Single Leg Dynamic Motion Planning with Mixed-Integer Convex Optimization

Yanran Ding, Chuanzheng Li and Hae-Won Park

***Abstract*— This paper proposes a mixed-integer convex programming formulation for dynamic motion planning. Many dynamic constraints such as the actuator torque constraint are nonlinear and non-convex due to the trigonometrical terms from the Jacobian matrix. This often causes the optimization problem to converge to local optima or even infeasible set. In this paper, we convexify the torque constraint by formulating a mixed-integer quadratically-constrained program (MIQCP). More specifically, the workspace is discretized into a union of disjoint polytopes and torque constraint is enforced upon a convex outer approximation of the torque ellipsoid, obtained by solving a semidefinite program (SDP). Bilinear terms are approximated by McCormick envelope convex relaxation. The proposed MIQCP framework could be solved efficiently to global optimum and the generated trajectories could exploit the rich features of the rough terrain without any initial guess from the designer. The demonstrated experiment results prove that this approach is currently capable of planning consecutive jumps that navigates a single-legged robot through challenging terrains.**

I. Introduction

The capability to choose isolated footholds in the environment to overcome wide gaps and high platforms is one of the advantages of legged locomotion systems. To traverse challenging terrains, legged systems are often required to execute dynamic motions which include carefully planned multiple jumps such as the one shown in Fig. 1. During such motion with multiple jumps, a robot is subject to stance dynamics, where the robot's leg is in contact with the ground, using its actuators to apply forces; whereas in flight phases the robot follows ballistic motion while carefully choosing the best footholds within the reachable terrain. Thus, the challenge of dynamic motion planning involves finding a feasible trajectory that satisfies constraints from actuator torque limit, robot dynamics, and the interaction with the environment.

One of the methods to solve the multistep motion planning problem is the optimal-control-based approach. In the work of [1], Sequential Action Control (SAC) was proposed to control a spring-loaded inverted pendulum (SLIP) robot to hop upstairs. The optimal control could be computed online by limiting the planning horizon. The sample-based approach, on the other hand, involves random search within the state space. It serves well in kinematic motion planning but poses a bigger challenge in dynamic motion planning, because connecting nodes in the Probabilistic Roadmap (PRM) [2] or reaching random samples from the nearest node in the Rapidly-exploring Random Tree (RRT) [3][4] is difficult trajectory generation problem with dynamic constraints. PRM was used in [5] to solve the ballistic motion planning problem. The method can navigate a point robot through complex terrains with relatively small computational cost, but the stance dynamics is ignored entirely. A RRT based planner was utilized in [6] to create bounding motion for LittleDog on rough terrain. This method incorporates the dynamics of the robot by choosing random samples using reachability guidance. Nevertheless, motion primitives were introduced to reduce the dimensionality of the state-space. Another approach is based on model predictive control. MIT cheetah 2 robot [7] demonstrated in [8] is able to clear a 40 cm high obstacle by solving a constrained nonlinear optimization online. However, the demonstrated applications only include single jumping motions.

Fig. 1: Sequential screenshots of an experiment in which the robot makes two consecutive jumps to reach the goal region on top of the right obstacle.

Many constraints in dynamic motion planning for legged robot are nonlinear and non-convex. Recent works such as [9] and [10] employ integer variables for legged robot motion planning, in which nonlinear and non-convex constraints are replaced by mixed-integer convex constraints. However, joint torque constraints are not considered since the joint torque limits are assumed to be large.

This paper proposes a mixed-integer convex program (specifically, a Mixed-Integer Quadratically Constrained Program, MIQCP) to solve the dynamic multistep motion planning problem with actuator torque constraints. The torque constraint is plagued by the trigonometric terms from the Jacobian matrix, which maps task-space forces to joint torque. In order to tackle this problem, the proposed MIQCP framework adopts the following procedure. The workspace of the robot is discretized into polytopes, and a set of binary variables is used to assign which polytope the center of

Yanran Ding, Chuanzheng Li and Hae-Won Park are with the department of Mechanical Science and Engineering Department, University of Illinois at Urbana-Champaign, IL, 61801, USA. {yding35, cli67, haewon}@illinois.edu.

mass (CoM) is located in. For each polytope, we minimize the convex upper bound of the non-convex torque saturation constraint so that the computed trajectories are guaranteed to be feasible. Bilinear terms are tackled by using McCormick envelope convex relaxation [11], which requires additional binary variables. The presence of large numbers of binary variables complicates the mixed-integer convex program formulation. However, recent advance in the off-the-shelf solvers for mixed-integer convex program [12][13][14] will enable solving the dynamic multistep motion planning problem with global optimality, within a reasonable computation time. For example, the trajectory shown in Fig. 1 can be obtained within 0.9 sec using our formulation and off-the-shelf solvers. The methodology introduced in this work could potentially be applied to more complex robots to solve dynamic multistep motion planning problem. Also, the trajectories obtained from this approach could serve as target motions for complex robots with better chance of satisfying torque constraints.

