Online Planning for Autonomous Running Jumps Over Obstacles in High-Speed Quadrupeds

Online Planning for Autonomous Running Jumps Over Obstacles in High-Speed Quadrupeds

Hae-Won Park, Patrick M. Wensing, and Sangbae Kim
Department of Mechanical Engineering, MIT, Cambridge, MA, 02139
{parkhw,pwensing,sangbae}@mit.edu

***Abstract*—This paper presents a new framework for the generation of high-speed running jumps to clear terrain obstacles in quadrupedal robots. Our methods enable the quadruped to autonomously jump over obstacles up to 40 cm in height within a single control framework. Specifically, we propose new control system components, layered on top of a low-level running controller, which actively modify the approach and select stance force profiles as required to clear a sensed obstacle. The approach controller enables the quadruped to end in a preferable state relative to the obstacle just before the jump. This multi-step gait planning is formulated as a multiple-horizon model predictive control problem and solved at each step through quadratic programming. Ground reaction force profiles to execute the running jump are selected through constrained nonlinear optimization on a simplified model of the robot that possesses polynomial dynamics. Exploiting the simplified structure of these dynamics, the presented method greatly accelerates the computation of otherwise costly function and constraint evaluations that are required during optimization. With these considerations, the new algorithms allow for online planning that is critical for reliable response to unexpected situations. Experimental results, for a stand-alone quadruped with on-board power and computation, show the viability of this approach, and represent important steps towards broader dynamic maneuverability in experimental machines.**

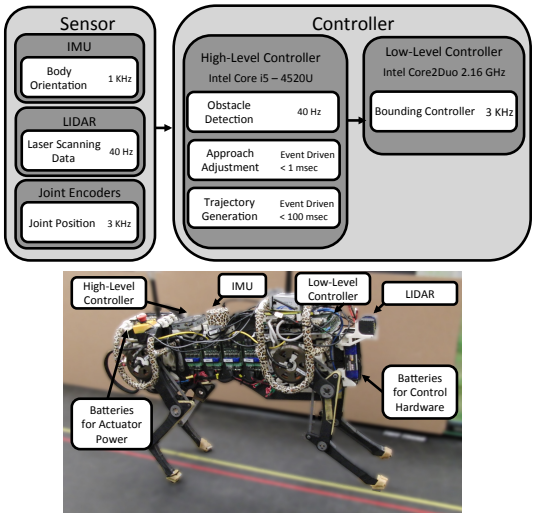

Fig. 1. System overview for online generation of running jumps to clear terrain obstacles. All computation and power is provided on-board.

I. Introduction

Quadrupedal animals display a remarkable versatility to dynamically negotiate challenging terrain in unstructured environments. Whether cornering to evade a predator in an open field or leaping to cross a cavernous gap, biological systems display a nearly seamless capacity to plan and execute complex motions which are tailored to the terrain and task at hand. In the robotics realm to date, a variety of mechanical and computational limitations have prevented legged robots from demonstrating an ability to dynamically negotiate unstructured terrain.

One class of terrain irregularities are terrain obstacles, which require strategies for the system to move around or over the obstruction. In the case of unavoidable extreme terrain obstacles, where static traversal strategies are not viable, the only option may be for the system to take advantage of its dynamics by performing a running jump to clear the obstacle. With this motivation, this paper presents a new set of algorithms that enable dynamic and autonomous running jumps over terrain obstacles through online optimization in an experimental quadruped, the MIT Cheetah 2, shown in Fig. 1.

Recent advances in a diverse set of quadrupedal robotic machines encourage the use of the quadrupedal morphology to study dynamic maneuverability. BigDog and its successor LS3 have demonstrated impressive and robust dynamic walking on a variety of terrain surfaces [19]. StarIETH [9] has displayed a wide array of gaits up to 0.7 m/s with the incorporation of series elastic actuation in its legs. The hydraulically actuated HyQ [21] has shown the capacity for high-power movements and is capable of terrain robust trot walking at 0.35 m/s [23]. The MIT Cheetah 1 [22] and Cheetah 2 [16] robots have shown the capabilities of electric DC motors to enable high-speed locomotion [12] up to 6 m/s.

In addition to limited terrain robustness for unperceived obstacles, many studies have focused on quasi-static locomotion strategies to climb over more challenging terrains. Kalakrishnan et al. [13] proposed learning algorithms for foothold selection based on expert demonstrations using terrain templates. Researchers working on HyQ studied the use of stereo vision to build a height map of the surrounding terrain [1]. These terrain maps were used online to modify the CPGs which governed cyclical leg motions, enabling rough terrain with obstacles up to 9.5 cm in height to be traversed. Reflexes layered on top of foot CPGs have also shown the ability to provide static ambling over obstacles up to 11cm in height [8].

