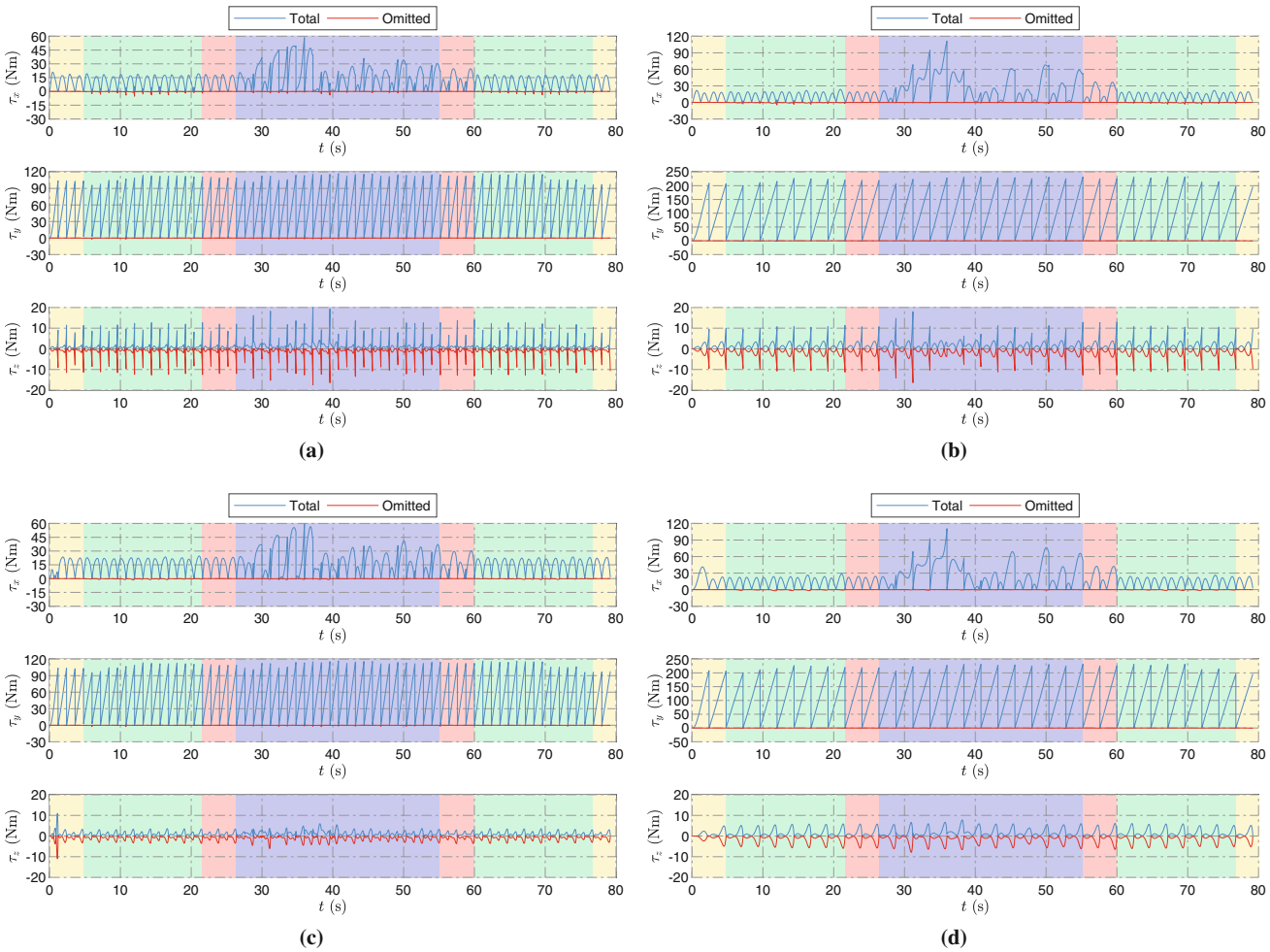


Fig. 9 Total and omitted torques in test **a** Half–01, **b** Full–01, **c** Half–03, and **d** Full–03

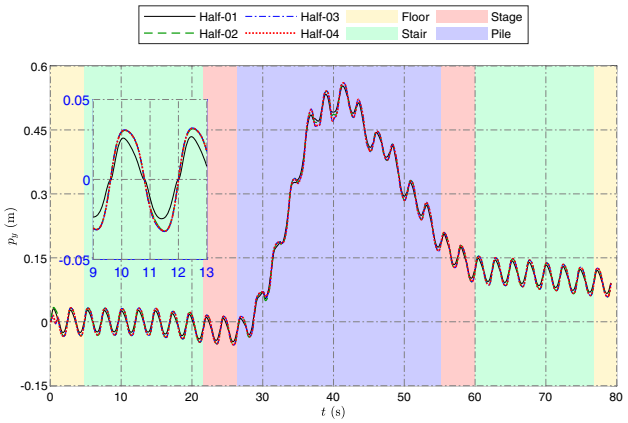
and stairs, footholds are selected within the detected flat areas such that the robot can walk straight forward. On the piles, their centers are determined and used as candidate footholds, among which we select the leftmost or rightmost reachable one for a swing leg or the nearest one ahead of the robot depending on which path we would like the robot to take, the leftmost, rightmost, or middle path on the piles. We let the motion generator repeatedly replan the motion for the next half walking cycle based on the terrain information consecutively updated by the perception system as the robot proceeds until the motion starts to be executed.

4.2.1 Hardware experiment results

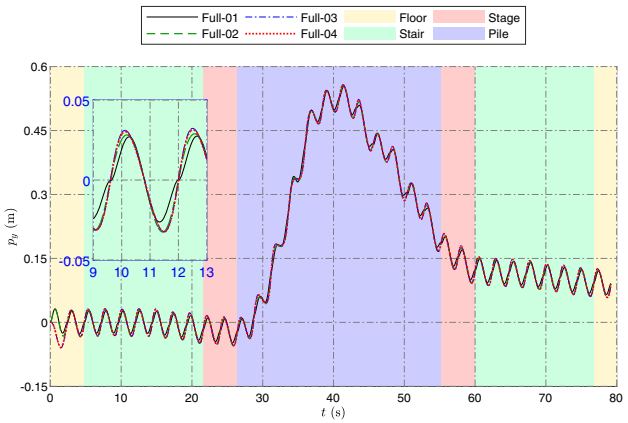
While the leftmost reachable foothold on piles is selected for every swing leg, the generated CoM trajectory is plotted as the blue dashdotted curve in Fig. 7. In this way, the robot successively traversed the terrain and followed the generated CoM trajectory very well, where the actually realized CoM trajectory is plotted as the red solid curve in Fig. 7. The joint torques of the robot are shown in Fig. 8, which shows that the desired joint torques determined by the motion controller can be nicely realized by the robot. Figure 6 exhibits the snapshots of key stages as well as the perceived and reconstructed terrain during robot's locomotion. Other foothold selection strategies on piles including the rightmost reachable one and the nearest one in front of the robot are also tried, in which cases the robot steadily walked over the testing terrain as well by following the trajectories generated online by the proposed method. These results are shown in the accompanying video, indicating that the motion generator can efficiently compute the required motions and work seamlessly between the vision system and the motion controller.

