2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Kinodynamic Motio
M

Yanra

2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-7281-6212-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/IROS45743.2020.9341572

Abstract— **This paper proposes a kin
ning framework for multi-legged robot
mixed-integer convex program (MICP
reasons about centroidal motion, conta
gait sequences. This method uniquely
space discretization and the construct
polytope (FWP) to encode kinematic co
friction cone constraint, and gait sequen
The MICP could be efficiently solved
by off-the-shelf numerical solvers and
jumping motions without requiring ini
and experimental results demonstrate th
could find novel and dexterous mane
deployable on the two-legged robot platf
challenging terrains.**

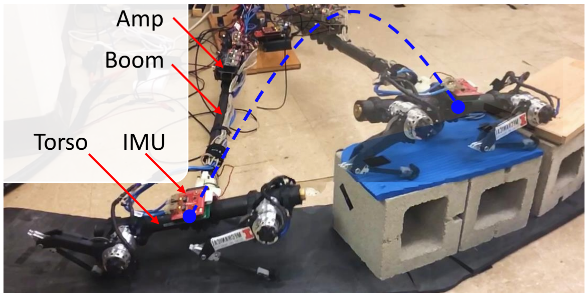


Fig. 1. The two-legged planar robot executing a dynamic jump to mount an obstacle 80% of its height.

I. Introductic

The ability to perform dynamic m
over gaps and jumping on high p
advantage of legged systems. Coordinating multiple limbs to execute dynamic motions is a challenging problem since it involves both continuous and discrete variables. This problem requires decision making in a semi-continuous search space, which involves continuous variables describing robot state, contact positions, and contact wrenches; It also involves discrete variables such as the gait sequence.

Many methods have been developed to solve this problem. For example, the trajectory optimization (TO) approach locally improves upon an initial motion plan by solving a general nonlinear optimization problem using a gradient-based nonlinear solver. There has been tremendous progress in using TO to solve locomotion problems. MIT Cheetah 2 robot could jump over obstacles by solving nonlinear constrained optimization online [1]. Optimized jumping trajectories are generated offline [2] and implemented on MIT Cheetah 3 [3]. Linear complementary problems (LCP) are formulated in [4] to generate trajectories without *a priori* contact scheduling. Dynamic movements without scheduled contact are also generated in [5] using a hierarchical framework. The combined planning problem is solved in [6] by incorporating all constraints into the objective function, and solve unconstrained nonlinear programming (NLP). Legged locomotions with gait sequences are generated on non-flat terrain in [7] in a single TO formulation using a phase-based parameterization method. These methods either rely on explicit contact schedules or require solving a large NLP. The size and non-convexity of these problems imply that the nonlinear solver is only effective searching for a local minima around the initial guess. Hence, proper initialization of the optimization is crucial in finding a feasible solution. Besides, the infeasible status returned by a local NLP solver is not informative since one is not sure whether the planning problem itself is infeasible or it is not initialized properly.

Mixed-integer convex optimization does not rely on the initial seed and warrants global solution [8]. With the recent advancement in numerical solvers, a medium-sized mixed-integer convex programming (MICP) can be solved efficiently by off-the-shelf solvers such as gurobi [9], Mosek [10] and CPLEX [11]. Due to its feature that warrants a global solution with either global optimality or infeasibility certificate, MICP has found many applications in robotics. For example, it has been used in global inverse kinematics [12], grasping [13], footstep planning [14], quadruped locomotion planning [15], and aggressive legged locomotion [16], [17].

This paper presents a novel MICP-based kinodynamic motion planning framework for aggressive jumping motions while considering joint torque constraints. This method simultaneously plans centroidal motion, contact location, contact wrench, and gait sequence for a planner two-legged robot. To address the non-convex torque constraints due to the trigonometric terms in the Jacobian matrix, we adopt the notion of feasible wrench polytope (FWP) [18]. The joint torque constraint is approximated as a polytope containment problem over disjoint convex sets from the discretization of the configuration space (C-space). Similarly, bilinear terms are approximated using the McCormick Envelope [19] relax-

This project is supported in part by NAVER LABS Corp. under grant 087387, Air Force Office of Scientific Research under grant FA2386-17-1-4665, and National Science Foundation under grant 1752262.

Yanran Ding and Chuanzheng Li are with the [1] Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, 61820 USA. (email:{yding35,cli67}@illinois.edu)

Hae-Won Park is with the [2] Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology, Daejeon-34141, South Korea. (email:haewonpark@kaist.ac.kr)

ation, hence resulting in a mixed-integer convex formulation. Our method reduces the number of decision variables by setting up a preferred ordering on the robot stance modes. As a result, any robot-state-trajectory in the C-space implies a unique gait sequence. The proposed meth
a hardware platform, as shown in Fig. 1. T
guides the robot to jump on a platform is
providing initial guesses to the MICP, whi
for local optimization based trajectory opti

The paper is organized as follows: Sec
the single rigid body dynamics model, mix
constraint, and other constraints including
tinuity and contact location choice. Sectic
simulation and experiment results of the
motion planning. Section IV discusses the
work and Section V provides the concludi

II. Technical Approac

In this section, we briefly introduce t
model before the construction of configu
cretization and feasible wrench polytopes
our formulation. And we also formalize
formulation with assumptions and constrair
and foothold position choice.

A. Dynamic Model

A simple model captures the major dynamical effect and reduce the number of optimization variables. As shown in Fig. 1, the robot legs are made of light-weight material such as 3D printed parts and carbon fiber tubes, which results in the leg mass being less than 10% of the total mass. Hence, we make the following assumption to simplify the dynamic model.