Other work in simulation has studied the capabilities of non-repetitive dynamic movements to enable clearance of more

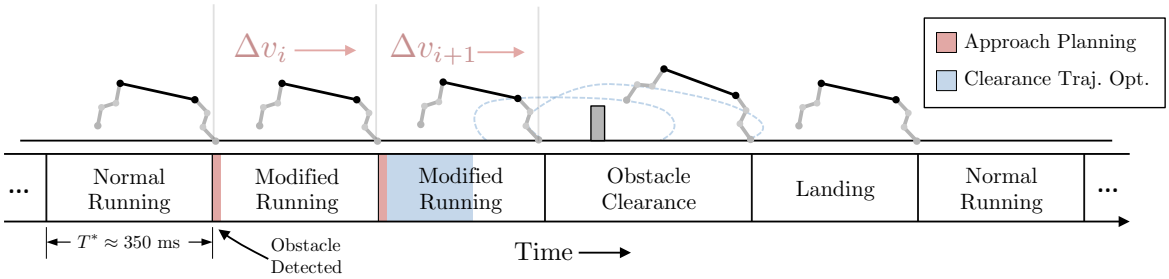

Fig. 2. Procedure overview showing computation breakdown with time. Upon detecting an obstacle, the system plans a modified approach using a multiple-horizon model predictive control strategy (Sec. IV). During the step immediately before the obstacle, the system performs online trajectory optimization (Sec. V-C) in order to select ground force profiles which will allow the quadruped to clear the obstacle.

extreme terrain obstacles. Wong and Orin [26] proposed the use of binary search for a small set of control parameters to produce standing jumps over obstacles. Coros et al. [2] developed an extensive trajectory dictionary and virtual model controller to perform running jumps over a variety of large gaps. Krasny and Orin [14] applied an evolutionary algorithm to optimize running jumps during quadruped galloping. These motions have also been studied in a limited capacity for humanoids [3, 25] by using simple models for trajectory optimization. Still, across each of these studies, the resultant controllers have required extensive offline optimization and have not demonstrated viability for use in experimental hardware.

We present a new framework which considers the dynamics of high-speed locomotion on a variety of time-scales for successful execution of an autonomous running jump to clear an upcoming obstacle. For reliable clearance of the obstacle, it is critical to adjust approach steps to place the robot in an optimal location for the jump. Furthermore, a feasible jumping trajectory must be tailored to the sensed obstruction. Running jumps are autonomously executed in hardware for obstacles between 27.5 cm and 40 cm tall during running at 2.4 m/s. All computation and power is provided on board, with all the algorithms operating at real-time rates sufficient for success of the jump. To the best of the authors' knowledge, this represents the first description and experimental validation of a framework for online planning of autonomous running jumps for obstacle clearance.

II. System Overview

A detailed overview of the framework for obstacle clearance is shown in Fig. 2. This figure highlights the control system components which enable the new autonomous capabilities demonstrated in this work. At a high-level, control system components for approach planning and clearance trajectory optimization are used to move the system into a preferable location relative to the obstacle and to select ground force profiles for a running jump. These components fit into the overall computation strategy as described in the following paragraphs.

During normal running, the quadruped continuously analyzes Lidar data in the sagittal plane to perform obstacle detection, as described in Sec. III. When an obstacle is present, this data is then used to plan a modified approach for the steps leading up to the obstacle. In this work, the approach adjustment strategy considers the use of velocity changes at each step in order to get into proper position. This planning problem is formulated as a series of quadratic programs, as described in Sect. IV, which are solved at each step to provide model-predictive approach control. A lower-level bounding controller is capable to track the desired speed changes commanded through this approach [17].

In the final step before the obstacle, the quadruped performs trajectory optimization for the clearance jump (Section V-C). To accelerate this trajectory optimization procedure, the quadruped is abstracted by a simple model with polynomial dynamics. The trajectory optimization problem is transcribed into a constrained nonlinear programming problem, and solved online. By expressing the forces acting on this system as Bézier polynomials, the polynomial character of the system dynamics enables analytical formula for the system state at any given time. This availability to query the simulated state of the system without numerical integration is a key result in order to generate obstacle clearance jumps on a real-time deadline.

III. Obstacle Detection

In order to perceive information about its surrounding environment, the quadruped was outfitted with a Hokuyo UTM-30LX-EW laser range finder as shown in Fig. 1. This sensor provides distance data in its scan plane with an angular resolution of $0.25°$. To minimize the scan time, data was collected in the sagittal plane only for angles between $45°$ above the horizontal, to $90°$ below the horizontal, as shown in Fig. 3. It was assumed that the motion of the robot during the scan time of 12.5 ms had a negligible effect on the data. Pitch correction from the IMU was used to rotate this data into global coordinates.