The paper is organized as follows. Section II introduces the point mass dynamic model, together with mixed-integer convex formulation of torque constraint and bilinear approximation. Section III experimentally demonstrates a highly dynamic robot leg [15] executing the motion plans from the proposed mixed-integer motion planning algorithm, Section IV benchmarks the proposed MIQCP framework, and Section V provides the concluding remarks.

II. Technical Approach

Our task is to drive the robot towards the goal region while traversing a rough terrain. In this work, we take ground reaction forces (GRF) as control inputs to the point mass dynamic model while satisfying the following constraints:

(i) Joint torques, which can be calculated from Jacobian mapping of GRF, do not exceed actuator limits.
(ii) Goal region should be reached at the end of the motion, while making steps in desirable locations in the environment such as horizontal or sloped surfaces.
(iii) Ground reaction forces should be within friction cone and the normal force should be non-negative.

Due to constraints (i) and (ii), formulation for motion planning problem becomes non-convex optimization problem. The main idea is to approximate nonlinear and nonconvex functions with piecewise convex functions, hence making the problem a mixed-integer convex program. In this paper, this idea has been implemented in a point-mass dynamic model which is described next.

A. Point Mass Dynamic Model

A point mass model is used to simplify the dynamics of the problem. The center of mass is assumed to be located at the geometric center of the base. This approximation is valid because the mass and inertia of the thigh and shank links are negligible (<10%) compared to that of the base. The equation of motion (EoM) of the point mass model is $\ddot{\boldsymbol{p}} = \boldsymbol{F}/m - \mathbf{g}$, where m is the total mass lumped at the base; $\ddot{\boldsymbol{p}} = [\ddot{x}, \ddot{z}]^T$ is the linear acceleration of CoM position $\boldsymbol{p} = [x, z]^T$, with x and z defined as the horizontal and vertical coordinates; $\mathbf{g} = [0, g]^T$ is the vector of gravitational acceleration; $\boldsymbol{F} = [F_x, F_z]^T$ is the ground reaction force vector.

Two 5^{th} order Bézier polynomials are used to parametrize the GRF profile, and their coefficients are $\boldsymbol{\alpha}_F = [\boldsymbol{\alpha}_{Fx}, \boldsymbol{\alpha}_{Fz}]$, where $\boldsymbol{\alpha}_{Fx} := [0, \alpha_{Fx(1-4)}, 0]^T$ and $\boldsymbol{\alpha}_{Fz} := [mg, \alpha_{Fz(1-4)}, 0]^T$. The first and last coefficients of $\boldsymbol{\alpha}_{F_x}$ are set as 0 to ensure that the GRF is smooth and feasible. The first coefficient of $\boldsymbol{\alpha}_{F_z}$ is set to mg to compensate for the weight of the leg. A Bézier polynomial is a linear combination of a Bernstein polynomial basis [16], so the integration of a Bézier polynomial is a linear operation [17] on the Bézier coefficients. For example, the linear relationship between GRF Bézier coefficients and velocity Bézier coefficients is

$$\frac{M+1}{T_{st}}\boldsymbol{\Phi}(M, T_{st})\boldsymbol{\alpha}_{\dot{\boldsymbol{p}}} = [\frac{1}{m}\boldsymbol{\alpha}_F^T - g, \dot{\boldsymbol{p}}_0]^T, \tag{1}$$

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

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

By the same token, the position trajectory could also be integrated given initial position $\boldsymbol{p}_0 \in \mathbb{R}^2$.

In order to make the multi-step dynamic motion planning problem finite dimensional, the continuous time trajectory $\boldsymbol{F}(t), \dot{\boldsymbol{p}}(t), \boldsymbol{p}(t)$ are sampled at a time sequence $\{t_k | k = 1, 2, \cdots, N_t\}$, where N_t is the number of time steps.