4.2.2 More numerical tests

By the above real experiment, we obtained a 3-D model of the testing terrain and continued to conduct a number of numerical tests on the proposed method with it, since the QP problem (12) can have many variants. In the above case, we used sex-



(a)



(b)

Fig. 10 CoM trajectories in y direction generated for every **a** half or **b** full walking cycle with different settings for the QP problem (12) as given in Table 1

Table 2 Results of the proposed method with different numbers of sample points. N_{sam} is the number of samples inside every motion phase

N_{sam}	QP size		InF	Computation time (ms)			
	N_{var}	N_{con}	(%)	Max	Min	Mean	Std
1	45	180	0	1.034	0.452	0.551	0.088
3	81	348	0	2.335	0.636	0.937	0.283
7	153	684	0	7.513	1.447	2.807	1.189
15	297	1356	0	36.449	6.022	15.692	7.926

Sample points at the phase conjunctions are as in Fig. 3. Other settings are as Half–01 in Table 1

tic polynomials for every half walking cycle and imposed the continuity of the CoM position and velocity at the conjunction of two half cycles as hard constraints in the QP problem. The sample points for checking the trajectory feasibility were selected as shown in Fig. 3. The order of polynomials, formulation of continuity condition, and selection of sample

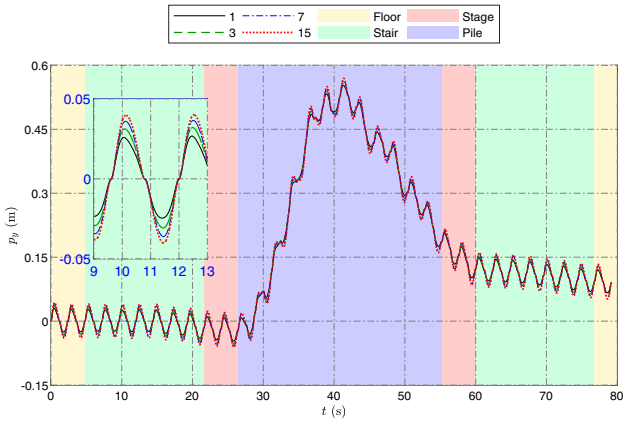


Fig. 11 CoM trajectories in y direction generated with different numbers of sample points for every half walking cycle

Table 3 Results of the proposed method with polynomials of different orders. Other settings are as Half–01 in Table 1

Order	QP size		InF	Computation time (ms)			
	N_{var}	N_{con}	(%)	Max	Min	Mean	Std
6	45	180	0	1.034	0.452	0.551	0.088
7	48	180	0	1.133	0.480	0.631	0.131
8	51	180	0	1.605	0.582	0.808	0.203
9	54	180	0	2.055	0.654	0.943	0.291

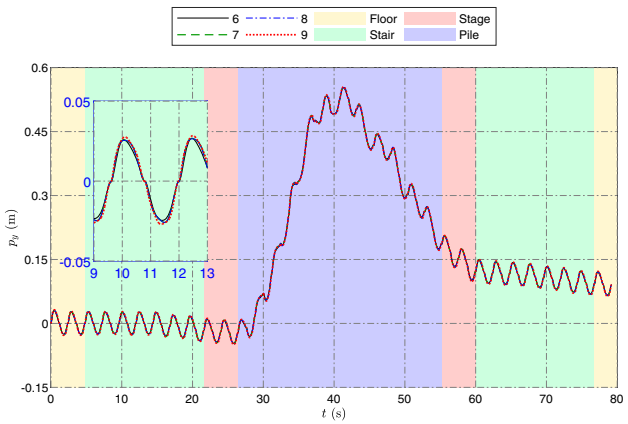


Fig. 12 CoM trajectories in y direction generated with polynomials of different orders for every half walking cycle

points can all be changed, and we would investigate how these changes influence the results.

First, we tried sextic/septic polynomials for every half/full walking cycle, in which some of the continuity conditions for the CoM position, velocity, and acceleration are added as penalties in the objective function of the QP problem instead of as constraints. The sample points were still selected as in Fig. 3. The settings and results of the proposed method are summarized in Table 1, where test Half–01 is the previous real experiment and the others are variants for comparison.

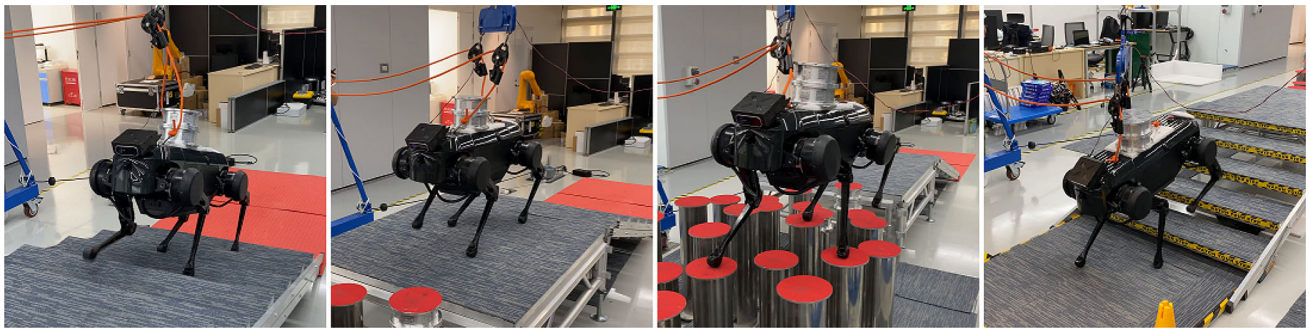


Fig. 13 Snapshots of the robot traversing the terrain while carrying the cargo without knowing its exact position

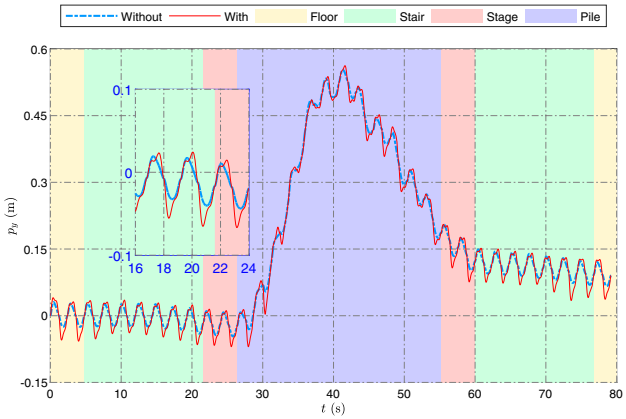


Fig. 14 CoM trajectories in y direction generated without and with consideration of the CoM uncertainty

It can be seen that the planning for full walking cycle takes a longer time since the number of sample points is doubled, leading to the doubled numbers of contact forces and contact force constraints in the QP. Moreover, we checked the feasibility of every generated CoM trajectory at every millisecond and observed no violation, though only a few sample points were used in the QP problem. Another important matter that we would like to verify is the omitted term in simplifying the centroidal dynamics as in Sect. 3.1. The omitted term represents a torque, which versus the total torque including the omitted term in four tests is plotted in Fig. 9, where the results in the other four tests are very similar and not included here. From them we see that the x and y components of the omitted torque are indeed small. In the z direction, interestingly, the omitted torque is almost equal but opposite to the total torque. Then, the locomotion generator actually considers twice as much as the total torque in the z direction. This favors the motion controller, which eventually needs to cover a smaller torque close to the total torque to produce the generated motion. The generated CoM trajectories are almost the same in x and z directions and only have some small variances in y, as shown in Fig. 10.