Following each scan, a simple line segmentation algorithm was applied to detect the ground plane and the front face of the obstacle. Nguyen et al. [15] provide a thorough overview of line segmentation algorithms for planar data, and report on the promising accuracy and processing speed of the Split and Merge algorithm [18]. In order to further decrease the

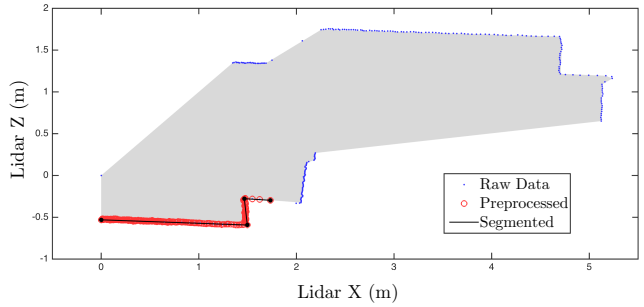

Fig. 3. Lidar data, preprocessing, and final segmentation for a characteristic scan over 135° in the sagittal plane.

computational overhead of the algorithm, a simplification of the Split and Merge algorithm, the Iterative-end-point-fit (IEPF) algorithm [20] was used here. This algorithm begins by constructing a line between the first and last points of the data. It then proceeds to take the point furtherest from the line, splits the data into two halves about this point, and reapplies the algorithm to each half. This recursive process is repeated until all data is within a threshold distance to any segment.

To decrease the size of the input to the IEPF algorithm, data was first preprocessed to extract a contiguous subset of points that contained the ground plane. Starting at 90° below the horizontal, the radial distance of successive data points was monitored for a large jump above a given threshold. All data after the first jump was discarded. In addition, (x, z) data in the sagittal plane within a small tolerance was clustered together and averaged to further decrease the input size. Figure 3 shows the preprocessed data passed to IEPF, as well as its segmented output. The long first segment in this data represents the ground plane, while the second segment represents the front face of the obstacle. The distance to the second segment d_0 and its length h_0 were then used to provide a sensed obstacle distance and height to the approach adjustment and trajectory optimization algorithms. This method was found to be effective during bounding at 3Hz despite significant pitch and small roll oscillations which perturbed the lidar scan plane, as demonstrated further in Sec.VI.

IV. Approach Adjustment

Following the detection of an obstacle, the quadruped has the opportunity to modify its approach. For instance, the system may speed up or slow down to get into a preferable position relative to the obstacle before attempting a clearance jump. In order to determine an optimal approach, this section formulates a series of constrained model-predictive control (MPC) problems over multiple planning horizons. These multiple planning horizons enable the quadruped to determine the optimal number of steps to take before reaching the obstacle as well as its approach velocities at each step.

A. MPC for Approach Adjustment

Suppose that at the current step, the robot is a distance of d_0 away from the obstacle and is traveling at an initial velocity of v_0. At each following step i, the state of the robot state is similarly abstracted by $\mathbf{x}_i \in \mathbb{R}^2$

$$\mathbf{x}_i = [d_i \;\; v_i]^T . \tag{1}$$

In order to control its approach, the system may select speed changes Δv_i at each step. It is assumed that this speed change Δv_i occurs over the duration of the step, providing an average speed of $v_i + \frac{\Delta v_i}{2}$. Given a nominal step period T^*, the state then evolves as

$$\mathbf{x}_{i+1} = \underbrace{\begin{bmatrix} 1 & -T^* \\ 0 & 1 \end{bmatrix}}_{\mathbf{A}} \mathbf{x}_i + \underbrace{\begin{bmatrix} -\frac{1}{2}T^* \\ 1 \end{bmatrix}}_{\mathbf{B}} \Delta v_i . \tag{2}$$

Although these dynamics are a drastic simplification of the full robot dynamics, they capture the main features that are important to approaching the obstacle.

Consider first an N step approach leading up to the obstacle, for some fixed $N \in \mathbb{N}^+$. We are then interested in finding controls $\Delta v_{0:(N-1)}$ that minimally disturb the nominal gait, while landing the final state $\mathbf{x}_N$ within a desired goal region. The goal is defined here by lower and upper bounds:

$$\underline{\mathbf{x}}_N = [\underline{d}_F \;\; \underline{v}_F]^T \quad \text{and} \tag{3}$$

$$\overline{\mathbf{x}}_N = [\overline{d}_F \;\; \overline{v}_F]^T . \tag{4}$$

A desired final state in the middle of the goal region is prescribed as

$$\mathbf{x}_N^d = \frac{\underline{\mathbf{x}}_N + \overline{\mathbf{x}}_N}{2} . \tag{5}$$