B. Mixed-integer Convex Torque Constraint

To make the dynamic motion planning problem convex, the non-convex torque constraint is convexified in a mixed-integer program fashion: first, the workspace is discretized as the union of disjoint convex polytopes; one binary variable is assigned to each polytope at every time instance to indicate whether the CoM position is inside the polytope or not; then, a convex outer-approximation of joint torque is calculated for each polytope. The torque constraint is imposed on the convex outer-approximation, making the dynamic motion planning problem a mixed-integer convex optimization problem. The approximation becomes more accurate as the number of polytopes increases and the size of polytopes decreases.

*1) **Workspace Discretization**:* The workspace $\boldsymbol{\Omega} \subset \mathbb{R}^2$ of the robot leg shown in Fig. 2(a) is delineated as follows:

$$0 \leq \boldsymbol{p} \cdot \hat{\boldsymbol{n}}_{ter} \tag{3a}$$

$$l \leq \|\boldsymbol{p} - \boldsymbol{p}_c\|_2 \tag{3b}$$

$$0 < r_{min} \leq \|\boldsymbol{p}\|_2 \leq r_{max} < 2l \tag{3c}$$

$$\theta_{offset} + \theta_{ter} \leq \boldsymbol{arg}(\boldsymbol{p}) \leq \theta_{offset} + \theta_{ter} + \pi \tag{3d}$$

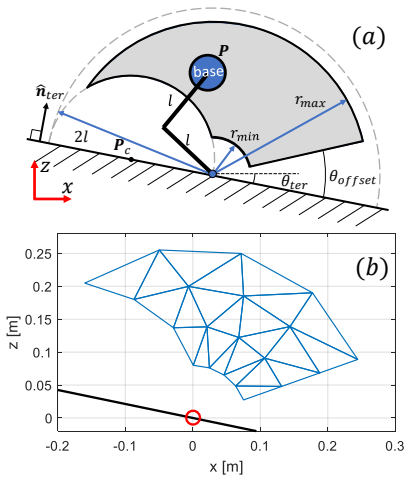

Fig. 2: (a) The viable workspace $\boldsymbol{\Omega}$ of the robot leg on an inclined surface is shown in gray (b) An example of workspace discretization using triangle polytopes ($N_r = 3, N_\theta = 8$). The red circle indicates foot location.

where $\hat{\boldsymbol{n}}_{ter}$ is the terrain unit normal vector, pointing from the ground; $\boldsymbol{p}_c$ is the point where the knee joint would be if it contacts the ground; r_{min} and r_{max} are the minimum and maximum workspace radii; θ_{offset} is the angle offset above the ground. Here the terrain is assumed to be piecewise affine and θ_{ter} is the angle between the terrain surface and the positive x axis; $\|\cdot\|_2$ is the Euclidean norm, while $\boldsymbol{arg}(\boldsymbol{p})$ is the angle between vector $\boldsymbol{p}$ and the positive x axis. Inequalities (3a) and (3b) ensure that both the base and the knee joint are above the ground; (3c) avoids singular configurations; (3d) is set to keep the robot away from extreme poses.

The workspace $\boldsymbol{\Omega}$ defined in equation (3) is discretized into $N_d \in \mathbb{Z}^+$ number of convex polytopes denoted by $\boldsymbol{\omega}_d \subset \boldsymbol{\Omega}$, where $d \in \{1, \cdots, N_d\}$ is the index for a polytope. The union of polytopes $\boldsymbol{\omega}_d$ constitutes an inner approximation of $\boldsymbol{\Omega}$, i.e. $\cup_{d=1}^{N_d} \boldsymbol{\omega}_d \subset \boldsymbol{\Omega}$. The type of polytope could be chosen as either a triangle or a square in 2D cases, or polyhedrons in 3D cases. Fig. 2(b) shows an example of workspace discretization, which uses triangle polytopes to divide $\boldsymbol{\Omega}$ into N_r parts in the radial direction and N_{th} parts in angular direction ($N_d = N_r \cdot N_\theta$). In this example, $N_r = 3, N_\theta = 8$.

In order to indicate the assignment of the CoM position to polytopes, a binary matrix $\boldsymbol{H} \in \{0,1\}^{N_t \times N_d}$ is constructed such that if $\boldsymbol{H}_{k,d} = 1$, then $\boldsymbol{p}(t_k)$ is within polytope $\boldsymbol{\omega}_d$.

$$\boldsymbol{H}_{k,d} \implies \boldsymbol{A}_d \cdot \boldsymbol{p}(t_k) \leq \boldsymbol{b}_d \tag{4}$$

$$\sum_{d=1}^{N_d} \boldsymbol{H}_{k,d} = 1, \ \forall k = 1, \cdots, N_t \tag{5}$$

where matrix $\boldsymbol{A}_d$ and vector $\boldsymbol{b}_d$ encodes the geometry of the polytope $\boldsymbol{\omega}_d$; the *implies* operator ($\implies$) in (4) is implemented using the big-M formulation [18]. The constraint (5) indicates that at each time step, the CoM position could only be assigned to one polytope.