***Assumption** 1 (Light Legs):* The leg mass is negligible.

Based on assumption 1, we employ the Centroidal Dynamics (CD) model [20], specifically, the 2D single rigid body model which only considers the torso dynamics. As shown in Fig. 2, the configuration of the robot could be represented by the Special Euclidean Group $SE(2)$ parameterized by $\boldsymbol{q} = [x, z, \theta]^T$, where $[x, z]^T$ is the location of the center of mass (CoM) and θ is the pitch angle. The input to the system is the spatial wrench $\boldsymbol{\mathcal{F}}_s$, and the dynamic model of the robot is

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{x} \\ \ddot{z} \\ \ddot{\theta} \end{bmatrix} = \boldsymbol{D}^{-1}\boldsymbol{\mathcal{F}}_s + \boldsymbol{a}_g, \tag{1}$$

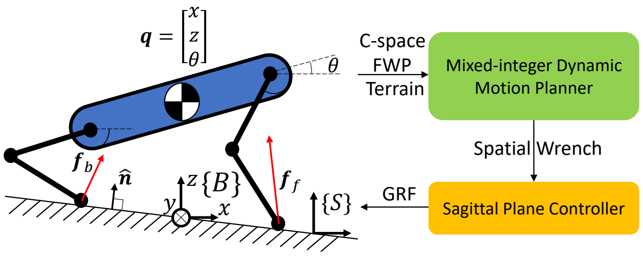
where $\boldsymbol{D} = \text{diag}(m, m, I_\theta)$ is the inertia tensor; diag$(\cdot)$ creates diagonal matrices; m is the total mass and I_θ is the moment of inertia around the CoM along the z-axis; $\boldsymbol{a}_g = [0, -g, 0]^T$ is the gravitational acceleration vector. Assuming there are N_c contact points, each with a ground reaction force (GRF) $\boldsymbol{f}_i \in \mathbb{R}^2$, the spatial wrench is given by $\boldsymbol{\mathcal{F}}_s$

$$\boldsymbol{\mathcal{F}}_s = \sum_{i}^{N_c} \begin{bmatrix} \boldsymbol{f}_i \\ \tau_i^y \end{bmatrix}, \tag{2}$$

TABLE I

Physical Parameter of the planar robot

Parameter	Unit	Value



$\boldsymbol{q} = \begin{bmatrix} x \\ z \\ \theta \end{bmatrix}$
θ
C-space
FWP
Terrain
Mixed-integer Dynamic Motion Planner
Spatial Wrench
$\boldsymbol{f}_b$
$\hat{\boldsymbol{n}}$
z {B}
y
x
$\boldsymbol{f}_f$
{S}
GRF
Sagittal Plane Controller

Fig. 2. Overview of the kinodynamic motion planning framework, which computes the motion and contact in a single MICP. The coordinate definition for $\{S\}$ (world frame) and $\{B\}$ (local frame) could be found on the left panel.

where τ_i^y is the moment generated by $\boldsymbol{f}_i$, $\tau_i^y = \boldsymbol{r}_i \wedge \boldsymbol{f}_i$; $\boldsymbol{r}_i$ is the vector from CoM to the i^{th} contact point; $\wedge : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ is the wedge product for two 2-dimensional vectors. The positive direction of the y-axis is pointing into the paper.

Fig. 2 shows a schematics of the robot with frame definitions. The origin of the body-fixed frame $\{B\}$ is located at the middle point of the foot contact points. The axes of $\{B\}$ are aligned with that of the world frame $\{S\}$. The configuration of the robot is represented using two variables. The local configuration $\boldsymbol{q}$ is expressed in frame $\{B\}$; the global variable is defined in frame $\{S\}$, named the touchdown state $\boldsymbol{q}^{TD} = [x^{TD}, z^{TD}, \theta^{TD}]^T \in \mathbb{R}^3$, where $[x^{TD}, z^{TD}]^T$ is the origin of frame $\{B\}$ and θ^{TD} is equal to the slope of the current terrain. This dichotomy of global and local states is convenient for imposing constraints of stance phase on the local state and choosing contact location using the global state. It could be observed from (1) that the translational and rotational dynamics are linear in terms of the spatial wrench $\boldsymbol{\mathcal{F}}_s$. Let the spatial wrench trajectory be parametrized by the Bézier polynomial with coefficient $\boldsymbol{\alpha}_{\mathcal{F}} = [\boldsymbol{\alpha}_{f_x}, \boldsymbol{\alpha}_{f_z}, \boldsymbol{\alpha}_{\tau_y}]^T$, then the trajectories $\dot{\boldsymbol{q}}(t), \boldsymbol{q}(t)$ are also parametrized by Bézier polynomials with coefficients $\boldsymbol{\alpha}_{\dot{q}}$ and $\boldsymbol{\alpha}_q$, respectively. Given initial conditions $\dot{\boldsymbol{q}}_0, \boldsymbol{q}_0$, coefficients $\boldsymbol{\alpha}_{\dot{q}}$ and $\boldsymbol{\alpha}_q$ could be obtained by linear operations.

$$\boldsymbol{\alpha}_{\dot{q}} = \mathcal{L}(\boldsymbol{\alpha}_{\mathcal{F}}, \dot{\boldsymbol{q}}_0), \;\; \boldsymbol{\alpha}_q = \mathcal{L}(\boldsymbol{\alpha}_{\dot{q}}, \boldsymbol{q}_0) \tag{3}$$

where the linear operation $\mathcal{L}(\cdot)$ is defined in Appendix I.

Although the dynamics in (1) are linear in terms of the spatial wrench, the coupling between forces $\boldsymbol{f}_i$ and the moment created by the force τ_i^y imposes bilinear constraints on the feasible wrench. Section II-C presents how these constraints are represented using the feasible wrench polytope (FWP).

The state and control trajectories during stance a
mials parametrized by the Bézier coefficients $\boldsymbol{\alpha}_{\mathcal{F}}$
trajectory during aerial phase could be parametri
second-order polynomial. To make the kinodynan
planning problem finite dimensional, the contin
trajectory $\boldsymbol{\mathcal{F}}_s(t), \dot{\boldsymbol{q}}(t), \boldsymbol{q}(t)$ are sampled at a time
$\{t_k|k = 1, 2, \cdots, N_t\}$, where N_t is the numbe
during stance phase.

