Kinodynamic Motion Planning for Multi-Legged Robot Jumping via Mixed-Integer Convex Program

Yanran Ding\textsuperscript{1}, Chuanzheng Li\textsuperscript{1}, and Hae-Won Park\textsuperscript{2}

Abstract—This paper proposes a kinodynamic motion planning framework for multi-legged robot jumping based on the mixed-integer convex program (MICP), which simultaneously reasons about centroidal motion, contact points, wrench, and gait sequences. This method uniquely combines configuration space discretization and the construction of feasible wrench polytope (FWP) to encode kinematic constraints, actuator limit, friction cone constraint, and gait sequencing into a single MICP. The MICP could be efficiently solved to the global optimum by off-the-shelf numerical solvers and provide highly dynamic jumping motions without requiring initial guesses. Simulation and experimental results demonstrate that the proposed method could find novel and dexterous maneuvers that are directly deployable on the two-legged robot platform to traverse through challenging terrains.

I. INTRODUCTION

The ability to perform dynamic motions such as leaping over gaps and jumping on high platforms is a unique 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-
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 method is evaluated on a hardware platform, as shown in Fig. 1. The trajectory that guides the robot to jump on a platform is obtained without providing initial guesses to the MICP, which is not possible for local optimization based trajectory optimization methods.

The paper is organized as follows: Section II introduces the single rigid body dynamics model, mixed-integer wrench constraint, and other constraints including aerial phase continuity and contact location choice. Section III presents the simulation and experiment results of the highly dynamic motion planning. Section IV discusses the results from this work and Section V provides the concluding remarks.

II. TECHNICAL APPROACH

In this section, we briefly introduce the robot dynamic model before the construction of configuration space discretization and feasible wrench polytopes that are used in our formulation. And we also formalize our mixed-integer formulation with assumptions and constraints for aerial phase 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 $q = [x, z, \theta]^T$, where $[x, z]^T$ is the location of the center of mass (CoM) and $\theta$ is the pitch angle. The input to the system is the spatial wrench $\mathcal{F}_s$, and the dynamic model of the robot is

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

where $D = \text{diag}(m, m, I_0)$ is the inertia tensor; $\text{diag}(\cdot)$ creates diagonal matrices; $m$ is the total mass and $I_0$ is the moment of inertia around the CoM along the z-axis; $\alpha_g = [0, -g, 0]^T$ is the gravitational acceleration vector. Assuming there are $N_c$ contact points, each with a ground reaction force (GRF) $f_i \in \mathbb{R}^2$, the spatial wrench is given by $\mathcal{F}_s$

$$\mathcal{F}_s = \sum_{i}^{N_c} \begin{bmatrix} f_i \\ \tau_i^y \end{bmatrix},$$

where $\tau_i^y$ is the moment generated by $f_i$, $\tau_i^y = r_i \wedge f_i$; $r_i$ is the vector from CoM to the $i^{th}$ contact point; $\wedge : \mathbb{R}^2 \times \mathbb{R}^2 \rightarrow \mathbb{R}$ 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 expressed in frame $\{B\}$; the global variable is defined in frame $\{S\}$, named the touchdown state $q^{TD} = [x^{TD}, z^{TD}, \theta^{TD}]^T \in \mathbb{R}^3$, where $[x^{TD}, z^{TD}]^T$ is the origin of frame $\{B\}$ and $\theta^{TD}$ 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 $\mathcal{F}_s$. Let the spatial wrench trajectory be parametrized by the Bézier polynomial with coefficient $\alpha = [\alpha_{f_x}, \alpha_{f_z}, \alpha_{\tau}]^T$, then the trajectories $\dot{q}(t), q(t)$ are also parametrized by Bézier polynomials with coefficients $\alpha_q$ and $\alpha_{\dot{q}}$, respectively. Given initial conditions $q_0, \dot{q}_0$, coefficients $\alpha_q$ and $\alpha_{\dot{q}}$ could be obtained by linear operations.