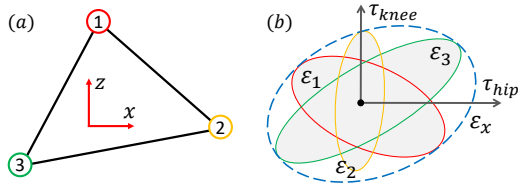

Fig. 3: Geometric interpretation of the minimum bounding ellipsoid problem for the torque ellipsoids at the three vertices of the polytope. $\boldsymbol{\varepsilon}_{1,2,3}$ are the torque ellipsoids associated with vertices $1, 2, 3$. $\boldsymbol{\varepsilon}_x$ is the minimum bounding ellipsoid for $\boldsymbol{\varepsilon}_{1,2,3}$ computed from (10).

*2) **Convex Outer-Approximation of Torque Ellipsoid**:* A main contribution of this work is the proposition of a mixed-integer convex formulation of torque constraint.

The original torque constraint of the robot leg is

$$||\boldsymbol{J}^T(\boldsymbol{p}) \cdot \boldsymbol{F}||_\infty \leq \tau_{max} \tag{6}$$

where τ_{max} is the maximum actuator torque from the motor specification. The Jacobian matrix could be obtained by solving inverse kinematics, given that $\boldsymbol{p}$ is known. The ∞-norm in (6) is first replaced by the 2-norm, which is a more conservative metric for the torque constraint. Then by squaring all terms we get (7).

$$\boldsymbol{F}^T \boldsymbol{J}(\boldsymbol{p}) \cdot \boldsymbol{J}^T(\boldsymbol{p}) \boldsymbol{F} \leq ||\boldsymbol{J}^T(\boldsymbol{p}) \cdot \boldsymbol{F}||_\infty^2 \leq \tau_{max}^2 \tag{7}$$

where the matrix $\boldsymbol{J}\boldsymbol{J}^T$ characterizes the torque ellipsoid. Then N_n points are sampled from a polytope; a example polytope is shown in Fig.3(a). The ellipsoids corresponding to the sample points should all satisfy

$$\boldsymbol{F}^T \boldsymbol{J}_n \cdot \boldsymbol{J}_n{}^T \boldsymbol{F} \leq \tau_{max}^2, \forall n \tag{8}$$

where $n \in \{1, \cdots, N_n\}$ is the index for a sample point.

The set of constraints in (8) could be replaced by one constraint on the convex outer-approximation of the torque ellipsoids, which could be characterized by a positive definite matrix $\boldsymbol{X}$ such that

$$\boldsymbol{F}^T \boldsymbol{J}_n \cdot \boldsymbol{J}_n{}^T \boldsymbol{F} \leq \boldsymbol{F}^T \boldsymbol{X} \boldsymbol{F}, \quad \forall n \tag{9a}$$

$$\boldsymbol{F}^T \boldsymbol{X} \boldsymbol{F} \leq \tau_{max}^2 \tag{9b}$$

As illustrated in Fig.3(b), the geometric interpretation of (9a) is that all the ellipsoids corresponding to matrices $\boldsymbol{J}_n \boldsymbol{J}_n^T$ are enclosed by the ellipsoid corresponding to the matrix $\boldsymbol{X}$, making $\boldsymbol{X}$ an upper bound on the given matrices $\boldsymbol{J}_n \boldsymbol{J}_n^T$. Additionally, the size of the bounding ellipsoid should be minimized to provide a tight convex outer-approximation. The aforementioned problem could be formulated and solved as a semidefinite program [19].

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & \mathbf{tr}(\boldsymbol{W}\boldsymbol{X}), \\ \text{subject to} \quad & \boldsymbol{X} \succeq \boldsymbol{J}_n \boldsymbol{J}_n{}^T, \ n = 1, \cdots, N_n. \end{aligned} \tag{10}$$

where $\boldsymbol{X} \in \mathbb{S}_{++}$ is the optimization matrix variable; $\boldsymbol{J}_n \boldsymbol{J}_n{}^T \in \mathbb{S}_{++}$ is the matrix associated with the ellipsoid $\boldsymbol{\varepsilon}_n$. The linear matrix inequality (LMI) $\boldsymbol{X} \succeq \boldsymbol{J}_n \boldsymbol{J}_n{}^T$ is valid if and only if $\boldsymbol{\varepsilon}_n \subseteq \boldsymbol{\varepsilon}$, $\forall n \in \{1, \cdots, N_n\}$. $\boldsymbol{W} \in \mathbb{S}_{++}$ is a scaling matrix, which is set to the identity matrix. One SDP is posed for each polytope $\boldsymbol{\omega}_d, \forall d \in \{1, \cdots, N_d\}$.