B. C-space Discretization

The configuration space (C-space) $\boldsymbol{\Omega} \subset \mathbb{R}^3$ of
is the set of configurations that the robot could re
stance phase without violating kinematic constrain
of constraints that define the C-space is

$$\boldsymbol{\Omega} := \{\boldsymbol{q} \in \mathbb{R}^3 | 0 \leq \hat{\boldsymbol{n}}^T \cdot (\boldsymbol{p}_i^h - \boldsymbol{p}_i^c) \tag{4a}$$

$$0 \leq \hat{\boldsymbol{n}}^T \cdot (\boldsymbol{p}_i^k - \boldsymbol{p}_i^c) \tag{4b}$$

$$r_{min} \leq |\boldsymbol{p}_i^h - \boldsymbol{p}_i^c|_2 \leq r_{max} \tag{4c}$$

$$\boldsymbol{q}_{min} \leq \boldsymbol{q} \leq \boldsymbol{q}_{max}\}, \tag{4d}$$

where $\hat{\boldsymbol{n}}$ is the normal vector of the terrain; $\boldsymbol{p}^h, \boldsymbol{p}^k$ are the hip and knee joint positions of the leg $i \in \{\boldsymbol{back}, \boldsymbol{front}\}$ and $\boldsymbol{p}^c$ is the foot contact position. (4a) and (4b) prohibit the hip and knee from penetrating the terrain surface; (4c) sets boundaries r_{min}, r_{max} on the leg extension, using two-norm $|\cdot|_2$; (4d) is the box constraint on $\boldsymbol{q}$. The C-space is calculated based on the following assumption.

***Assumption** 2 (Constant Stance Width):* The distance between two contact feet is equal to the body length L.

This assumption removes the dependency on the stance width, simplifying the C-space construction. Since all quantities in (4) could be retrieved through kinematic calculation given the robot configuration $\boldsymbol{q}$, the C-space could be clearly defined once the robot parameters are given. Fig. 3 shows the C-space of the robot with parameters in Table I.

During the stance phase, the robot could be in one of three stance modes, namely, front, double and back stance. When the robot could take double stance, it could also take either front or back stance by lifting one of its legs, complicating the contact scenario. To simplify the choice of stance mode, the following assumption is made.

***Assumption** 3 (Preference on Double Stance):* The robot prefers double stance to single stance, and would be in double stance whenever possible.

Based on the observation that double stance provides more control authority compared with single stance, assumption 3 establishes a one-to-one mapping between robot configuration $\boldsymbol{q}$ and stance mode. As shown in Fig. 3, the C-space is divided into three disjoint regions corresponding to front stance $\boldsymbol{\Omega}_{fs}$ (blue), double stance $\boldsymbol{\Omega}_{ds}$ (black), and back stance $\boldsymbol{\Omega}_{bs}$ (red), or equivalently,

$$\boldsymbol{\Omega} := \boldsymbol{\Omega}_{bs} \cup \boldsymbol{\Omega}_{fs} \cup \boldsymbol{\Omega}_{ds}. \tag{5}$$

An illustration corresponding to each stance mode is visualized on the right panel of Fig. 3. Due to the one-to-one mapping between robot configuration $\boldsymbol{q}$ and stance mode, the gait sequence planning is encoded into the C-space construction. Hence, a state trajectory $\boldsymbol{q}(t)$ in $\boldsymbol{\Omega}$ also contains the gait sequence information.

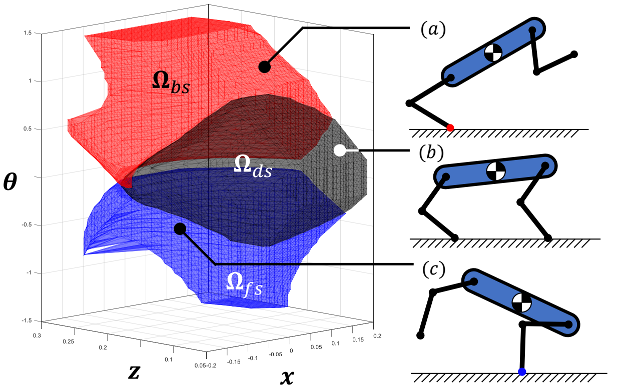


Fig. 3. (left) The 3-dimensional (3D) configuration space of the robot; (right) The illustration of robot configuration in each stance mode (a) back stance (b) double stance (c) front stance.

As shown in Fig. 3, the C-space is a non-convex set. To tackle this problem, the C-space $\boldsymbol{\Omega}$ is discretized into $N_d \in \mathbb{Z}^+$ convex polytopic cells, denoted by $\boldsymbol{c}_i \subset \boldsymbol{\Omega}, i \in \{1, \cdots, N_d\}$, where N_d is the total number of cells. The union of the cells is contained within the C-space, namely, $\cup_{i=1}^{N_d} \boldsymbol{c}_i \subset \boldsymbol{\Omega}$. For simplicity, tetrahedrons are used to discretize the C-space, whose distribution is designed such that each cell resides within the same stance region. The geometry of each cell is encoded by linear inequalities

$$\boldsymbol{A}_i^{geo} \cdot \boldsymbol{q} \leq \boldsymbol{b}_i^{geo}, i = 1, \cdots, N_d, \tag{6}$$

where the matrices $\boldsymbol{A}_i^{geo}$ and $\boldsymbol{b}_i^{geo}$ delineate the cell $\boldsymbol{c}_i$ using the half-plane representation ($\mathcal{H}$-Rep).

C. Feasible Wrench Polytope

This section presents a formulation of the feasible wrench polytope (FWP) that is pertinent to the robot system studied in this work. For a more comprehensive derivation of the FWP, please refer to [18].

*1) **FWP of One Leg**:* First we define the feasible force polytope (FFP) for one leg as

$$FFP := \{\boldsymbol{f} \in \mathbb{R}^2 \mid |\boldsymbol{J}^T \boldsymbol{f}|_\infty \leq \tau_{max} \tag{7a}$$

$$|\boldsymbol{f} - \hat{\boldsymbol{n}}^T \boldsymbol{f}|_\infty \leq \mu \hat{\boldsymbol{n}}^T \boldsymbol{f}\}, \tag{7b}$$

where $\boldsymbol{J}$ is the Jacobian matrix; τ_{max} is the joint torque limit; μ is the coefficient of friction. Based on the assumption 1, the GRF is mapped to joint torque via $\boldsymbol{\tau} = \boldsymbol{J}^T \boldsymbol{f}$. Furthermore, assumption 2 implies that given the robot configuration $\boldsymbol{q}$, joint angles could be calculated through inverse kinematics. The inequality (7a) encodes the joint torque constraint $|\boldsymbol{\tau}|_\infty < \tau_{max}$, where $|\cdot|_\infty$ is the infinity norm. The inequality (7b) represents the friction cone constraint.