Table 4 Results with consideration of the CoM uncertainty. Other settings of the QP are same as test Half–01 in Table 1

QP size		InF	Computation time (ms)			
N_{var}	N_{con}	(%)	Max	Min	Mean	Std
225	1080	5.09	69.396	11.648	23.915	8.642

Next, we increased the number of sample points and kept all the other settings as test Half–01 in Table 1. The results are exhibited in Table 2 and the generated CoM trajectories are plotted in Fig. 11. Apparently, increasing the sample points increases the variables and constraints of the QP problem (12) and its computation time, while the generated CoM trajectories remain similar since the motion obtained in test Half–01 is feasible all the time. Adding more sample points does not bring more active constraints but does change the proportion of every term in the objective function of (12), causing the resulting trajectories to be slightly different in y.

Moreover, we changed the order of polynomials from test Half–01 and obtained the results in Table 3 and Fig. 12, which reveal that simply changing the orders of polynomials would not substantially change the results and even the differences in y are tiny. As a matter of fact, connecting two points with a high-order polynomial is naturally accompanied by large oscillation or lateral acceleration. Then, using a high-order polynomial as the CoM trajectory will induce large contact forces or overall length of the trajectory. However, since the contact forces and trajectory length have been considered in the objective function of (12), the computed trajectories tend to be low-order polynomials or have very small coefficients corresponding to the high-order terms even if higher-order polynomials are initially set in (12), unless it is necessary.

We have tried many other settings, in which cases the omitted term did not appear to be important either. Some settings were also tested on the real robot and all successfully enabled the robot to traverse the testing terrain. Since those results are similar to the ones reported above, we do not include them in this paper but would be happy to share upon request.

Table 5 Results of solving the MIQP problems (28) and (29) for a full walking cycle. The settings of polynomials, continuity conditions, and sample points are same as test Full–01 in Table 1. The leg's workspace (26) is considered. $N_{B\text{var}}$ and $N_{R\text{var}}$ are the number of binary and real variables, respectively

problem	MIQP size			InF	Computation time (ms)			
	$N_{B\text{var}}$	$N_{R\text{var}}$	N_{con}	(%)	Max	Min	Mean	Std
(28)	7–10	225–279	1198–1307	0	843.32	573.88	687.21	81.17
(29)	250–350	765–1065	4058–4804	0	121659.0	27129.7	55789.3	22512.0

4.3 Experiments with an unknown load on the robot

We let our robot traverse the same terrain and carry a cargo of 10 kg whose position on the robot is not exactly known or measured, as shown in Fig. 13. The convex region around the nominal CoM position is taken to be an octahedron (i.e., $N_V = 6$), as depicted in Fig. 1. Figure 14 shows the CoM trajectory generated by solving (19) for every half walking cycle used in this experiment in comparison with the one in the previous experiment. Here, the CoM movement in the y direction becomes slightly greater such that the octahedron can be well above the support area, while its movement in x or z does not change obviously (not shown). Table 4 shows the performance of the proposed method in this case to be compared with test Half–01 in Table 1. While there is a small difference in the generated CoM trajectories, the computation time in the case of considering the CoM uncertainty increases significantly since the size of the QP problem has increased several times. Furthermore, it is noticed that the CoM trajectory here can be infeasible sometimes. We think that the groups of constraints at the sample points cause the CoM motion to be more acute and likely to be infeasible somewhere away from the sample points. The violation percentage can be reduced to 2% by taking 3 times more sample points, but the computation time rises to about 1 s. To solve (19) and apply the method online, we used the same setting of sample points as in the first experiment and let the motion controller to overcome any local motion infeasibility in the real experiment. As a result, the robot succeeded in carrying the cargo over the terrain. The whole motion is shown in the accompanying video.

4.4 Experiments with variable walking pattern

We now compute the CoM trajectory for the robot to walk over the piles by solving (29) for every full walking cycle. Here we still call it "walking cycle" since every foot is required to take a step one after another, but it should be noted that the orders and timings of steps are determined by (29) rather than predefined. The time interval in Fig. 5 is taken to be 0.1 s and the lower bound of the swing duration is set to 0.4 s, i.e., $K_{Si} = 4$. With this setting, the computation time of (29) is shown in Table 5 but unfortunately it is too long for the MIQP problem to be solved online. So far, we offline solved (29) with the recorded terrain model and let the robot track the computed CoM trajectory online. The generated and realized CoM trajectories are plotted in Fig. 15 and the joint torques in controlling the robot are plotted in Fig. 16. Figure 17 reveals the actual walking patterns that the robot used to traverse the piles, among which we notice four different step sequences, as highlighted in Fig. 18. It is also observed that the swing duration of every step tends to be the lower bound and the robot is still inclined to have one swing foot at a time even though we did not explicitly impose such a constraint in (29). This is due to the contact force term J_{grf} defined by (13) in the objective function, which favors the case of more supporting legs such that the contact force on each foot at a time and the sum of their squares can be smaller even if the resultant contact wrench needed for a motion remains the same. This experiment is also collected in the accompanying video, which shows that the robot took a different path with different footholds to traverse the piles. In addition, the computation time for solving (28) is also provided in Table 5. Without considering the step sequence and timing variances, it is still possible to solve (28) online for just determining the footholds and the CoM trajectory together.

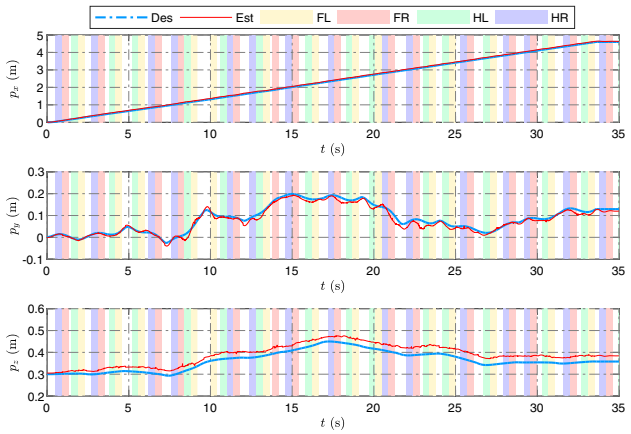


Fig. 15 CoM trajectories generated by solving the MIQP problem (29) and realized by the robot on the piles