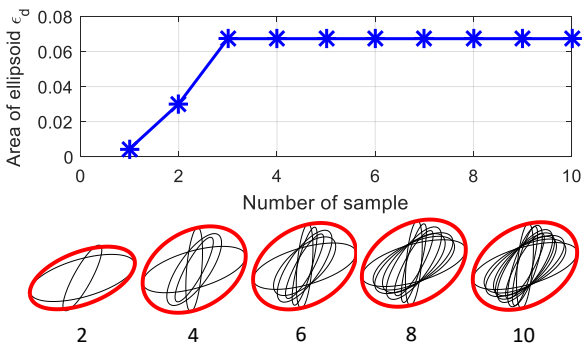

Fig. 4: The change of minimal outer-approximation ellipsoid area with respect to the number of sample points taken in one polytope. The lower part of the figure shows the trend of minimal outer-approximation ellipsoid shape change as N_n increases.

Fig. 4 shows the change of minimal outer-approximation ellipsoid area with respect to the increase in the number of sample points N_n. It is observed that choosing the three vertices of the polytope provides an ellipsoid whose shape and area is almost the same as those with larger number of samples.

Similar SDPs could be posed for all the polytopes, and the LMIs and objective functions could be collected and stacked in a block diagonal fashion to form a single SDP [19]. By solving this single SDP, the minimum torque estimation for all the polytopes could be obtained. The resulting matrix $\boldsymbol{X}_d$ is then used in the torque constraint:

$$\boldsymbol{H}_{k,d} \implies {\boldsymbol{F}_k}^T \cdot \boldsymbol{X}_d \cdot \boldsymbol{F}_k \leq \tau_{max}^2 \tag{11}$$

Note that (11) is a convex quadratic constraint on $\boldsymbol{F}_k$, which is in affine form of the optimization variables.

The minimal upper bound matrix $\boldsymbol{X}$ only needs to be calculated once, for a given grid resolution and terrain inclination.

C. McCormick Envelope Approximation

Since the robot is required to reach the goal region at the end of the motion, the kinematics of flight phase is investigated. After the completion of stance phase, the robot starts the flight phase as a projectile, whose trajectory is described by (12),

$$\begin{bmatrix} x_f^{flt} \\ z_f^{flt} \end{bmatrix} = \begin{bmatrix} x_f^{st} \\ z_f^{st} \end{bmatrix} + \begin{bmatrix} \dot{x}_f^{st} \\ \dot{z}_f^{st} \end{bmatrix} T_{flt} - \frac{1}{2}\boldsymbol{g} \cdot T_{flt}^2 \tag{12}$$

where $[x_f^{flt}, z_f^{flt}]^T$ is the CoM position at the end of the flight phase; $[x_f^{st}, z_f^{st}]^T$ and $[\dot{x}_f^{st}, \dot{z}_f^{st}]^T$ are the CoM position and velocity at the end of stance phase, respectively. Constraints could be imposed on $[x_f^{flt}, z_f^{flt}]^T$ to specify where the robot should be at the end of motion. However, the fact that T_{flt} is also an optimization variable makes the problem nonconvex because of the bilinear terms $\dot{x}_{to}T_{flt}$ and $\dot{z}_{to}T_{flt}$.

The method of using McCormick Envelope [11] to approximate the nonconvex bilinear terms has been used for planning aggressive motions of legged robots [9]. This approach is adopted here to preserve the convex formulation. It could also provide close approximation using a relatively small number of binary variables because the range of each quantity could be obtained from simulation and experiment. Similarly, the quadratic term T_{flt}^2 in (12) is approximated by a piecewise affine function with one binary variable assigned to each section.