The FWP of one leg is the set of spatial wrenches that could be provided by the FFP

$$FWP_i := \{\boldsymbol{\mathcal{F}}_i \in \mathbb{R}^3 \mid \boldsymbol{\mathcal{F}}_i = \begin{bmatrix} \boldsymbol{f}_i^k \\ \boldsymbol{r}_i \wedge \boldsymbol{f}_i^k \end{bmatrix}, \boldsymbol{f}_i^k \in FFP_i\}, \tag{8}$$

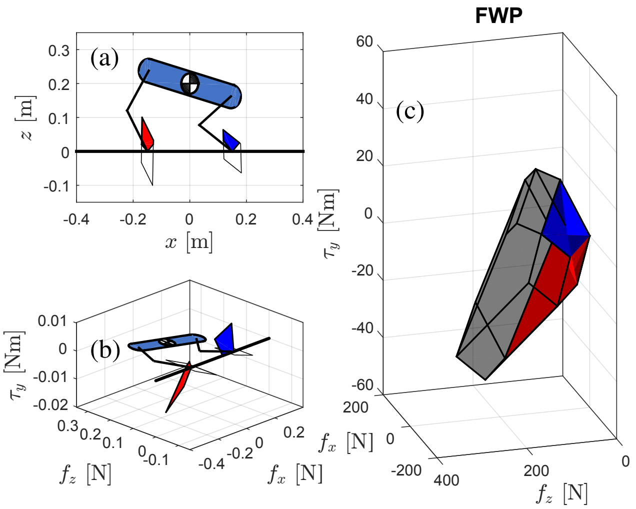


Fig. 4. The FWP of the two-legged robot. (a) The FFP of front (blue) and back (red) legs. (b) The FWP of both legs plotted in the 3D wrench space. (c) The FWP of the robot is the Minkowski sum of the FWP of both legs.

where $\boldsymbol{f}_i^k$ is the k^{th} vertex of FFP for contact point i. Note that the FWP is defined using the vertex-representation of a polytope ($\mathcal{V}$-Rep). Fig. 4(b) shows an example of FWPs of front leg (blue) and back leg (red), which are 2-D polytopes embedded in the 3-D wrench space.

2) ***FWP of Two Legs:*** The FWP when the two-legged robot is in double stance is defined as the Minkowsi sum [21] of the FWP created by both contact legs,

$$FWP_q = \bigoplus_{i=1}^{2} FWP_i, \tag{9}$$

where FWP_q indicates the FWP of the robot at configuration $\boldsymbol{q}$. The Minkowski sum of two sets X and Y is $X \oplus Y := \{x + y | x \in X, y \in Y\}$. Fig. 4(c) shows an example of FWP_q as the Minkowski sum of the FWP of both legs. Assumption 3 implies that when $\boldsymbol{q} \in \boldsymbol{\Omega}_{ds}$, the corresponding FWP_q is defined as in (9).

For a cell $\boldsymbol{c}_i$ in the C-space discretization, its representative FWP is defined as

$$FWP_{c_i} = \begin{cases} \cap_{k=1}^{4} FWP_{c_i}^{k}, & \text{double stance} \\ FWP_{c_i}^{cbsv}, & \text{single stance,} \end{cases} \tag{10}$$

where $FWP_{c_i}^k$ is the FWP of the k^{th} node of the cell $\boldsymbol{c}_i$. $FWP_{c_i}^{cbsv}$ is the FWP at the Chebyshev center [8], which is the center of the largest Euclidean ball that lies in a polytope. The choice of FWP for double stance in (10) is a conservative approximation, since the wrench in FWP_{c_i} could be achieved at all 4 vertices in the tetrahedron of the cell $\boldsymbol{c}_i$. This formulation provides robustness when the robot is in the double stance since it is an inner approximation. In comparison, for a cell in the single stance, the FWP at each vertex degenerates to a 2-D polytope due to the coupling between forces and moment. Since each node corresponds to a different $\boldsymbol{r}_i$, the FWPs have no intersection except at the origin. Therefore, the FWP for cells in the single stance is defined at the Chebyshev center of the cell.

The FWP_{c_i} could be represented using the half-plane representation ($\mathcal{H}$-Rep) consisting of a set of linear constraints

$$\boldsymbol{A}_i^{fwp} \cdot \boldsymbol{\mathcal{F}}_s \leq \boldsymbol{b}_i^{fwp}, i = 1, \cdots, N_d, \tag{11}$$

where $\boldsymbol{A}_i^{fwp}$ and $\boldsymbol{b}_i^{fwp}$ are the matrices that describe half-planes. The single stance FWP is subject to equality constraints $\boldsymbol{A}_{i,e}^{fwp} \cdot \boldsymbol{\mathcal{F}}_s = \boldsymbol{b}_{i,e}^{fwp}$, which could also be incorporated into the form of inequality constraint as in (11).

Note that for a given physical parameter of the robot and a C-space discretization, FWP_{c_i} only needs to be computed once.

D. Mixed-integer Wrench Constraint

The nonlinear and non-convex wrench constraint is imposed in a piecewise constant fashion over the discretized C-space, enabling a mixed-integer convex formulation. A binary matrix $\boldsymbol{B}^{cs} \in \{0, 1\}^{N_t \times N_d}$ is constructed such that $\boldsymbol{B}_{i,j}^{cs} = 1$ indicates that $\boldsymbol{q}(t_i)$ is within cell $\boldsymbol{c}_j$ and the spatial wrench should be chosen within FWP_{c_i}

$$\begin{aligned} \boldsymbol{B}_{i,j}^{cs} \implies & \boldsymbol{A}_j^{geo} \cdot \boldsymbol{q}(t_i) \leq \boldsymbol{b}_j^{geo} \\ \implies & \boldsymbol{A}_j^{fwp} \cdot \boldsymbol{\mathcal{F}}_s(t_i) \leq \boldsymbol{b}_j^{fwp}, \end{aligned} \tag{12}$$

where the *implies* operator ($\implies$) in (12) is implemented using the big-M formulation [22]. Additional constraints are imposed for physical feasibility

$$\sum_{j=1}^{N_d} \boldsymbol{B}_{i,j}^{cs} = 1, \; \forall i = 1, \cdots, N_t \tag{13}$$

which requires that at each time step t_i, the robot state $\boldsymbol{q}(t_i)$ can only reside within one cell $\boldsymbol{c}_j$.