$$\alpha_q = L(\alpha_f, q_0), \alpha_{\dot{q}} = L(\alpha_{\dot{f}}, q_0)$$

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

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

### Table I

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Unit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>[kg]</td>
<td>2.56</td>
</tr>
<tr>
<td>$I_0$</td>
<td>[kg·m²]</td>
<td>0.04</td>
</tr>
<tr>
<td>$L$</td>
<td>[m]</td>
<td>0.3</td>
</tr>
<tr>
<td>$l_{knee}$</td>
<td>[m]</td>
<td>0.14</td>
</tr>
<tr>
<td>$l_{ankle}$</td>
<td>[m]</td>
<td>0.14</td>
</tr>
<tr>
<td>$\tau_{max}$</td>
<td>[Nm]</td>
<td>9.8</td>
</tr>
</tbody>
</table>

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.
The state and control trajectories during stance are polynomials parametrized by the B´ezier coefficients αF. The state trajectory during aerial phase could be parametrized by the second-order polynomial. To make the kinematic motion planning problem finite dimensional, the continuous time trajectory F_s(t), q(t), q(t) are sampled at a time sequence \{t_k|k = 1, 2, · · · , N_f\}, where N_f is the number of nodes during stance phase.

B. C-space Discretization

The configuration space (C-space) Ω ⊂ R^3 of the robot is the set of configurations that the robot could reach during stance phase without violating kinematic constraints. The set of constraints that define the C-space is

\[ Ω := \{q ∈ R^3 | 0 ≤ \hat{n}_l^T \cdot (p^h_l - p^c_l) \}, \]
\[ 0 ≤ \hat{n}_l^T \cdot (p^k_l - p^c_l) \}, \]
\[ r_{min} ≤ |p^k_l - p^c_l|2 ≤ r_{max} \]
\[ q_{min} ≤ q ≤ q_{max} \}, \]

where \( \hat{n}_l \) is the normal vector of the terrain; \( p^h_l, p^k_l \) are the hip and knee joint positions of the leg \( i \in \{ \text{back, front} \} \) and \( p^c_l \) 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 \( |·|2 \); (4d) is the box constraint on \( q \). 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 \( q \), 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 \( q \) and stance mode. As shown in Fig. 3, the C-space is divided into three disjoint regions corresponding to front stance \( Ω_{fs} \) (blue), double stance \( Ω_{ds} \) (black), and back stance \( Ω_{bs} \) (red), or equivalently,

\[ Ω := Ω_{bs} ∪ Ω_{fs} ∪ Ω_{ds}. \]

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 \( q \) and stance mode, the gait sequence planning is encoded into the C-space construction. Hence, a state trajectory \( q(t) \) in \( Ω \) also contains the gait sequence information.

As shown in Fig. 3, the C-space is a non-convex set. To tackle this problem, the C-space \( Ω \) is discretized into \( N_d ∈ Z^+ \) convex polytopic cells, denoted by \( c_i ∈ Ω, i ∈ \{ 1, · · · , N_d \} \), where \( N_d \) is the total number of cells. The union of the cells is contained within the C-space, namely, \( \cup_{i=1}^{N_d} c_i ⊂ Ω \). 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

\[ A^p_{i geo} \cdot q ≤ b^p_{i geo}, i = 1, · · · , N_d, \]

where the matrices \( A^p_{i geo} \) and \( b^p_{i geo} \) delineate the cell \( c_i \) using the half-plane representation (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 := \{ f ∈ R^2 | |J^T f|_∞ ≤ τ_{max} \}
\[ |f - \hat{n}^T f|_∞ ≤ µn^T f \}, \]

where \( J \) 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 \( τ = J^T f \). Furthermore, assumption 2 implies that given the robot configuration \( q \), joint angles could be calculated through inverse kinematics. The inequality (7a) encodes the joint torque constraint \( |τ|_∞ < τ_{max} \), where \( |·|_∞ \) 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

\[ FWP_i := \{ F_i ∈ R^3 | F_i = \begin{bmatrix} f^k_i \\ r_i \land f^k_i \end{bmatrix}, f^k_i ∈ FFP_i \}, \]
where $f^k$ is the $k^{th}$ vertex of FFP for contact point $i$. Note that the FWP is defined using the vertex-representation of a polytope (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 Minkowski sum [21] of the FWP created by both contact legs,

$$\text{FWP}_q = \bigoplus_{i=1}^{2} \text{FWP}_i,$$

where FWP$_q$ indicates the FWP of the robot at configuration $q$. 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 $q \in \Omega_{ds}$, the corresponding FWP$_q$ is defined as in (9).