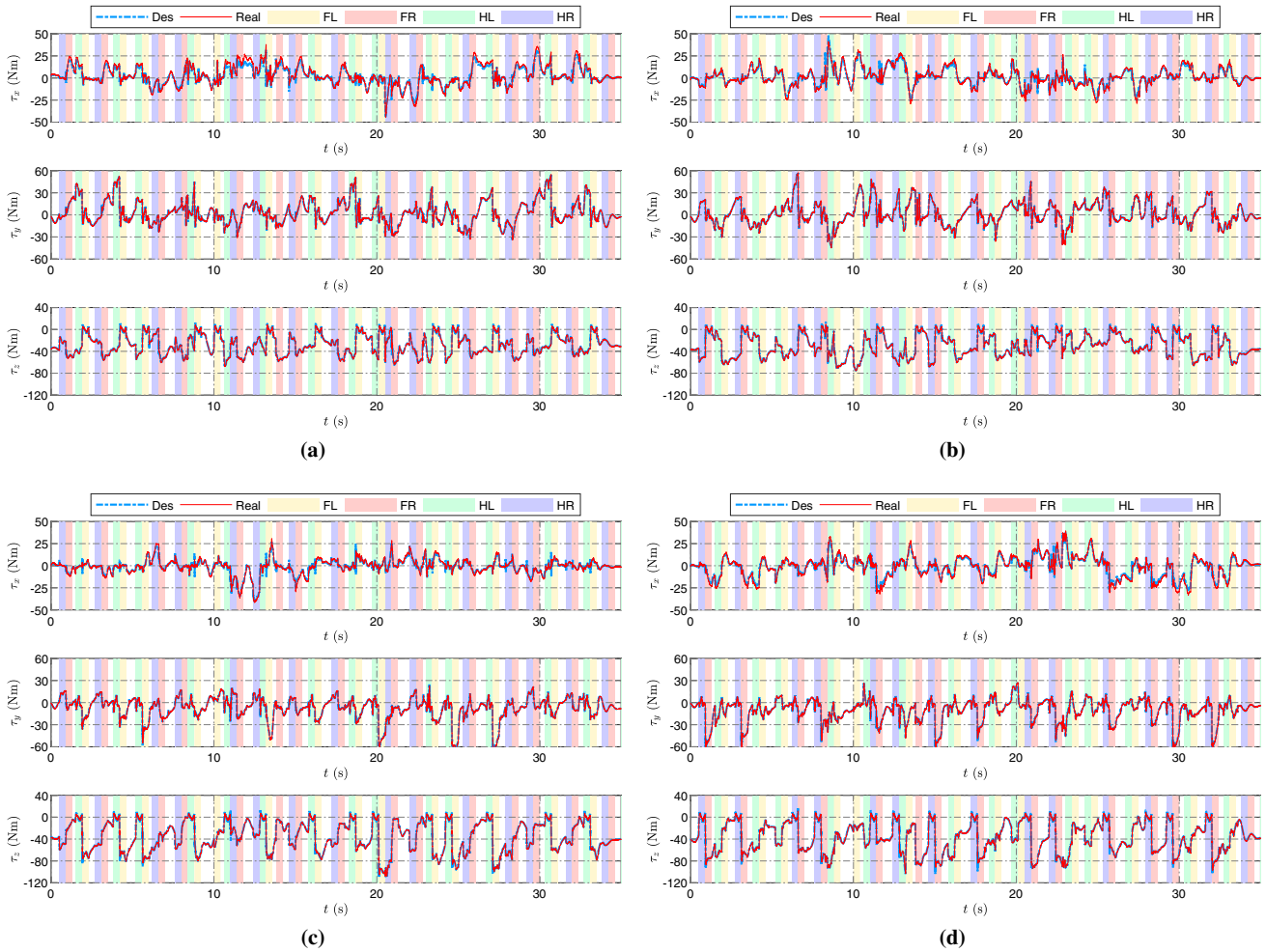


Fig. 16 Desired and real joint torques of each leg on the piles. **a** FL. **b** FR. **c** HL. **d** HR

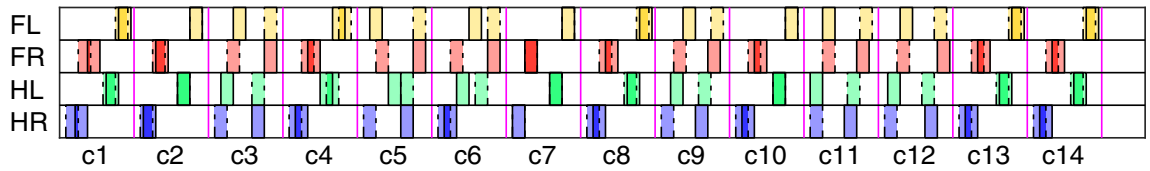


Fig. 17 Comparison of the walking patterns automatically determined by (29) (marked by the solid lines) and the predefined one (marked by the dashed lines) for the robot walking over the piles. The step sequences and timings of determined walking patterns are varying, while the swing duration of every step (marked by the colored areas) tends to be the lower bound

5 Conclusion and future work

In this paper, we propose a locomotion generation method for quadruped robots. We first discover that the quadratic term in the robot's centroidal dynamics induced by the cross product of the position change and acceleration of the robot's CoM is minor and can be omitted for a medium-length motion. Then, we parameterize the CoM trajectory as polynomials and write its optimization as an easily-solvable QP problem. Furthermore, the proposed method is extended to include the consideration of the CoM uncertainty and the leg workspace as additional linear constraints in the QP problem. We also allow variable step sequences, timings, and footholds and rewrite the computing of CoM trajectory as a single MIQP problem. The proposed locomotion generator together with a vision system and a motion tracking controller has been implemented on a real quadruped robot such that it can nicely execute generated motions to traverse challenging terrains.

In the future, we will explore possible ways to combine the angular motion with the CoM trajectory in the locomo-