E. Aerial Phase Constraints

The objective of the kinodynamic motion planner is to reach the goal region through a series of jumping motions, which involves both stance and aerial phases. During the aerial phase, the robot is airborne and only subject to gravity, whose trajectory is described by

$$\begin{bmatrix} \bar{\boldsymbol{q}} \\ \dot{\boldsymbol{q}} \end{bmatrix}_{i+1}^{TD} = \begin{bmatrix} \bar{\boldsymbol{q}} \\ \dot{\boldsymbol{q}} \end{bmatrix}_{i}^{TO} + \begin{bmatrix} \dot{\boldsymbol{q}}_i^{TO} \\ \boldsymbol{a}_g \end{bmatrix} T_{i,air} + \begin{bmatrix} \frac{1}{2}\boldsymbol{a}_g \\ \boldsymbol{0} \end{bmatrix} T_{i,air}^2 \tag{14}$$

where the superscripts $(\cdot)^{TO}$ and $(\cdot)^{TD}$ indicate variables at take-off and touch-down, respectively; $T_{i,air}$ is the aerial time of the i^{th} jump; where $i = 1, \cdots N_{jp}$ and N_{jp} is the number of jumps. The state $\bar{\boldsymbol{q}}_i^{TD}$ is the sum of the global state $\boldsymbol{q}_i^{TD}$ and the local state at the start of the next stance phase $\boldsymbol{q}_i^{start}$. Similarly, $\bar{\boldsymbol{q}}_i^{TO} = \boldsymbol{q}_i^{TD} + \boldsymbol{q}_i^{end}$, where $\boldsymbol{q}_i^{end}$ is the end state of the i^{th} stance phase. This dichotomy of global and local state is convenient for imposing wrench constraint (12) on the local state and choosing contact location using the global state.

Although the stance time T_{st} of each jump is set to a constant to simplify the problem, the aerial time T_{air} is an optimization variable. Hence, the bilinear terms $\dot{\boldsymbol{q}}_i^{TO} T_{i,air}$ in

(14) leads to non-convex constraints. To this end, methods such as using the McCormick Envelope [19] to approximate the bilinear terms have been applied on planning aggressive motions of legged robots [16]. McCormick Envelope is used in our formulation because it could provide a close approximation with a relatively small number of binary variables since the range of each quantity could be empirically bounded from simulation and experiment. Using a similar technique, the quadratic term $T_{i,air}^2$ in (14) is approximated by a piecewise affine function.

F. Foothold Position

With assumption 2, the foothold position choice is simplified to finding $\boldsymbol{q}^{TD}$ on the given terrain. To simplify terrain geometry, the following assumption is made.

***Assumption** 4 (Terrain):* The terrain consists of segments which are modeled by piecewise affine functions.

A binary matrix $\boldsymbol{B}^{fp} \in \{0,1\}^{N_s \times N_j}$ is constructed to assign foothold positions, where N_s is the total number of terrain segments. $\boldsymbol{B}_{i,j}^{fp} = 1$ implies that at the j^{th} jump, the $\boldsymbol{q}_j^{TD}$ lies on the i^{th} terrain segment $\boldsymbol{seg}_i$

$$\boldsymbol{B}_{i,j}^{fp} = 1 \implies \boldsymbol{q}_j^{TD} \in \boldsymbol{Q}_i \tag{15a}$$

$$\sum_{i=1}^{N_s} \boldsymbol{B}_{i,j}^{fp} = 1,\ \forall j = 1, \cdots, N_{jp}, \tag{15b}$$

where $\boldsymbol{Q}_i$ refers to the i^{th} terrain segment. At the end of the last jump, the robot should reach the goal region

$$\boldsymbol{q}_{N_{jp}}^{TD} \in \boldsymbol{Q}_{goal} \tag{16}$$

where $\boldsymbol{q}_{N_{jp}}^{TD}$ is the touchdown state after the final jump and $\boldsymbol{Q}_{goal}$ is the goal region.

G. The Mixed-integer Convex Program

The kinodynamic motion planning problem of a planar two-legged robot could be transcribed to a MICP. The decision variable vector for this particular problem is $\boldsymbol{x}_{opt} = [\boldsymbol{\alpha}_{\mathcal{F}}, \boldsymbol{q}_0, \dot{\boldsymbol{q}}_0, \boldsymbol{q}^{TD}, \boldsymbol{T}_{air}, \boldsymbol{B}^{cs}, \boldsymbol{B}^{fp}]^T$. The complete formulation of the MICP is:

$$\begin{aligned}
\min_{\boldsymbol{x}_{opt}}.\ & f(\boldsymbol{x}_{opt}) && \text{(17a)}\\
\text{s.t. } & \boldsymbol{q}(t_i) \in \boldsymbol{c}_k \subset \boldsymbol{\Omega} && \text{(17b)}\\
& \ddot{\boldsymbol{q}}(t_i) = \boldsymbol{D}^{-1}\boldsymbol{\mathcal{F}}_s(t_i) + \boldsymbol{a}_g && \text{(17c)}\\
& \boldsymbol{\mathcal{F}}_s(t_i) \in FWP_{c_k} && \text{(17d)}\\
& \boldsymbol{q}_j^{TD} \in \boldsymbol{Q}_l && \text{(17e)}\\
& \boldsymbol{q}_0 \in \boldsymbol{Q}_{init},\ \boldsymbol{q}_{N_{jp}}^{TD} \in \boldsymbol{Q}_{goal} && \text{(17f)}\\
& \text{aerial phase constraint: } (14) && \text{(17g)}\\
& i = 1, \cdots, N_t; j = 1, \cdots, N_{jp} && \text{(17h)}\\
& k = 1, \cdots, N_d; l = 1, \cdots, N_s && \text{(17i)}
\end{aligned}$$

where (17b) is the kinematic constraint baked into the C-space; (17c) is the dynamic constraint and (17d) is the wrench constraint. (17e) constrains the touchdown state to be on the terrain; (17f) are the boundary condition constraints, where $\boldsymbol{Q}_{init}$ is the feasible set for the initial condition. $f(\boldsymbol{x}_{opt})$ is a task-specific convex objective function. For example, $f(\cdot)$ could be the deviation from the goal $|\boldsymbol{q}_{N_{jp}}^{TD} - \boldsymbol{q}_{goal}|_2$, which makes the problem a mixed-integer quadratic program (MIQP); or it could be $-x_{N_{jp}}^{TD}$ to maximize the horizontal jumping distance, which leads to a mixed-integer linear program (MILP); $f(\cdot)$ could also be set as a constant value to solve a feasibility problem.