For a cell $c_i$ in the C-space discretization, its representative FWP is defined as

$$\text{FWP}_{c_i} = \begin{cases} \bigoplus_{k=1}^{r_i} \text{FWP}^k_{c_i}, & \text{double stance} \\ \text{FWP}^{cbsv}_{c_i}, & \text{single stance} \end{cases}$$

where FWP$^k_{c_i}$ is the FWP of the $k^{th}$ node of the cell $c_i$. FWP$^{cbsv}_{c_i}$ 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 $c_i$. 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 $r_i$, 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

$$A_{i}^{fwp} \cdot \mathcal{F}_s \leq b_{i}^{fwp}, i = 1, \cdots, N_d, \quad (11)$$

where $A_{i}^{fwp}$ and $b_{i}^{fwp}$ are the matrices that describe half-planes. The single stance FWP is subject to equality constraints $A_{s}^{fwp} \cdot \mathcal{F}_s = b_{s}^{fwp}$, 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 $B^{cs} \in \{0,1\}^{N_i \times N_d}$ is constructed such that $B^{cs}_{i,j} = 1$ indicates that $q(t_i)$ is within cell $c_j$ and the spatial wrench should be chosen within FWP$_{c_j}$

$$B^{cs}_{i,j} \Rightarrow A^{geo} \cdot q(t_i) \leq b^{geo}_{j}, \quad A^{fwp} \cdot \mathcal{F}_s(t_i) \leq b^{fwp}_{j}, \quad (12)$$

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

$$\sum_{j=1}^{N_d} B^{cs}_{i,j} = 1, \forall i = 1, \cdots, N_t \quad (13)$$

which requires that at each time step $t_i$, the robot state $q(t_i)$ can only reside within one cell $c_j$.

**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

$$\begin{bmatrix} q^T \bar{T}_T \end{bmatrix} = \begin{bmatrix} q_i^{TO} \\ \bar{q}_i \end{bmatrix} = \begin{bmatrix} \bar{q}_i \end{bmatrix}^{T} + \begin{bmatrix} \bar{q}_i \end{bmatrix} a_\theta T^2_{i,air} \quad (14)$$

where the superscripts ($\cdot$)$^{TO}$ and ($\cdot$)$^{TD}$ indicate variables at take-off and touch-down, respectively; $T^2_{i,air}$ is the aerial time of the $j^{th}$ jump; where $i = 1, \cdots, N_{jp}$ and $N_{jp}$ is the number of jumps. The state $q_i^{TD}$ is the sum of the global state $q_i^{TD}$ and the local state at the start of the next stance phase $q_i^{start}$. Similarly, $q_i^{TO} = q_i^{TD} + q_i^{end}$, where $q_i^{end}$ is the end state of the $j^{th}$ 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 $\bar{q}_i^{TOT} T^2_{i,air}$ 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 $T_{i, \text{air}}^2$ in (14) is approximated by a piecewise affine function.

**F. Foothold Position**

With assumption 2, the foothold position choice is simplified to finding $q_{T_D}$ 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 $B_{i,j}^p \in \{0, 1\}^{N_s \times N_j}$ is constructed to assign foothold positions, where $N_s$ is the total number of terrain segments. $B_{i,j}^p = 1$ implies that at the $j^{th}$ jump, the $q_{T_D}^j$ lies on the $i^{th}$ terrain segment $seg_i$:

$$B_{i,j}^p = 1 \implies q_{T_D}^j \in Q_i, \quad (15a)$$

$$\sum_{i=1}^{N} B_{i,j}^p = 1, \quad \forall j = 1, \ldots, N_{jp}, \quad (15b)$$

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

$$q_{T_D}^{N_{jp}} \in Q_{goal} \quad (16)$$

where $q_{T_D}^{N_{jp}}$ is the touchdown state after the final jump and $Q_{goal}$ 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 $x_{opt} = [\alpha_{C}, q_0, q_{T_D}^D, T_{air}, B^{l'}, B^{r'}]^T$. The complete formulation of the MICP is:

$$\min_{x_{opt}} \quad f(x_{opt})$$

s.t. $q(t_i) \in c_k \subset \Omega$ \quad (17b)