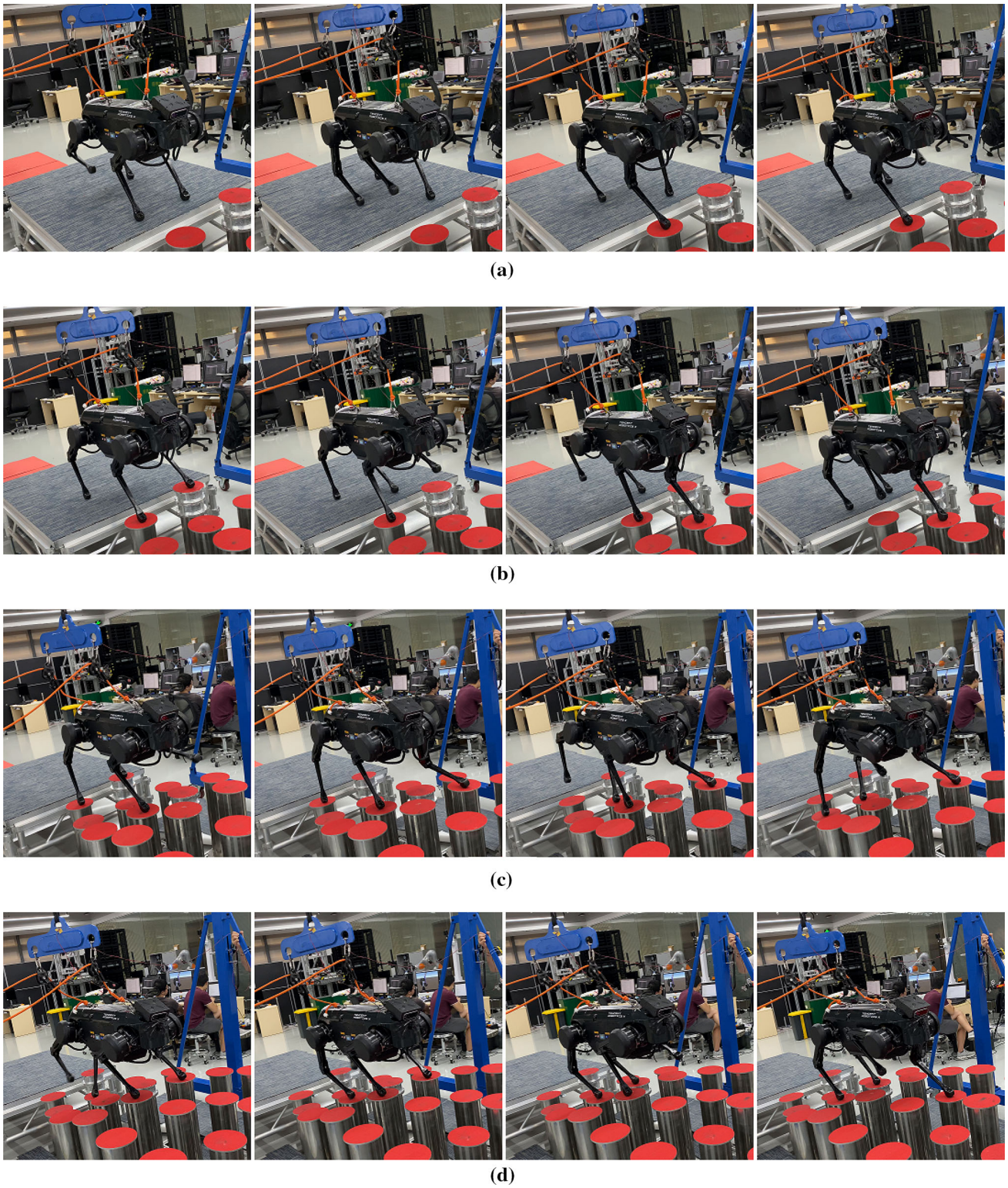


Fig. 18 Four different step sequences automatically determined by (29) for the robot walking over the piles. **a** HR-FR-HL-FL. **b** HL-FL-HR-FR. **c** FL-HL-HR-FR. **d** HR-HL-FL-FR

tion generation such that the generated motion of the robot is more effective and natural. Moreover, we will continue to explore more efficient ways to formulate and solve the TO problems. One way is to combine the modern machine learning methods, which can be trained by the motion trajectories generated by model-based methods and help quickly deter-

mine some settings in the TO. Last but not least, the capability of our perception system will be improved in order for our robot to travel over a wider variety of terrains and further verify the capability of the proposed method.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10514-022-10068-3.

Acknowledgements We would like to thank DeepRobotics Co. Ltd for the assistance in customizing the API and communication protocol between the vision and control systems of our robot.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

6 Appendix—motion controller

Similar to many existing motion controllers for legged robots (Gehring et al., 2013; Park et al., 2015; Focchi et al., 2017; Bledt et al., 2018; Di Carlo et al., 2018; Feng et al., 2015; Caron et al., 2015a; Kuindersma et al., 2016; Zheng et al., 2019), our controller computes the required contact forces to determine the feedforward joint torques and then the final joint torques by adding the feedback joint torques, as depicted by the green block in Fig. 2.

6.1 Computing of contact forces

The first step is aimed at computing the required contact forces for the robot to produce the reference CoM trajectory and base's angular motion, which can be written as the following QP problem:

$$\begin{cases} \min\limits_{f_1, f_2, \ldots, f_{N_C}} & J_w + J_\tau + J_f \\ \text{subject to} & \boldsymbol{N}_i^T \boldsymbol{f}_i \leq \boldsymbol{0} \\ & \boldsymbol{n}_i^T \boldsymbol{f}_i \leq f_i^U \\ & i = 1, 2, \ldots, N_C. \end{cases} \tag{30}$$

The above objective function consists of three terms to integrate different control goals. The purpose of the first term J_w is to realize the reference CoM trajectory and angular motion of the robot's base. From (4), it is written as

$$J_w = \left(\boldsymbol{w}^{\text{des}} - \boldsymbol{G}\boldsymbol{f}\right)^T \boldsymbol{W}_w \left(\boldsymbol{w}^{\text{des}} - \boldsymbol{G}\boldsymbol{f}\right), \tag{31}$$

where $\boldsymbol{W}_w \in \mathbb{R}^{6\times 6}$ is a positive-definite diagonal weight matrix and $\boldsymbol{w}^{\text{des}}$ is the desired wrench given by

$$\boldsymbol{w}^{\text{des}} = \begin{bmatrix} m(\ddot{\boldsymbol{p}}_G^{\text{des}} - \boldsymbol{g}) \\ m\hat{\boldsymbol{p}}_G(\ddot{\boldsymbol{p}}_G^{\text{des}} - \boldsymbol{g}) + \dot{\mathcal{L}}^{\text{des}} \end{bmatrix}. \tag{32}$$

Here, $\ddot{\boldsymbol{p}}_G^{\text{des}}$ is the desired CoM acceleration derived from the reference acceleration $\ddot{\boldsymbol{p}}_G^{\text{ref}}$, velocity $\dot{\boldsymbol{p}}_G^{\text{ref}}$, and position $\boldsymbol{p}_G^{\text{ref}}$ together with the actual velocity $\dot{\boldsymbol{p}}_G$ and position $\boldsymbol{p}_G$ of the CoM using the proportional-derivative (PD) control law:

$$\ddot{\boldsymbol{p}}_G^{\text{des}} = \ddot{\boldsymbol{p}}_G^{\text{ref}} + \boldsymbol{K}_d^G \left(\dot{\boldsymbol{p}}_G^{\text{ref}} - \dot{\boldsymbol{p}}_G\right) + \boldsymbol{K}_p^G \left(\boldsymbol{p}_G^{\text{ref}} - \boldsymbol{p}_G\right), \tag{33}$$

where $\boldsymbol{K}_p^G$ and $\boldsymbol{K}_d^G$ are PD gains. From (5), we write the desired time derivative of angular momentum as

$$\dot{\mathcal{L}}^{\text{des}} = \mathcal{I}_0^S \dot{\boldsymbol{\omega}}_0^{\text{des}} - (\mathcal{I}_0^S \boldsymbol{\omega}_0^S) \times \boldsymbol{\omega}_0^S, \tag{34}$$

where $\dot{\boldsymbol{\omega}}_0^{\text{des}}$ is the desired angular acceleration as

$$\dot{\boldsymbol{\omega}}_0^{\text{des}} = \dot{\boldsymbol{\omega}}_0^{\text{ref}} + \boldsymbol{K}_d^\omega \left(\boldsymbol{\omega}_0^{\text{ref}} - \boldsymbol{\omega}_0\right) + \boldsymbol{K}_p^\omega \left(\boldsymbol{R}_0^{\text{ref}} \boldsymbol{R}_0^T\right)^\vee, \tag{35}$$

where $\boldsymbol{R}_0^{\text{ref}}, \boldsymbol{R}_0 \in SO(3)$ represent the reference and actual orientations of the robot's base with respect to the global frame, respectively, and $\left(\boldsymbol{R}_0^{\text{ref}} \boldsymbol{R}_0^T\right)^\vee$ is the associated vector of rotation to change $\boldsymbol{R}_0$ to $\boldsymbol{R}_0^{\text{ref}}$.