The MICP problem is formulated in MATLAB using YALMIP [23]. The computational geometry calculation related to FWP is done using the Multi-Parametric Toolbox 3 (MPT3) [24]. The MICP is solved by the solver Gurobi [9]. All of the computation is performed on a desktop with 2.9 GHz Intel i7.

III. Results

To validate the proposed kinodynamic motion planning algorithm, jumping experiments are conducted on the robot. Experiment results for both jumping forward and backward, together with the simulation result of a dynamic Parkour motion are presented. Note that the trajectories of all three motions are solved by the proposed MICP without any initial guesses.

A. Experimental Setup

As shown in Fig. 1, the planner two-legged robot similar to the one used in [25] is composed of the torso made of a carbon fiber tube and two legs modules, which enable dynamic maneuvers that demands high joint torques [26]. Each joint of the robot is equipped with a RLS-RMB20 magnetic encoder, and an inertial measurement unit (IMU) is mounted for state estimation, which is the same as that in [27]. The robot is mounted on the end of a passive boom system with a radius $R_{boom} = 1.25$ m. Two encoders are installed at the base of the boom to measure the global positioning of the robot, and another encoder is mounted at the connection between the tip of the boom and the robot to measure the pitch angle θ. The robot is externally powered and the Elmo Gold Twitter amplifiers are mounted on the boom. The control loop runs at 4kHz on an Intel i5 desktop in Simulink Real-Time. The physical parameters of the robot could be found in Table I.

B. Sagittal Plane Control

The spatial wrench trajectory obtained from solving the MICP is distributed to the GRF in the sagittal plane using the closed-chain-constrained operational-space control [28]. Similar to the frontal plane controller in [1], a linear operator $\boldsymbol{\Pi} \in \mathbb{R}^{4\times3}$ is calculated to map the spatial wrench $\boldsymbol{\mathcal{F}}_s$ to the joint torque $\boldsymbol{u}_{sag} = \boldsymbol{\Pi}\ \boldsymbol{\mathcal{F}}_s$. Let $\boldsymbol{q}_{sag} := [\boldsymbol{q}^T, \boldsymbol{q}_l^T]^T \in \mathbb{R}^{7\times1}$ be the generalized coordinates of the sagittal dynamics, where $\boldsymbol{q}_l$ is the vector of joint angle. The no-slip ground contact constraints are

$$\begin{aligned}
\dot{\boldsymbol{p}}_{foot} &= \boldsymbol{J}_{foot}\dot{\boldsymbol{q}}_{sag} = 0\\
\ddot{\boldsymbol{p}}_{foot} &= \boldsymbol{J}_{foot}\ddot{\boldsymbol{q}}_{sag} + \dot{\boldsymbol{J}}_{foot}\dot{\boldsymbol{q}}_{sag} = 0,
\end{aligned} \tag{18}$$

where $\boldsymbol{p}_{foot}$ is the foot position, and the foot Jacobian $\boldsymbol{J}_{foot} \in \mathbb{R}^{4\times7}$ is partitioned as $\boldsymbol{J}_{foot} = [\boldsymbol{J}_B, \boldsymbol{J}_l]$. The

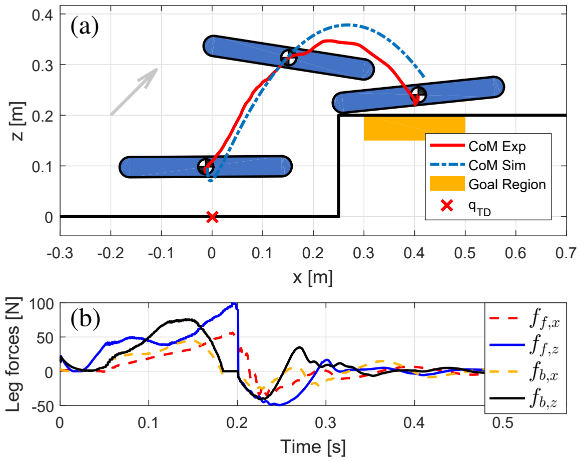


Fig. 5. The result of the experiment where the robot jumped forward on a platform. (a) CoM trajectories of simulation (blue) and experiment (red); the orange area is the goal region; the gray arrow indicates the jumping direction. (b) The leg forces during the stance phase $t \leq 0.2$ s and the aerial phase.

dynamics of the sagittal plane system are

$$\begin{bmatrix} \boldsymbol{D} & 0 & -\boldsymbol{J}_B^T \\ 0 & 0 & -\boldsymbol{J}_l^T \\ \boldsymbol{J}_B & \boldsymbol{J}_l & 0 \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \ddot{\boldsymbol{q}}_l \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{u}_{sag} \\ -\dot{\boldsymbol{J}}_{foot}\dot{\boldsymbol{q}}_{sag} \end{bmatrix} \qquad (19)$$

where $\boldsymbol{\lambda}$ is the Lagrange multiplier associated with the constraints in (18). By solving the system of equations one could derive $-\boldsymbol{J}_B^T \boldsymbol{J}_l^{-T} \boldsymbol{u}_{sag} = \boldsymbol{D}\ddot{\boldsymbol{q}} = \boldsymbol{\mathcal{F}}_s$. Thus, as one of many possible solutions, $\boldsymbol{\Pi}$ is selected as $\boldsymbol{\Pi} = -(\boldsymbol{J}_B \boldsymbol{J}_l^{-T})^{\dagger}$, where $(\cdot)^{\dagger}$ provides the Moore-Penrose pseudoinverse.

Using the saggittal plan control, the feedforward wrench trajectory $\boldsymbol{\mathcal{F}}_{ff}$ from MICP and a proportional-derivative (PD) feedback controller is used to track the desired state trajectory. $\boldsymbol{\mathcal{F}}_{fb} = \boldsymbol{K}_p(\boldsymbol{q}_d - \boldsymbol{q}) + \boldsymbol{K}_d(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}})$, where $\boldsymbol{K}_p, \boldsymbol{K}_d$ are diagonal gain matrices; $\boldsymbol{q}_d$ and $\dot{\boldsymbol{q}}_d$ are the desired state trajectories. During the swing phase, a workspace PD controller is applied on the swing foot to track the prescribed swing trajectory.