D. Foothold Position Choice

The terrain is composed of segments which are modeled by piecewise affine functions. A binary matrix $\boldsymbol{\Xi} \in \{0,1\}^{N_\xi \times N_\delta}$ is constructed to assign foothold positions, where N_ξ is the total number of segments and N_δ is the total number of jumps. $\boldsymbol{\Xi}_{\xi,\delta} = 1$ implies that at the δ^{th} jump, the foothold position lies on the ξ^{th} segment.

$$\boldsymbol{\Xi}_{\xi,\delta} = 1 \implies \begin{cases} x_{fh}^\delta \in [(x_\xi)_{min}, (x_\xi)_{max}] \\ z_{fh}^\delta = a_\xi \cdot x_{fh}^\delta + b_\xi \end{cases} \tag{13}$$

$$\sum_{\xi=1}^{N_\xi} \boldsymbol{\Xi}_{\xi,\delta} = 1, \ \forall \delta = 1, \cdots, N_\delta \tag{14}$$

where $[x_{fh}^\delta, z_{fh}^\delta]^T$ is the foothold position for the δ^{th} jump; a_ξ and b_ξ are the affine function coefficients for the ξ^{th} segment. $[(x_\xi)_{min}, (x_\xi)_{max}]$ is the domain for the ξ^{th} segment.

E. GRF Constraints

The friction cone and unilateral constraints are respected in every jump. For each jump δ, we require that

$$\boldsymbol{\Xi}_{\xi,\delta} = 1 \implies \begin{bmatrix} 1 & -\mu \\ -1 & -\mu \\ 0 & -1 \end{bmatrix} \cdot \boldsymbol{rot}^T(\theta_{ter}^\xi) \cdot \begin{bmatrix} F_x^\delta \\ F_z^\delta \end{bmatrix} \leq 0 \tag{15}$$

where μ is the friction coefficient; $\boldsymbol{rot}(\cdot) \in SO(2)$ is the rotation matrix; θ_{ter}^ξ is the terrain slope for segment ξ, which could be inferred from a_ξ and b_ξ; $[F_x^\delta, F_z^\delta]^T$ is the GRF for step δ.

F. Solving the Problem

The proposed method is implemented in MATLAB [20], using YALMIP [21] and the commercial solver CPLEX [14]. The MIQCP is solved on a desktop with 2.9 GHz Intel i7.

III. Results

A. Experiment Setup

The robot leg used in the experiment is equipped with custom high-fidelity/high-torque force control actuators, enabling execution of torque-demanding dynamic maneuvers [15]. It is mounted on a passive boom system with radius $R_{boom} = 1.25$ m. Two encoders are installed on the base of the boom to measure rotation angles q_x, q_z. Hence, the CoM position of the robot could be retrieved as $[q_x, q_z]^T \cdot R_{boom}$. The base of the robot leg is rigidly fixed to the end of the boom. Other two encoders are mounted on the rotational axis of the hip and knee joints of the robot to measure q_{hip}, q_{knee}. The robot leg was tethered to an off-board power source and computer, and the robot weighs 0.91 kg. Prescribed feedforward force was applied during stance phase, and a simple PD controller was used to control the position of the foot during flight phase. Contact detection was implemented to initiate stance phases.

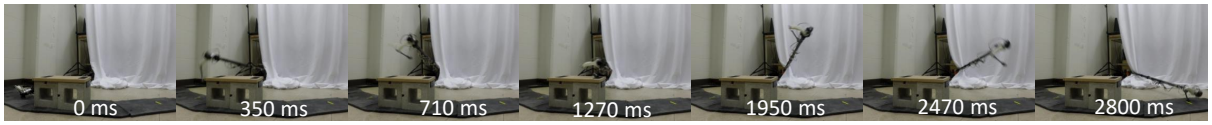

Fig. 5: Sequential screenshots of the first experiment where the robot traverses a terrain with small height variation and reaches the goal region

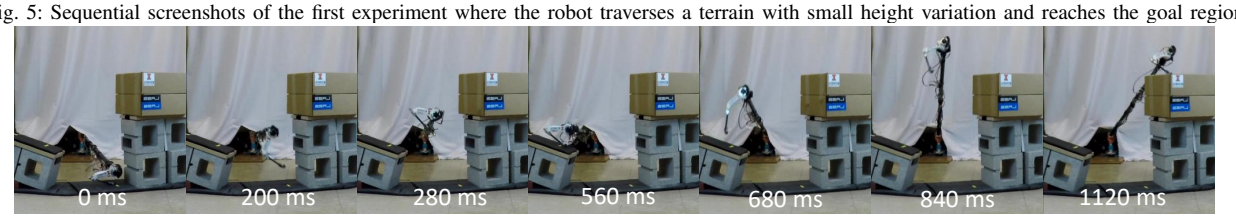

Fig. 6: Sequential screenshots of the second experiment where the robot makes two consecutive jumps to traverse a terrain with large height variation. The MIQCP planner is able to find solutions that utilize the left obstacle as a stepping stone to reach the 0.66 m high goal region.