The second term J_τ is used to penalize high joint toques and written as

$$J_\tau = \boldsymbol{f}^T \boldsymbol{J}_\tau \boldsymbol{W}_\tau \boldsymbol{J}_\tau^T \boldsymbol{f}, \tag{36}$$

where $\boldsymbol{W}_\tau \in \mathbb{R}^{N_J \times N_J}$ is a positive-definite diagonal weight matrix and $\boldsymbol{J}_\tau^T \in \mathbb{R}^{N_J \times 3N_C}$ is the transposed Jacobian matrix, comprising the last N_J rows of $\boldsymbol{J}^T$ in (1) and converting contact forces to joint torques.

The last term J_f regulates the contact forces, i.e.,

$$J_f = \left(\boldsymbol{f}^{\text{des}} - \boldsymbol{f}\right)^T \boldsymbol{W}_f \left(\boldsymbol{f}^{\text{des}} - \boldsymbol{f}\right), \tag{37}$$

where $\boldsymbol{W}_f \in \mathbb{R}^{3N_C \times 3N_C}$ is a weight matrix and $\boldsymbol{f}^{\text{des}}$ comprises the desired contact forces. When a foot is contacting and expected to remain on the ground, we take its desired contact force to be the optimal value obtained by solving the QP problem (30) in the previous control cycle. For a foot that is in the air but expected to contact or is contacting but expected to leave the ground, we simply set its desired contact force to zero.

6.2 Computing of joint torques

The second step of the motion controller is to compute the joint torques for controlling the robot. Since the leg's mass of a quadruped robot is often negligible, we can skip the robot's whole-body dynamics and simply compute the joint torques for controlling the robot as (Focchi et al., 2017; Mastalli et al., 2017)

$$\boldsymbol{\tau} = -\boldsymbol{J}_{\tau}^{T}\boldsymbol{f}^{\text{opt}} + \tilde{\boldsymbol{K}}_{d}^{q}\left(\dot{\boldsymbol{q}}^{\text{ref}} - \dot{\boldsymbol{q}}\right) + \tilde{\boldsymbol{K}}_{p}^{q}\left(\boldsymbol{q}^{\text{ref}} - \boldsymbol{q}\right), \quad (38)$$

where $\dot{\boldsymbol{q}}$ and $\boldsymbol{q}$ with or without the superscript "ref" are the reference or current joint velocities and angles, respectively, and $\tilde{\boldsymbol{K}}_{p}^{q}$ and $\tilde{\boldsymbol{K}}_{d}^{q}$ are the PD gains mapping the errors in joint angles and velocities directly into the joint torques for joint trajectory tracking. Here, the reference joint trajectories $\boldsymbol{q}^{\text{ref}}$ are calculated through inverse kinematics with the reference footholds and body trajectories determined by the motion generator, as illustrated in Fig. 2 and discussed in this paper.

References

Aceituno-Cabezas, B., Dai, H., Cappelletto, J., Grieco, J. C., & Fernández-López, G. (2017). A mixed-integer convex optimization framework for robust multilegged robot locomotion planning over challenging terrain. In *IEEE/RSJ International conference on intelligent robots and systems* (pp. 4467–4472).

Aceituno-Cabezas, B., Mastalli, C., Dai, H., Focchi, M., Radulescu, A., Caldwell, D. G., et al. (2018). Simultaneous contact, gait and motion planning for robust multi-legged locomotion via mixed-integer convex optimization. *IEEE Robotics and Automation Letters, 3*(3), 2531–2538.

Barasuol, V., Buchli, J., Semini, C., Frigerio, M., De Pieri, E. R., & Caldwell, D. G. (2013). A reactive controller framework for quadrupedal locomotion on challenging terrain. In *Proceedings of the IEEE international conference on robotics and automation*. (pp 2554–2561).

Bellicoso, C. D., Jenelten, F., Fankhauser, P., Gehring, C., Hwangbo, J., & Hutter, M. (2017). Dynamic locomotion and whole-body control for quadrupedal robots. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp 3359–3365).

Bellicoso, C. D., Jenelten, F., Gehring, C., & Hutter, M. (2018). Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots. *IEEE Robotics and Automation Letters, 3*(3), 2261–2268.

Bledt, G., Powell, M. J., Katz, B., Di 'Carlo, J., Wensing, P. M., & Kim, S. (2018). MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. pp 2245–2252.

Bretl, T., & Lall, S. (2008). Testing static equilibrium for legged robots. *IEEE Transactions on Robotics, 24*(4), 794–807.

Caron, S., & Kheddar, A. (2016). Multi-contact walking pattern generation based on model preview control of 3D COM accelerations. *Proceedings of the IEEE-RAS international conference on humanoid robots*. (pp. 550–557).

Caron, S., Pham, Q. C., & Nakamura, Y. (2015a). Leveraging cone double description for multi-contact stability of humanoids with applications to statics and dynamics. In *Robotics: science and system*.

Caron, S., Pham, Q. C., & Nakamura, Y. (2015b). Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 5107–5112).

Caron, S., Pham, Q., & Nakamura, Y. (2017). ZMP support areas for multicontact mobility under frictional constraints. *IEEE Transactions on Robotics, 33*(1), 67–80.

Carpentier, J., Tonneau, S., Naveau, M., Stasse, O., & Mansard, N. (2016). A versatile and efficient pattern generator for generalized legged locomotion. *Proceedings of the IEEE International Conference on Robotics and Automation*. (pp. 3555–3561).

CPLEX (2010). User's manual for CPLEX. https://www.ibm.com/hk-en/analytics/cplex-optimizer

Dai, H. K., & Tedrake, R. (2016). Planning robust walking motion on uneven terrain via convex optimization. In *Proceedings of the IEEE-RAS international conference on humanoid robots*. (pp. 104–109).

DeepRobotics. (2017). http://www.deeprobotics.cn/

Deits, R., & Tedrake, R. (2014). Footstep planning on uneven terrain with mixed-integer convex optimization. In *Proceedings of the IEEE-RAS international conference on humanoid robots*. (pp. 279–286).

Del Prete, A., Tonneau, S., & Mansard, N. (2016). Fast algorithms to test robust static equilibrium for legged robots. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 1601–1607).

Di Carlo, J., Wensing, P.M., Katz, B., Bledt, G., & Kim, S. (2018). Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control. In Proceedings of the IEEE/RSJ international conference on intelligent robots and systems. (pp. 1–9).

Di Gaspero, L. (1998). QuadProg++. http://quadprog.sourceforge.net/

Fankhauser, P., Bjelonic, M., Dario Bellicoso, C., Miki, T., & Hutter, M. (2018). Robust rough-terrain locomotion with a quadrupedal robot. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 5761–5768).

Farshidian, F., Neunert, M., Winkler, A. W., Gonzalo, R., & Buchli, J. (2017). An efficient optimal planning and control framework for quadrupedal locomotion. *Proceedings of the IEEE international conference on robotics and automation*. (pp. 93–100).