An MPC problem is formulated as a quadratic programming (QP) problem with optimal cost $c(N)$ as:

$$\begin{aligned} c(N) = \min \;\; & (\mathbf{x}_N - \mathbf{x}_N^d)^T \mathbf{Q}_F (\mathbf{x}_N - \mathbf{x}_N^d) \\ & + \frac{1}{N} \sum_{i=0}^{N-1} r_i \Delta v_i^2 \qquad (6) \\ \text{s.t. } & \mathbf{x}_{i+1} = \mathbf{A}\,\mathbf{x}_i + \mathbf{B}\,\Delta v_i \\ & \underline{\mathbf{x}}_N \le \mathbf{x}_N \le \overline{\mathbf{x}}_N \\ & \underline{v} \le v_i \le \overline{v} \\ & |\Delta v_i| \le \beta v_i \end{aligned}$$

where $\underline{v} > 0$ and $\overline{v} > 0$ provide velocity limits on the gait, and $\beta \in (0, 1)$ limits the relative acceleration during any given step. To penalize gait changes which happen closer to the obstacle, the cost scalars r_i are selected such that

$$r_i = \max(\mu^{i-(N-1)}, \mu_{min}) \tag{7}$$

for some μ and μ_{min} with $0 < \mu_{min} < \mu < 1$. This discounting strategy encourages larger accelerations to occur further from the obstacle in order to provide additional time to recover from transient effects in the full system which are not captured in this MPC formulation. The quadratic cost on the final state, with $\mathbf{Q}_F = \mathbf{Q}_F^T$ positive definite, rewards final states that have margin to remain in the goal when unmodelled effects in the full system disturb the optimized trajectories.

Parameter	β	μ	μ_{min}	$\mathbf{Q}_F$
Value	0.3	1.1	0.4	diag(2.5, 1)

Variable Bound	d_F	v_F	v_i
Min	0.5 m	1.8 m/s	1.0 m/s
Max	0.9 m	3.2 m/s	4.0 m/s

TABLE I
PARAMETERS FOR THE APPROACH ADJUSTMENT ALGORITHM

Given the bounds on the speed changes and states, it is possible that there is no feasible approach for a given N. In this case, we define $c(N) = \infty$. While most applications of MPC seek to regulate the long-term state of a system and can employ an infinite planning horizon, the unknown number of steps before the obstacle requires us instead to search for the desired planning horizon.

B. Multiple-Horizon MPC

Given the bounds on the forward velocities and final states, $c(N) < \infty$ only if

$$\frac{d_0 - \overline{d}_F}{\overline{v}_F} \leq N \leq \frac{d_0 - \underline{d}_F}{\underline{v}_F}, \tag{8}$$

where the lower step number bound is derived from traveling the shortest distance to the goal region at the maximum speed, and the upper bound follows similarly. This condition introduces a finite range for N to be searched over, given the initial state of the system. The optimal approach length N^* can then be found through evaluation of $c(N)$ for each N in this range. Figure 4 shows the optimal number of steps to be taken before the obstacle as a function of initial state. Algorithm parameters for this example are given in Table I. As distances d_0 become larger, the range of potential planning horizons (8) can introduce many QPs that need to be solved. However, as these QPs are trying to minimize accelerations while directing the state to middle of the goal region, a reasonable estimate for N^* is

$$N^* \approx \text{round}\left(\frac{d_0 - d_F^d}{v_0 T^*}\right). \tag{9}$$

The optimum N^* is found within ± 1 step of this approximation for all cases shown in Fig. 4, allowing us to minimize the number of QPs to be solved.

Recent advances in linear MPC using interior-point [24] and active-set [6] solvers have greatly decreased the computational overhead of online MPC approaches. Here, to solve the QP for $c(N)$, the open-source package qpOASES [7] was used. This package provides a parametric active-set solver that is particularly well suited for MPC. With this solver, QPs can be solved in 250 μs for the horizons considered in our experiments. Although MPC has been used extensively for ZMP optimization [4] and footstep planning in humanoids [11], its use for quadrupeds gait planning has yet been unexplored.

V. REAL-TIME JUMPING TRAJECTORY GENERATION

This section presents methods for online generation of the robot's jumping trajectory to clear an obstacle during running.

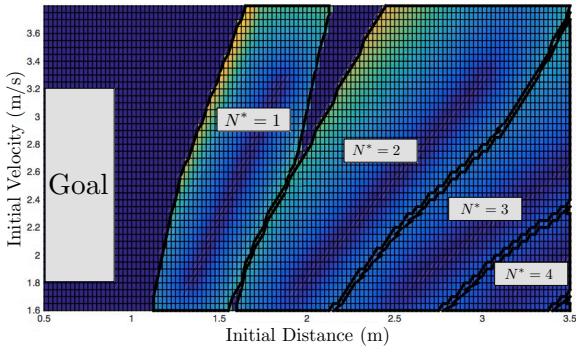

Fig. 4. Optimal number of steps to the goal based on initial state. Color indicates the optimal cost $c(N^*)$. Unenclosed regions represent infeasible regions for the start state, and show the need for early obstacle detection.

A simplified model with polynomial dynamics is introduced. When forced by control inputs which are polynomial with respect to time, this enables the state of the system at any time to be rapidly obtained without numerical integration. This simplification is used to accelerate online trajectory optimization to generate force profiles that are tailored to the sensed obstacle and robot state. In the hardware, these force profiles are ultimately generated using joint torques of the robot through a Jacobian Transpose mapping described in [17].