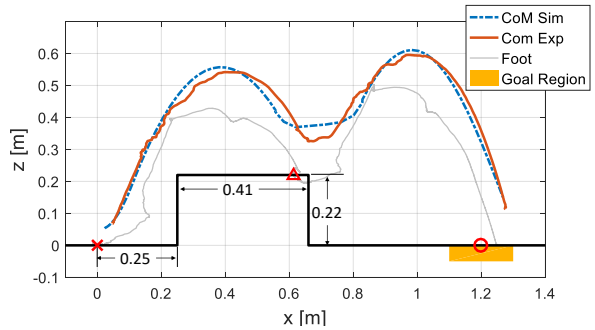

Fig. 7: In the first scenario, the MIQCP algorithm provided a motion plan in which the robot leg first steps on the obstacle and then jumps towards the goal region.

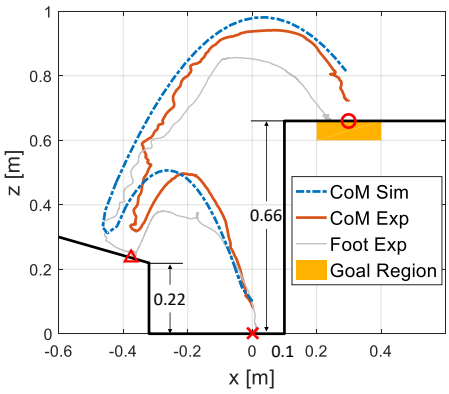

Fig. 8: In the second scenario, the MIQCP algorithm could produce the strategy that uses the left obstacle as a stepping stone for the second jump towards the goal region.

B. Experiment Result

The MIQCP dynamic motion planning algorithm is tested on two terrains which are shown in Fig. 7 and 8. In the first case, the robot is given the task of reaching a goal region while overcoming an obstacle. The Bézier polynomial coefficients for GRF, obtained from MIQCP, was applied on the hardware platform. A sequential screenshot of the experiment is shown in Fig. 5 in which the robot chooses foothold on the obstacle and made two consecutive jumps and landed within the goal region. The simulation and experimental results are shown in Fig. 7. The MIQCP could be solved within 0.71 seconds with the grid resolution $N_r = 3, N_{th} = 5$, time resolution $N_t = 3$, and McCormick envelope resolution of 8×8.

As shown in Fig. 7, the CoM trajectories from the simulation and the experiment are shown as a dotted blue curve and a solid red curve, repectively. The foot position measured from the experiment is shown as a gray curve. The leg swings rapidly at the beginning of the flight phase to track the swing trajectory, causing the unsmoothness of the flight phase CoM trajectory. The stance time is set to be a constant 0.15 sec for both simulation and experiment.

The second terrain has larger height variation, and an additional challenge of an obstacle with sloped surface. In this scenario, the robot is required to reach the goal region on the right obstacle, which cannot be reached with a single jump. The proposed MIQCP algorithm could find the strategy which utilizes the sloped obstacle on the left as a stepping stone, as shown in Fig. 6. Note that the robot cannot stay on the sloped surface statically, and thus has to make two consecutive jumps to reach the goal region. It could be seen from Fig. 8 that the experiment CoM trajectory closely follow the trajectory generated by the proposed MIQCP formulation. The forces outside of the shaded region are induced by tracking the desired swing foot trajectory. This problem could be solved within 0.9 seconds using the same resolution as the previous case.

Movies featuring these two experiments are available in the accompanying video.

IV. Discussion

The proposed MIQCP framework is benchmarked to the canonical Nonlinear Programming (NLP) method. Fig. 9 (b) presents the NLP formulation for the problem of traversing the first terrain. The terrain is modeled by a continuous and smooth function in order to avoid discontinuity of terrain height. To replace the foot choice constraints (13) (14), we design the cost function (shown as green curve in Fig. 9(b)) to penalize landing on the edges of the obstacle. MATLAB's *fmincon* function is used to solve the NLP, which is initiated from three initial conditions: all zeros, heuristic guesses, and solutions from the MIQCP (warm-start). No initial guess is provided to the MIQCP. The continuous time trajectories are discretized identically in both MIQCP and NLP approaches ($M = 5, N_t = 6$). In MIQCP formulation, the workspace is discretized to the refinement of $Nr = 3, N_{th} = 4$, and McCormick envelope to 8-by-8 grid. The torque constraint