C. Jump On Platforms

As shown in Fig. 1, the robot jumps forward on a 0.2 m high platform. Both simulation and experimental results are shown in Fig. 5, where the CoM trajectory from the simulation is shown in blue and that from the experiment is shown in red. The leg forces are shown in Fig. 5 (b) during the stance phase ($t \leq 0.2$ s) and the aerial phase. Another experiment where the robot jumps back onto the platform is shown in Fig. 6, where the sequential snapshots show that the solution involves using large body pitch oscillation to aid the robot to jump on the platform.

For these less dynamic jumping motions, only double stance region $\boldsymbol{\Omega}_{ds}$ is used. The number of variables for both motions is 216 (27 continuous, 189 integer), and the computational time to solve the MICP is 0.84 s for the jumping forward problem and 5.94 s for the jumping backward problem. The solve time difference may be explained by that the knee-bending-back configuration provides more forward force authority. Additionally, the knee-bending-forward configuration imposes stricter collision avoidance constraints between the knee and terrain. The Bézier coefficients of the wrench trajectories are summarized in Table II.

Experiment	Wrench	Bézier Coefficients
Jump forward	$\boldsymbol{\alpha}_{f_x}$	0.0, -7.6, 33.8, -33.7, 90.1, 0.0
	$\boldsymbol{\alpha}_{f_z}$	25.1, -69.0, 152.0, -50.0, 262.7, 0.0
	$\boldsymbol{\alpha}_{\tau_y}$	0.0, 5.5, -25.3, 25.5, -5.8, 0.0
Jump backward	$\boldsymbol{\alpha}_{f_x}$	0.0, 198.4, -404.8, 296.7, -135.5, 0.0
	$\boldsymbol{\alpha}_{f_z}$	25.1, -171.9, 631.8, -786.8, 623.5, 0.0
	$\boldsymbol{\alpha}_{\tau_y}$	0.0, 44.5, -79.4, 48.2, -14.5, 0.0

TABLE II

BÉZIER COEFFICIENT FOR THE JUMPING ON PLATFORM EXPERIMENTS

D. Parkour Motion

The proposed kinodynamic motion planning framework could generate plans that traverse terrains that require complex maneuvers. For example, Fig. 7 shows one problem setup where the goal region is on the high platform, and the robot cannot reach it with a single jump due to actuation limitations. As shown in Fig. 7, the MICP provides the solution where the left platform is used as a stepping stone towards the goal region. By making two jumps, the proposed framework solves the problem without initial guess nor user input about the step planning. With the grid resolution $N_{bs} = 10, N_{fs} = 10, N_{ds} = 21$, the MICP involves 485 variables (53 continuous, 432 integer), and the computational time is 27 s. The algorith m utilized double stance and back stance. The back stance is used towards the end of the second jump, presumably to take advantage of the extra kinematic reachability in body pitch.

This simulation result showcases one of the advantages of mixed-integer program based motion planning algorithms, which is that it could reason about making discrete decisions.

IV. Discussion

In this section, we discuss some of the findings and limitations of this work. This work has extended the single-leg model in [17] to the multi-legged model, which poses unique challenges since different stance modes dictate whether the system is over-actuated or under-actuated.

Compared with [15], we focus on generating dynamic jumping motions that are not limited to the vicinity of the nominal pose. This work is most similar to [16] in spirit, where dynamic legged motions are generated via MIQP. The construct of this work allows the joint torque constraint to be explicitly imposed using the notion of FWP. However, with the series of assumptions made intending to reduce the number of decision variables, the proposed method is only applicable to planner models. The stance time and number of jumps should also be chosen *a priori*.

In experiments, the sagittal plane control works well for the double stance. Nevertheless, for Parkour motion in Section III-D, the jump on the lower platform often

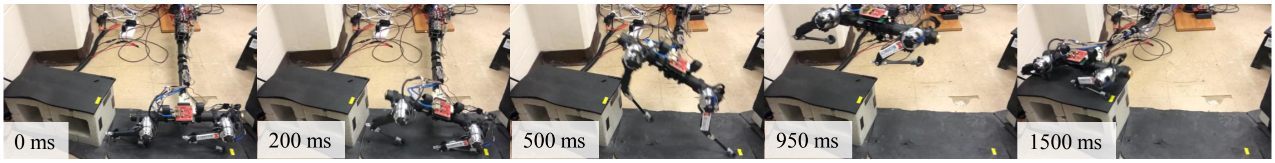


Fig. 6. Sequential snapshots of the experiment where the robot jumps backwards onto a 0.2 m high platform to reach the goal region.

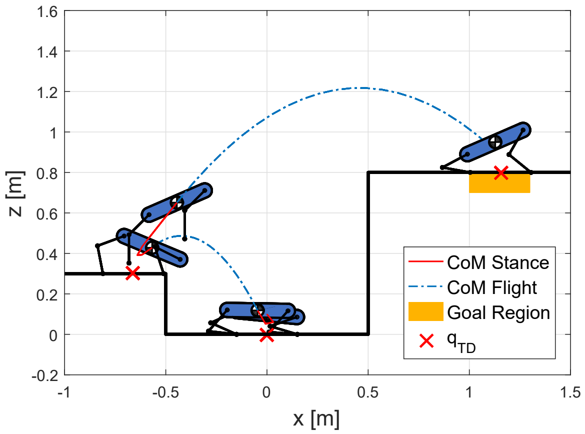


Fig. 7. Simulation result of the Parkour motion. The proposed formulation can find the strategy of utilizing the left platform as a stepping-stone to reach the goal region on the high platform.

involves undesirable single stance phase which the reactive sagittal plane control could not stabilize well. In hindsight, model-predictive control (MPC) may provide more robust performance since it could reason about the robot dynamics within the prediction horizon.