A. Simplified Model and Temporal Gait Pattern of Bounding

In a quadrupedal bounding gait, the front and hind leg pairs act in parallel. As a result, the quadruped can be modeled as a two-legged system in the sagittal plane, as shown in Figure 5. The model assumes massless legs, and thus the forces/moments exerted by each leg onto the body are statically equivalent to the horizontal and vertical ground reaction forces F^x and F^z on the foot[1]. In practice, these ground reaction forces can be generated by joint torques in the legs. In this simplified model, the robot's generalized coordinates are $q := (x, z, \theta)$, where x is the horizontal position of the center of mass (CoM) with respect to the foot, z is the vertical position of the CoM with respect to the ground, and θ is the body pitch angle as displayed in Figure 5.

Assuming a fixed gait timing, this simplified model is time-switched hybrid, and follows a sequential phase order of Front Stance phase, Aerial I phase, Hind Stance phase, and Jumping Aerial phase (see Figure 5). For later use, we denote t_1, t_2, t_3, and t_4 as the time corresponding to the end of the Front Stance phase, the start and end of the Hind Stance phase, and the end of the Jumping Aerial phase, respectively.

During the Front and Hind Stance phase, in which the front pair of the legs and hind pair of the legs touch the ground, the equations of motion of the robot are given by,

$$\begin{aligned} m\ddot{x} &= F_x \\ m\ddot{z} &= F_z - mg \\ I\ddot{\theta} &= -xF_z + zF_x \end{aligned} \tag{10}$$

[1]Note: This assumption is reasonable as the legs of the cheetah are very light, composing approximately only 10% of the total mass of the system.

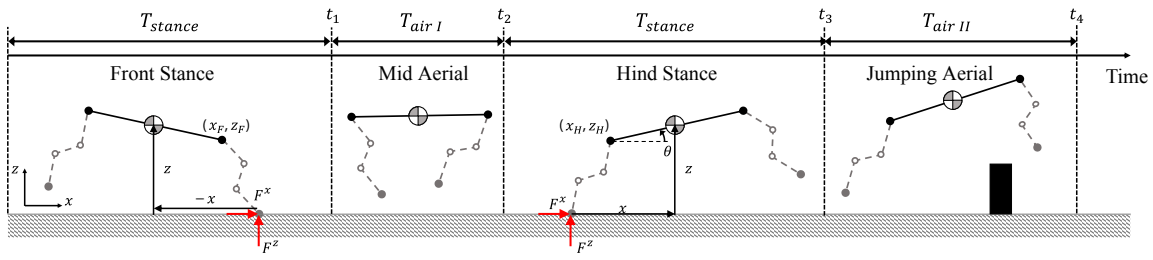

Fig. 5. A simplified time-switched hybrid quadrupedal bounding model. Assuming massless legs, the quadruped is abstracted by a planar rigid body evolving under the influence of ground reaction forces and gravity. A fixed horizontal foot placement relative to the hip at the beginning of stance is used to determine the point of application for the ground reaction force.

where, F_x and F_z are the ground reaction forces applied on the foot of the robot, m and I are the combined mass and inertia of the robot, and g is the gravitational constant. We can further simplify (10) to provide

$$\ddot{x} = u_x \tag{11}$$

$$\ddot{z} = u_z - g \tag{12}$$

$$\ddot{\theta} = -\alpha x u_z + \alpha z u_x \tag{13}$$

where, $u_x = \frac{F_x}{m}$, $u_z = \frac{F_z}{m}$, and $\alpha = \frac{m}{I}$.

In this paper, the scaled forces u_x and u_z are chosen as n^{th}-order Bézier polynomials during the interval $t \in [t_0, t_f]$ where t_0 and t_f represent the beginning and end of the force profile. The Bézier polynomials are given by,

$$u_x(s) = \sum_{i=0}^{n} \beta_{i,x} b_{i,n}(s) \tag{14}$$

$$u_z(s) = \sum_{i=0}^{n} \beta_{i,z} b_{i,n}(s),$$

where $\beta_{i,x}$ and $\beta_{i,z}$ are the Bézier coefficients as optimized in Section V-C, $s := \frac{t-t_0}{t_f-t_0} \in [0,1]$ is the normalized time, and $b_{i,n}(s)$ is the i-th Bernstein Polynomial of degree n

$$b_{i,n}(s) = \binom{n}{i} s^i (1-s)^{n-i}. \tag{15}$$

During the two aerial phases, Mid Aerial phase and Jumping Aerial phase, the ground reaction forces u_x and u_z become zero, and the robot follows ballistic dynamics of $\ddot{x} = \ddot{\theta} = 0$ and $\ddot{z} = -g$.