$$\dot{q}(t_i) = D^{-1} \mathcal{F}_s(t_i) + a_g$$

$$\mathcal{F}_s(t_i) \in FWP_{ek}$$

$$q_{T_D}^j \in Q_l$$

$$q_0 \in Q_{init}, \quad q_{T_D}^{N_{jp}} \in Q_{goal}$$

$$\text{aerial phase constraint: (14)}$$

$$i = 1, \ldots, N_i; \quad j = 1, \ldots, N_{jp}$$

$$k = 1, \ldots, N_d; \quad l = 1, \ldots, N_s$$

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 $Q_{init}$ is the feasible set for the initial condition. $f(x_{opt})$ is a task-specific convex objective function. For example, $f(\cdot)$ could be the deviation from the goal $|q_{T_D}^D - q_{goal}|_2$, which makes the problem a mixed-integer quadratic program (MIQP); or it could be $-x_{T_D}^{N_{jp}}$ 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 $\theta$. The robot is externally powered and the Elmo Gold Twitter amplifiers are mounted on the boom. The control loop runs at 4 kHz 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 $\Pi \in \mathbb{R}^{4 \times 3}$ is calculated to map the spatial wrench $\mathcal{F}_s$ to the joint torque $u_{sag} = \Pi \mathcal{F}_s$. Let $q_{sag} := [q^T, q_{T_D}^D]^T \in \mathbb{R}^{7 \times 1}$ be the generalized coordinates of the sagittal dynamics, where $q_i$ is the vector of joint angle. The no-slip ground contact constraints are

$$\dot{p}_{foot} = J_{foot} q_{sag} = 0$$

$$\ddot{p}_{foot} = J_{foot} \ddot{q}_{sag} + J_{foot} \dot{q}_{sag} = 0,$$

where $p_{foot}$ is the foot position, and the foot Jacobian $J_{foot} \in \mathbb{R}^{4 \times 7}$ is partitioned as $J_{foot} = [J_B, J_I]$. The
dynamics of the sagittal plane system are

\[
\begin{bmatrix}
D & 0 & -J_f^T \\
0 & 0 & -J_f^T \\
J_B & J_l & 0
\end{bmatrix}
\begin{bmatrix}
\dot{q} \\
\dot{q}_sag \\
\lambda
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
-f_s \\
-f_j
\end{bmatrix}
\]

(19)

where \( \lambda \) is the Lagrange multiplier associated with the constraints in (18). By solving the system of equations one could derive \(-J_f^T J_f^{-T} u_{sag} = D\ddot{q} = F_s\). Thus, as one of many possible solutions, \( \Pi \) is selected as \( \Pi = -\left( J_B J_l^{-T} \right)^T \), where \((\cdot)^T\) provides the Moore-Penrose pseudoinverse.

Using the sagittal plan control, the feedforward wrench trajectory \( F_{ff} \) from MICP and a proportional-derivative (PD) feedback controller is used to track the desired state trajectory. \( F_{fb} = K_{d}(q_d - q) + K_{a}(q_d - \dot{q}) \), where \( K_a, K_d \) are diagonal gain matrices; \( q_d \) and \( q_d \) 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 \((t \leq 0.2 \text{ s})\) 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 \( \Omega_{ds} \) 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.

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 \( N_{sa} = 10, N_{fs} = 10, N_{ds} = 21 \), the MICP involves 485 variables (53 continuous, 432 integer), and the computational time is 27 s. The algorithm 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
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

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

\[
\frac{M+1}{T_{st}} \Phi(M, T_{st}) \alpha_{\dot{q}} = \left[ D^{-1} \alpha_{\dot{T}} + a_g, \dot{q}_0 \right]^T, \quad (20)
\]

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

\[
\Phi_{i,j} := \begin{cases} 
-1, & j = i = 1, 2, \ldots, M + 1 \\
1, & j = i + 1 = 2, 3, \ldots, M + 2 \\
\frac{T_{st}}{M+1}, & i = M + 2, j = 1 \\
0, & \text{otherwise}.
\end{cases} \quad (21)
\]

The linear operation \(\alpha_{\dot{q}} = L(\alpha_{\dot{T}}, \dot{q}_0)\) is obtained by inverting the matrix in front of \(\alpha_{\dot{q}}\) in (20). Similarly, the Bézier coefficients of the configuration trajectory \(q(t)\) could also be integrated given initial configuration \(q_0 \in \mathbb{R}^3\).