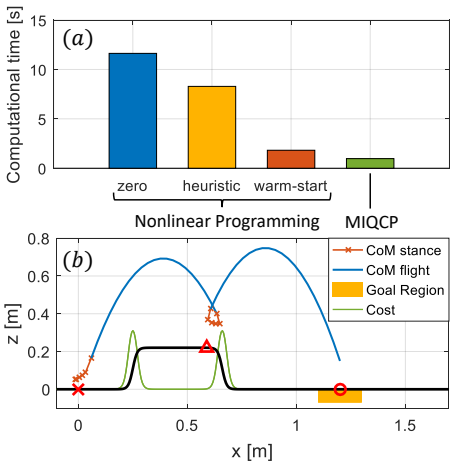

Fig. 9: (a) Comparison of computational time for traversing the first terrain using Nonlinear Programming and MIQCP (b) The nonlinear programming formulation of the first terrain problem: the terrain is modeled as a continuous function, and the cost function penalizes landing on the edges of the obstacle.

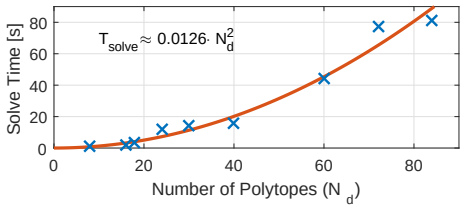

Fig. 10: The change of solve time with respect to the number of polytopes for the workspace discretization in MIQCP

is imposed as (6) in NLP formulation, and as (11) in MIQCP formulation. As shown in Fig. 9, MIQCP outperformed NLP in all the initialization cases. The NLP formulation was not able to provide a feasible solution to the second terrain.

We also investigated how the solve time changes as the dimension of the problem increases in MIQCP. The second terrain problem was solved with increasing number of polytopes N_d for workspace discretization. It could be observed in Fig. 10 that the solve time T_{solve} increases with respect to N_d, whose trend is captured by a quadratic regression curve.

V. Conclusion and Future Work

In this paper, we present a dynamic motion planning framework for single leg multistep planning using MIQCP formulation. The nonlinear, nonconvex torque constraint are approximated with mixed-integer convex constraints by discretizing the workspace into a union of disjoint polytopes. A main contribution of this work is the proposition of a convexification method for the torque constraint, in which the minimal upper bound of torque is found by solving a SDP. The major advantages of this MIQCP formulation is that it provides globally optimal trajectories and it could be solved efficiently. The experiments showed that on a rough terrain with sloped obstacle surface, the MIQCP planner could generate a feasible motion that exploits the rich features of the terrain within a few seconds. Future work will focus on addressing the robustness issue by employing feedback control during the stance phase. Additionally, the potential application of MIQCP framework to support the motion planning of more complex robots provides an exciting direction for future work.

VI. Acknowledgment

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

References

[1] A. R. Ansari and T. D. Murphey, "Sequential action control: closed-form optimal control for nonlinear and nonsmooth systems," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1196–1214, 2016.

[2] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[3] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Computer Science Department, Iowa State University, 1998. [Online]. Available: http://janowiec.cs.iastate.edu/papers/rrt.ps.

[4] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[5] M. Campana and J.-P. Laumond, "Ballistic motion planning," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1410–1416.

[6] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the littledog robot," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 192–215, 2011.

[7] 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.

[8] H.-W. Park, P. M. Wensing, S. Kim *et al.*, "Online planning for autonomous running jumps over obstacles in high-speed quadrupeds," in *2015 Robotics: Science and Systems Conference*. Sapienza University of Rome, pp. 1–9.

[9] 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.

[10] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *2014 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2014, pp. 279–286.

[11] 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.

[12] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2016. [Online]. Available: http://www.gurobi.com

[13] Mosek ApS, "The mosek optimization software," 2014. [Online]. Available: http://www.mosek.com

[14] IBM Corp, "User's manual for cplex," 2010. [Online]. Available: http://pic.dhe.ibm.com/infocenter/cosinfoc/v12r2/topic/com.ibm.common.doc/doc/banner.htm

[15] 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)*. IEEE, 2017.

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

[17] 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.

[18] 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.

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

[20] MATLAB, *version R2016a*. Natick, Massachusetts: The MathWorks Inc., 2016.

[21] J. Lofberg, "Yalmip: A toolbox for modeling and optimization in matlab," in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*. IEEE, 2004, pp. 284–289.