During the aerial phase, the virtual foot position with respect to the shoulder follows the trajectory shown in Figure 6. After these trajectories are completed, the position of the foot with respect to the shoulder is held to wait for the impact with the ground. These foot swing trajectories are designed considering the workspace of the leg and energy consumption through a separate one-time offline optimization procedure. Their detailed design procedure is omitted here for the brevity of the paper.

The durations of phases T_{stance} and $T_{air\ I}$ are fixed at values required during normal bounding, as given in [16], while the duration of the last phase $T_{air\ II}$ is elongated as the robot jumps up to clear the obstacle. The value of $T_{air\ II}$ is varied according to the height of the robot jumping and is selected by optimization process introduced in Section V-C.

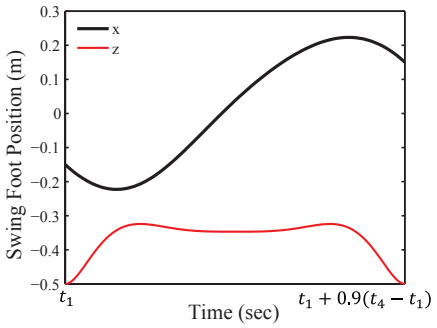

Fig. 6. Swing foot trajectory for the front leg with respect to the shoulder during its aerial period of motion. The trajectory is designed to complete within 90% of the expected flight time to recover from unexpected early impact. The back leg trajectory follows a similar pattern.

B. Solution to the Equations of Motion

Given initial conditions $\left[x_0, z_0, \theta_0, \dot{x_0}, \dot{z_0}, \dot{\theta_0}\right]$ and ground reaction forces (14) in the interval $t \in [t_0, t_f]$, solutions to (11)-(13) can be obtained analytically without numerical integration. As shown in (11) and (12), $x(s)$ and $z(s)$ can be obtained by integrating $u_x(s)$ and $u_z(s)$ twice. Since $u_x(s)$ and $u_z(s)$ are n^{th}-order Bézier polynomials, $x(s)$ and $z(s)$ are $(n+2)^{\text{th}}$-order Bézier polynomials

$$x(s) = \sum_{i=0}^{n+2} c_{i,x} b_{i,n+2}(s) \tag{16}$$

$$z(s) = \sum_{i=0}^{n+2} c_{i,z} b_{i,n+2}(s) \tag{17}$$

for some set of $c_{i,x}$ and $c_{i,z}$.

By taking the second derivative of (16) using common formula [5] for Bernstein polynomials, the dynamics (11) can be used to form $n+1$ linear equations which relate the Bezier coefficients for u_x to those for x. An additional two linear equations relating the Bezier coefficients to initial conditions x_0, $\dot{x}_0$ can be formed to provide

$$\mathbf{C}_x \, \mathbf{c}_x = \mathbf{b}_x(\boldsymbol{\beta}_x, x_0, \dot{x}_0) \tag{18}$$

where, $\mathbf{c}_x \in \mathbb{R}^{n+3}$ is the vector consisting of the coefficients $c_{i,x}$, and $\mathbf{C}_x$ is a constant matrix. Thus, given a selection of a horizontal ground force profile, along with initial conditions, (18) can be quickly solved to obtain the Bezier coefficients

$\mathbf{c}_x$ for x. A similar approach can be used to find the Bezier coefficients $\mathbf{c}_z$ for z

$$\mathbf{C}_z\,\mathbf{c}_z = \mathbf{b}_z(\boldsymbol{\beta}_z, z_0, \dot{z}_0)\,. \tag{19}$$

Similarly, since x and z are $(n+2)^{\text{th}}$-order Bézier polynomials and $u_x(s)$ and $u_z(s)$ are n^{th}-order Bézier polynomials, we can easily deduce that the right hand side of (13), $-\alpha x u_z + \alpha z u_x$, is an $(2n+2)^{\text{th}}$-order polynomial. Thus θ is a $(2n+4)^{\text{th}}$-order Bézier polynomial

$$\theta(s) = \sum_{i=0}^{2n+4} c_{i,\theta} b_{i,2n+4}(s) \tag{20}$$

for some set of $c_{i,\theta}$. Common formula [5] for the products of Bernstein polynomials can be applied to express the right hand side of (13) using Bézier coefficients. Equating these coefficients to those with the second derivative of (20), and using the initial conditions θ_0 and $\dot{\theta}_0$ provides

$$\mathbf{C}_\theta\,\mathbf{c}_\theta = \mathbf{b}_\theta(\boldsymbol{\beta}_x, \boldsymbol{\beta}_y, \mathbf{c}_x, \mathbf{c}_z, \theta_0, \dot{\theta}_0), \tag{21}$$

where, $\mathbf{c}_\theta \in \mathbb{R}^{2n+5}$ is the vector consisting of elements with $c_{i,\theta}$. Hence, Bézier coefficients of $x(s), z(s), \theta(s)$ are easily obtained by solving (18), (19), and (21). Following this procedure, the analytical formula for x, z, and θ from (16), (17), and (20) can be used to quickly query the state of the system at any time $t \in [t_0, t_f]$. To handle the multiple phases of motion, this procedure can be cascaded, enforcing continuous differentiability across transition boundaries.