Feng, S. Y., Whitman, E., Xinjilefu, X., & Atkeson, C. (2015). Optimization-based full body control for the DARPA robotics challenge. *Journal of Field Robotics, 32*(2), 293–312.

Fernbach, P., Tonneau, S., & Taïx, M. (2018). CROC: Convex resolution of centroidal dynamics trajectories to provide a feasibility criterion for the multi contact planning problem. *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp. 8367–8373).

Ferreau, H. J., Kirches, C., Potschka, A., Bock, H. G., & Diehl, M. (2014). qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation, 6*(4), 327–363.

Focchi, M., Del Prete, A., Havoutis, I., Featherstone, R., Caldwell, D. G., & Semini, C. (2017). High-slope terrain locomotion for torque-controlled quadruped robots. *Auton Robots, 41*(1), 259–272.

Gehring, C., Coros, S., Hutter, M., Bloesch, M., Hoepflinger, M. A., Siegwart, R. (2013). Control of dynamic gaits for a quadrupedal robot. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 3287–3292).

Gill, P. E., Murray, W., & Saunders, M. A. (2005). SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review, 47*(1), 99–131.

GLPK. (2000). GNU linear programming kit. http://www.gnu.org/software/glpk/glpk.html

Grandia, R., Farshidian, F., Ranftl, R., & Hutter, M. (2019). Feedback MPC for torque-controlled legged robots. *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp. 4730–4737).

GUROBI, (2014). Gurobi optimizer reference manual. http://www.gurobi.com/

Herzog, A., Schaal, S., & Righetti, L. (2016). Structured contact force optimization for kino-dynamic motion generation. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp. 2703–2710).

Hirai, S. (1991). Analysis and planning of manipulation using the theory of polyhedral convex cones. Ph. D. Dissertation, Kyoto University.

Hirukawa, H., Hattori, S., Harada, K., Kajita, S., Kaneko, K., Kanehiro, F., Fujiwara, K., Morisawa, M. (2006). A universal stability criterion of the foot contact of legged robots - adios zmp. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 1976–1983).

Huang, Q., Yokoi, K., Kajita, S., Kaneko, K., Arai, H., Koyachi, N., & Tanie, K. (2001). Growth distance: New measures for object separation and penetration. *IEEE Transactions on Robotics and Automation, 17*(3), 280–289.

Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C. D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., & Hoepflinger, M. (2016). ANYmal: A highly mobile and dynamic quadrupedal robot. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp. 38–44).

Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., & Hirukawa, H. (2003). Biped walking pattern generation by using preview control of zero-moment point. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 1620–1626).

Kalakrishnan, M., Buchli, J., Pastor, P., Mistry, M., & Schaal, S. (2011). Learning, planning, and control for quadruped locomotion over challenging terrain. *The International Journal of Robotics Research, 30*(2), 236–258.

Kanehiro, F., Suleiman, W., Lamiraux, F., Yoshida, E., & Laumond, J. P. (2008). Integrating dynamics into motion planning for humanoid robots. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp. 660–667).

Koolen, T., De Boer, T., Rebula, J., Goswami, A., & Pratt, J. (2012). Capturability-based analysis and control of legged locomotion, Part 1: Theory and application to three simple gait models. *The International Journal of Robotics Research, 31*(9), 1094–1113.

Kuffner, J. J., Kagami, S., Nishiwaki, K., Inaba, M., & Inoue, H. (2002). Dynamically-stable motion planning for humanoid robots. *Autonomous Robots, 12*(1), 105–118.

Kuindersma, S., Deits, R., Fallon, M., Valenzuela, A., Dai, H. K., Permenter, F., et al. (2016). Optimization-based locomotion planning, estimation, and control design for the Atlas humanoid robot. *Autonomous Robots, 40*(3), 429–455.

Mastalli, C., Budhiraja, R., Merkt, W., Saurel, G., Hammoud, B., Naveau, M., Carpentier, J., Righetti, L., Vijayakumar, S., & Mansard, N. (2020). Crocoddyl: An effcient and versatile framework for multi-contact optimal control. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 2536–2542).

Mastalli, C., Focchi, M., Havoutis, I., Radulescu, A., Calinon, S., Buchli, J., Caldwell, D. G., & Semini, C. (2017). Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 1096–1103).

Mastalli, C., Havoutis, I., Focchi, M., Caldwell, D. G., & Semini, C. (2020). Motion planning for quadrupedal locomotion: coupled planning, terrain mapping, and whole-body control. *IEEE Transactions on Robotics, 36*(6), 1635–1648.

Mordatch, I., Todorov, E., & Popović, Z. (2012). Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics, 31*(4), 43.

MOSEK. (2014). The MOSEK optimization software. http://www.mosek.com/.

Nagasaka, K., Masayuki, I., & Inoue, H. (1999). Dynamic walking pattern generation for a humanoid robot based on optimal gradient method. In *Proceedings of the IEEE systems, man, and cybernetics (SMC) society*. (pp. 908–913).

Neo, E. S., Yokoi, K., Kajita, S., & Tanie, K. (2007). Whole-body motion generation intergrating operator's intention and robot's autonomy in controlling humanoid robots. *IEEE Transactions on Robotics, 23*(4), 763–775.

Neunert, M., Farshidian, F., Winkler, A. W., & Buchli, J. (2017). Trajectory optimization through contacts and automatic gait discovery for quadrupeds. *IEEE Robotics and Automation Letters, 2*(3), 1502–1509.

Neunert, M., Stäuble, M., Giftthaler, M., Bellicoso, C. D., Gehring, C., Hutter, M., & Buchli, J. (2018). Whole-body nonlinear model predictive control through contacts for quadrupeds. *IEEE Robotics and Automation Letters, 3*(3), 1458–1465.

Orsolino, R., Focchi, M., Caron, S., Raiola, G., Barasuol, V., Caldwell, D. G., & Semini, C. (2020). Feasible region: an actuation-aware extension of the support region. *IEEE Transactions on Robotics, 34*(4), 1239–1255.

Orsolino, R., Focchi, M., Mastalli, C., Dai, H., Caldwell, D. G., & Semini, C. (2018). Application of wrench-based feasibility analysis to the online trajectory optimization of legged robots. *IEEE Robotics and Automation Letters, 3*(4), 3363–3370.

Park, H. W., Wensing, P. M., & Kim, S. (2015). Online planning for autonomous running jumps over obstacles in high-speed quadrupeds. In *Proceedings of robotics: science and systems*.

Park, J., & Youm, Y. (2007). General ZMP preview control for bipedal walking. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 2682–2687).

Perrin, N., Lau, D., & Padois, V. (2015). Effective generation of dynamically balanced locomotion with multiple non-coplanar contacts. In *Proceedings of the international symposium on robotics research*. (pp. 1–16).

Ponton, B., Herzog, A., Del Prete, A., Schaal, S., & Righetti, L. (2018). On time optimization of centroidal momentum. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 5776–5782).

Ponton, B., Herzog, A., Schaal, S., & Righetti, L. (2016). A convex model of humanoid momentum dynamics for multi-contact motion generation. *Proceedings of the IEEE-RAS international conference on humanoid robots*. (pp. 842–849).