V. Conclusion and Future Work

This paper proposed a novel MICP-based kinodynamic motion planning framework for aggressive jumping motions on 2-D multi-legged robots. The proposed method could produce highly dynamic jumping motions by solving for the centroidal motion, contact location, contact wrench and gait sequence simultaneously in a single MICP, with a global optimality certificate. The MICP approximates the nonlinear and non-convex constraints into piecewise convex constraints, and a preferred ordering on the robot stance modes is introduced to encode the gait sequence into the robot state trajectory. Both simulation and experiment show that the proposed planner could generate dynamically feasible motions on complex terrain. In the future, we plan to combine the MICP planner with MPC for robust jumping motions in experiments. We also envision combining sampling-based methods to tackle more complex terrains.

VI. Acknowledgment

The authors would like to thank Prof. Kris Hauser, Prof. João Ramos and Dr. Zherong Pan for the helpful insightful discussions.

References

[1] H.-W. Park, P. M. Wensing, and S. Kim, “High-speed bounding with the MIT Cheetah 2: Control design and experiments,” *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 167–192, 2017.

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

[3] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, “MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 2245–2252.

[4] M. Posa, C. Cantu, and R. Tedrake, “A direct method for trajectory optimization of rigid bodies through contact,” *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[5] C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, “Hierarchical planning of dynamic movements without scheduled contact sequences,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4636–4641.

[6] I. Mordatch, E. Todorov, and Z. Popović, “Discovery of complex behaviors through contact-invariant optimization,” *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 1–8, 2012.

[7] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, “Gait and trajectory optimization for legged systems through phase-based end-effector parameterization,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, July 2018.

[8] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[9] Gurobi Optimization, “Gurobi optimizer reference manual,” 2020.

[10] MOSEK ApS, “The MOSEK optimization software,” 2014.

[11] IBM Corp, “User’s manual for CPLEX,” 2010.

[12] H. Dai, G. Izatt, and R. Tedrake, “Global inverse kinematics via mixed-integer convex optimization,” *The International Journal of Robotics Research*, vol. 38, no. 12-13, pp. 1420–1441, 2019.

[13] M. Liu, Z. Pan, K. Xu, and D. Manocha, “New formulation of mixed-integer conic programming for globally optimal grasp planning,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4663–4670, 2020.

[14] R. Deits and R. Tedrake, “Footstep planning on uneven terrain with mixed-integer convex optimization,” in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 279–286.

[15] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, “Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, July 2018.

[16] A. K. Valenzuela, “Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain,” Ph.D. dissertation, Massachusetts Institute of Technology, 2016.

[17] Y. Ding, C. Li, and H.-W. Park, “Single leg dynamic motion planning with mixed-integer convex optimization,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1–6.

[18] R. Orsolino, M. Focchi, C. Mastalli, H. Dai, D. G. Caldwell, and C. Semini, “Application of wrench-based feasibility analysis to the online trajectory optimization of legged robots,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3363–3370, Oct 2018.

[19] G. P. McCormick, “Computability of global solutions to factorable nonconvex programs: Part I-convex underestimating problems,” *Mathematical programming*, vol. 10, no. 1, pp. 147–175, 1976.

[20] D. E. Orin, A. Goswami, and S.-H. Lee, “Centroidal dynamics of a humanoid robot,” *Autonomous robots*, vol. 35, no. 2-3, pp. 161–176, 2013.

[21] G. Varadhan and D. Manocha, "Accurate Minkowski sum approximation of polyhedral models," in *12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings.* IEEE, 2004, pp. 392–401.

[22] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Control Conference (ECC), 2001 European*. IEEE, 2001, pp. 2603–2608.

[23] J. Löfberg, "YALMIP : A toolbox for modeling and optimization in matlab," in *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[24] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510.

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

[26] Y. Ding and H.-W. Park, "Design and experimental implementation of a quasi-direct-drive leg for optimized jumping," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2017, pp. 300–305.

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

[28] M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart, "Quadrupedal locomotion using hierarchical operational space control," *The International Journal of Robotics Research*, vol. 33, no. 8, pp. 1047–1062, 2014.

[29] Ç. Dişibüyük and H. Oruç, "A generalization of rational bernstein–bézier curves," *BIT Numerical Mathematics*, vol. 47, no. 2, pp. 313–323, 2007.

[30] E. H. Doha, A. H. Bhrawy, and M. A. Saker, "Integrals of Bernstein polynomials: An application for the solution of high even-order differential equations," *Applied Mathematics Letters*, vol. 24, no. 4, pp. 559–565, 2011.

Appendix I
Integration of Bézier Polynomial

A Bézier polynomial is a linear combination of a Bernstein polynomial basis [29], so the integration of a Bézier polynomial is a linear operation [30] on the Bézier coefficients. For example, the linear relationship between wrench Bézier coefficients and twist Bézier coefficients is

$$\frac{M+1}{T_{st}}\boldsymbol{\Phi}(M,T_{st})\boldsymbol{\alpha}_{\dot{q}} = [\boldsymbol{D}^{-1}\boldsymbol{\alpha}_{\mathcal{F}}^T + \boldsymbol{a}_g, \dot{\boldsymbol{q}}_0]^T, \tag{20}$$

where M is the order of Bézier polynomial; T_{st} is stance duration; $\dot{\boldsymbol{q}}_0 \in \mathbb{R}^3$ is the initial body twist; $\boldsymbol{\alpha}_{\dot{q}} \in \mathbb{R}^{(M+2)\times 3}$ is the Bézier coefficients for the spatial twist trajectory; $\boldsymbol{\Phi} \in \mathbb{R}^{(M+2)\times(M+2)}$ is a matrix whose elements are defined as

$$\boldsymbol{\Phi}_{i,j} := \begin{cases} -1, & j = i = 1,2,\cdots,M+1 \\ 1, & j = i+1 = 2,3,\cdots,M+2 \\ \frac{T_{st}}{M+1}, & i = M+2, j = 1 \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

The linear operation $\boldsymbol{\alpha}_{\dot{q}} = \mathcal{L}(\boldsymbol{\alpha}_{\mathcal{F}}, \dot{\boldsymbol{q}}_0)$ is obtained by inverting the matrix in front of $\boldsymbol{\alpha}_{\dot{q}}$ in (20). Similarly, the Bézier coefficients of the configuration trajectory $\boldsymbol{q}(t)$ could also be integrated given initial configuration $\boldsymbol{q}_0 \in \mathbb{R}^3$.