C. Jumping Trajectory Generation via Constrained Nonlinear Programming

This section explains the new trajectory generation approach for jumping over obstacles. A feasible jumping trajectory to clear the obstacle is obtained by solving a nonlinear programming problem. Optimization variables for this problem include Bézier coefficients for the ground reaction forces $u_x(s)$ and $u_z(s)$ for the front and hind pairs of legs as well as and the duration of the jumping aerial phase $T_{air\ II}$.

The ground reaction force inputs $u_x(s)$ and $u_z(s)$ are designed by adjoining two 3^{th}-order Bézier polynomials, connected at the mid-stance. Force profiles are constrained to follow a common shape in both the x and z directions and in the front and hind legs. In the first half of each stance these profiles are designed with Bézier polynomial coefficients

$$\boldsymbol{\beta}_i^j := \alpha_i^j\,[0\ 0.8\ 1\ 1]$$

where $i \in \{x, z\}$ and $j \in \{f, h\}$ with superscripts f, h representing front and hind legs, respectively. The four scaling factors α_i^j provide flexibility to the optimization to shape the jump trajectory. Profiles in the second half of each stance are given symmetrically by coefficients $\boldsymbol{\beta}_i^j := \alpha_i^j\,[1\ 1\ 0.8\ 0]$. This choice of Bézier polynomial's coefficients provides a single peak trajectory where the peak is located in the middle of the stance phase while the trajectory starts and ends with 0. Figure 7 shows example force profiles when $\alpha_x^f = -180$ N, $\alpha_x^h = -178$ N, $\alpha_z^f = 756$ N, and $\alpha_z^h = 1000$ N.

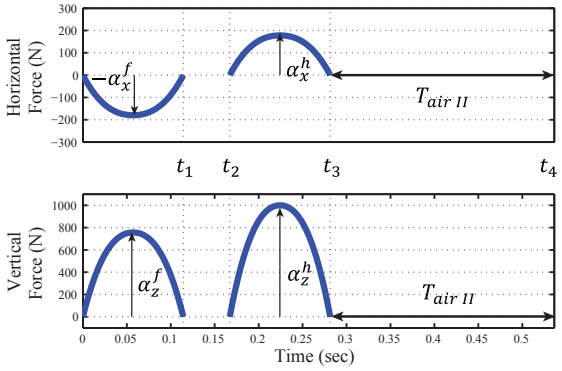

Fig. 7. Example force profile for the front (f) and hind (h) legs optimized to clear an obstacle 33 cm high at a distance of 1 m.

TABLE II
CONSTRAINTS IMPOSED FOR OBSTACLE CLEARANCE FEASIBILITY.

$$|\theta(t)| < \Theta, \qquad \text{for } 0 \le t \le t_4 \tag{22}$$
$$z^f_{foot}(t) > h_{obs}(x^f_{foot}(t)), \qquad \text{for } t_1 \le t \le t_4 \tag{23}$$
$$z^h_{foot}(t) > h_{obs}(x^h_{foot}(t)), \qquad \text{for } t_3 \le t \le t_4 \tag{24}$$
$$x^f_{foot}(t_4) > d_0 + 0.5w_0 \tag{25}$$
$$x^h_{foot}(t_4) > d_0 + 0.5w_0 \tag{26}$$
$$\underline{z} < z^f(t_1) < \overline{z} \tag{27}$$
$$\underline{z}^h < z^h(t_2) < \overline{z}^h \tag{28}$$
$$\underline{z} < z^h(t_3) < \overline{z} \tag{29}$$
$$z^f(t_4) = \tilde{z}^f \tag{30}$$
$$\underline{\theta} < \theta(t_4) < \overline{\theta} \tag{31}$$
$$|\alpha_x^j| < \overline{F}_x \qquad \text{for } j \in \{f, h\} \tag{32}$$
$$0 < \alpha_z^j < \overline{F}_z \qquad \text{for } j \in \{f, h\} \tag{33}$$

To obtain α_x^f, α_x^h, α_z^f, α_z^h, and the arial phase time $T_{air\ II}$, a constrained nonlinear feasibility problem was posed. Since success of the clearance was the dominant goal in this work, an objection function of 0 was employed. This selection also served to accelerate the optimization in comparison to including a more complex objective function. Constraints over the variables were selected as described in Table II. In this table h_0, d_0 are the height and distance to the obstacle, respectively. The parameter w_0 controls the fore-aft obstacle clearance and was chosen as 15 cm in experiments. The desired clearance height around the obstacle is given as a function of forward position $h_{obs}(x)$ by multiplying two sigmoid functions:

$$h_{obs}(x) = h_0 \frac{1}{1 + e^{-\sigma(x-d_1)}} \frac{1}{1 + e^{\sigma(x-d_2)}} \tag{34}$$

with $\sigma = 200$ and $d_1, d_2 = d_0 \pm .5w_0$ respectively.