Pratt, J., Carff, J., Drakunov, S., & Goswami, A. (2006). Capture point: a step toward humanoid push recovery. In *Proceedings of the IEEE-RAS international conference on humanoid robots*. (pp. 200–207).

Pratt, J., Koolen, T., De Boer, T., Rebula, J., Cotton, S., Carff, J., et al. (2012). Capturability-based analysis and control of legged locomotion, Part 2: Application to M2V2, a lower-body humanoidation to m2v2, a lower-body humanoid. *The International Journal of Robotics Research, 31*(10), 1117–1133.

Qiu, Z., Escande, A., Micaelli, A., & Robert, T. (2011). Human motions analysis and simulation based on a general criterion of stability. In *Proceedings of the international symposium on digital human modeling*. (pp. 1–8).

Raibert, M., Blankespoor, K., Nelson, G., & Playter, R. (2008). BigDog, the rough-terrain quadruped robot. In *IFAC proceedings*. (vol. 41, 2, pp. 10,822–10,825).

Saida, T., Yokokoji, Y., & Yoshikawa, T. (2003). FSW (feasible solution of wrench) for multi-legged robots. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 3815–3820).

Semini, C., Tsagarakis, N. G., Guglielmino, E., Focchi, M., Cannella, F., & Caldwell, D. G. (2011). Design of HyQ - a hydraulically and electrically actuated quadruped robot. *Proceedings of the institution of mechanical engineers, Part I: Journal of Systems and Control Engineering,225*(6), 831–849.

Sugihara, T. (2008). Simulated regulator to synthesize ZMP manipulation and foot location for autonomous control of biped robots. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 1264–1269).

Tonneau, S., Fernbach, P., Del Prete, A., Pettré, J., & Mansard, N. (2018). 2PAC: Two-point attractors for center of mass trajectories in multi-contact scenarios. *ACM Transactions on Graphics, 37*(5), 176.

Unitree. (2017). http://www.unitree.cc/

Vukobratović, M., & Borovac, B. (2004). Zero-moment point: Thirty five years of its life. *International Journal of Humanoid Robotics, 1*(1), 157–173.

Wäechter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming, 106*(1), 25–57.

Wieber, P. B. (2006). Holonomy and nonholonomy in the dynamics of articulated motion. *Lecture Notes in Control and Information Sciences, 340,* 411–425.

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

Winkler, A.W., Farshidian, F., Neunert, M., Pardo, D., & Buchli, J. (2017a). Online walking motion and foothold optimization for quadruped locomotion. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 5308–5313).

Winkler, A. W., Farshidian, F., Pardo, D., Neunert, M., & Buchli, J. (2017b). Fast trajectory optimization for legged robots using vertex-based ZMP constraints. *IEEE Robotics and Automation Letters, 2*(4), 2201–2208.

Winkler, A. W., Havoutis, I., Bazeille, S., Ortiz, J., Focchi, M., Dillmann, R., Caldwell, D., & Semini, C. (2014). Path planning with force-based foothold adaptation and virtual model control for torque controlled quadruped robots. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 6476–6482).

Winkler, A. W., Mastalli, C., Havoutis, I., Focchi, M., Caldwell, D. G., & Semini, C. (2015). Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain. In *Proceedings of the IEEE international conference on robotics and automation*. (pp. 5148–5154).

Zheng, Y., Lin, M. C., Manocha, D., Adiwahono, A. H., & Chew, C. M. (2010). A walking pattern generator for biped robots on uneven terrains. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems*. (pp 4483–4488).

Zheng, Y., Liao, S. W., & Yamane, K. (2019). Humanoid locomotion control and generation based on contact wrench cones. *International Journal of Humanoid Robotics, 16*(5), 1950021.

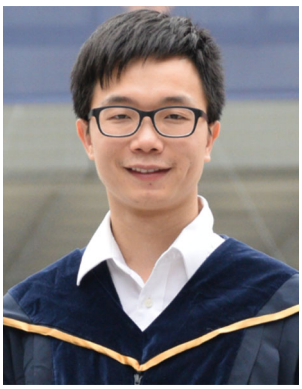
Zheng, Y., & Yamane, K. (2015). Generalized distance between compact convex sets: Algorithms and applications. *IEEE Transactions on Robotics, 31*(4), 988–1003.

Xinyang Jiang received his M.Res degree in mechanical engineering from Beijing Institute of Technology in 2019. Since July 2019, he is a research engineer at Tencent Robotics X. His research focuses on motion trajectory planning and control of legged robots and wheel -legged robots.

Wanchao Chi received his Ph.D. degree in Mechatronics and Design from Nanyang Technological University (NTU), Singapore in 2016. His research scope covered biomimetics, mechanical design, perception, path planning, collision avoidance, and dynamics modelling and control of unmanned aerial vehicles (UAVs). He has lead or contributed to the technical development of several UAV-related projects funded by Singapore government agencies, ranging from drone traffic management to control of formation flight. He joined Tencent Robotics X in 2018, and has been mainly working on projects of quadrupedal robots since then. His current research interests include dynamics control, motion planning, state estimation, foot-eye calibration of legged robots.

Yu Zheng received the Ph.D. degree in mechatronics from Shanghai Jiao Tong University, China, in 2007 and the Ph.D. degree in computer science from the University of North Carolina at Chapel Hill in 2014. Between 2007 and 2009, he was a Postdoctoral Research Fellow with the Department of Mechanical Engineering, National University of Singapore. He worked as a Lab Associate, a Research Associate, and finally a Postdoctoral Researcher at Disney Research Pittsburgh between 2010 and 2014. From 2014 to 2018, he was an Assistant Professor with the Department of Electrical and Computer Engineering, University of Michigan-Dearborn. He joined Tencent Robotics X in September 2018 and currently is a Principal Research Scientist and Team Lead of the Control Center. His research interests include multi-contact/multi-body robotic systems, dexterous grasping and manipulation, legged locomotion, and various algorithms for robotics. He serves as an Associate Editor for IEEE Robotics and Automation Letters.

Shenghao Zhang received the M.S. degree in computer application technology from Peking University, China, in 2019. He is currently a researcher in Tencent Robotics X. His research interests include state estimation, sensor fusion, simultaneous localization and mapping, and 3D reconstruction.



Yonggen Ling receives his Ph.D. at the Robotics Institute, the Hong Kong University of Science and Technology in 2017. He is now a senior researcher in Robotics X, Tencent. He is interested in SLAM, object pose estimation, visual-inertial fusion, visual-tactile fusion and autonomous navigation.

Jiafeng Xu received the B.S. and M.S. degrees in mechatronic engineering from the Beijing Institute of Technology, Beijing, China, in 2017 and 2020, respectively. He is currently working with Tencent Robotics X. His research interests include robot system identification and motion control of quadruped robot and robotic dynamic systems.

Zhengyou Zhang is currently serving as Director of AI Lab and Robotics X, Tencent. He is an ACM and IEEE Fellow. His research interests include computer vision, multimedia technology, motion analysis, robot navigation, immersive remote interaction along with other areas of contribution. He has published more than 250 papers in top international conferences and journals, which have been cited more than 50,500 times, and he holds nearly 200 issued patents. He received the IEEE Helmholtz Test of Time Award in 2013 for his "Zhang's camera calibration method".