The constraints in Table II are motivated as follows. The constraint (22) is posed to avoid excessive torso oscillation during the jumping, with the constant $\Theta > 0$ chosen to be $45°$. The constraints (23) and (24) are posed to avoid a foot tripping over the obstacle, (25) and (26) ensure that the robot's foot has achieved the desired lateral clearance at the completion of the jump. The constraints (27) and (29) are used to guarantee that

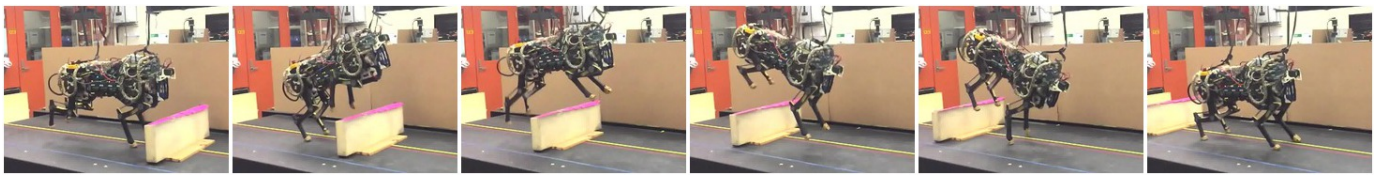

Fig. 8. Motion snapshots from a successful obstacle clearance jump. Frames are separated by 150 ms.

the robot's shoulder position from the ground at the end of the stance phase is within the range of the leg's workspace, and the constants $\underline{z}$ and $\overline{z}$ are chosen from the leg's linkage design. The constraint (28) ensures that the Hind Stance phase starts with the hind shoulder height within the range of the hind leg's workspace, and $\underline{z}^h$ and $\overline{z}^h$ are chosen from the leg's linkage design. The constraints (31) and (30) are posed to provide a configuration for safe landing. $\underline{\theta}$ and $\overline{\theta}$ are lower and upper bounds of pitch angle at the landing, $\tilde{z}^f$ is the vertical foot position with respect to the shoulder at the end of the jumping.

The above nonlinear feasibility problem was solved using the nonlinear solver SNOPT [10]. Continuous constraints (22)-(24) were discretized in time with 50-100 time steps per gait phase. Computation of the parameters for a feasible jumping trajectory takes less than 100 msec on average using the on-board computer with Intel Core i5-4520U. Figure 7 shows an example force profile with parameters obtained solving this nonlinear programming problem for an obstacle with the distance of 1 m and height of 0.33 m.

In implementation for the experimental hardware, trajectory optimization was carried out on any step when the approach modification provided $N^* \leq 2$. The optimization was carried out when $N^* = 2$ as a precaution in case the following step placed the quadruped too close to the obstacle and an emergency jump was required. In all cases, a predicted distance to the obstacle at the next step $d_0 - v_0 T^* - \frac{1}{2}\Delta v_0^*$ was passed to the trajectory optimization using the results of the approach modification.

D. Landing Control

The purpose of the landing control is to provide a safe and robust transition back to the normal running gait by handling disturbances specific to the high impacts on the front legs at landing that are experienced following the running jump. We modify the horizontal and vertical force profiles of the Front Stance phase to provide additional impulse to recover the robot's velocity states to their nominal values. Because we use Bézier polynomials in (14) for force profiles $u_x(s)$ and $u_z(s)$, the impulse $\int_0^{T_{stance}} u_x(s)\mathrm{d}t$ and $\int_0^{T_{stance}} u_z(s)\mathrm{d}t$ can be calculated by simply averaging Bézier coefficients and multiplying the length of duration. Therefore, by uniformly scaling Bézier coefficients accordingly, we can modify the impulse created by $u_x(s)$ and $u_z(s)$.

The vertical landing speed of the CoM $\dot{z}(t_4)$ at the moment of the landing is obtained from the jumping trajectory optimization explained in Section V-C. The required vertical impulse to steer $\dot{z}(t_4)$ to the nominal value of $\dot{z}_0$ can be calculated using linear impulse and momentum change relationship which is given by,

$$\int_0^{T_{stance}} u_z \mathrm{d}t = \dot{z}_0 + \dot{z}(t_4) + gT_{stance} \tag{35}$$

On the other hand, the horizontal force is modulated to modify rotational impulse. Unlike the normal running case where the swing foot retracts fast enough to provide ground speed matching, the swing foot speed with respect to the ground will be large after the jumping over obstacles because the foot position with respect to the shoulder position is held in position after the completion of the desired swing foot trajectories. Due to the large relative swing foot speed at impact, excessive rotational momentum about the CoM will be generated upon impact with the ground. To handle this disturbance, the horizontal force profile is adjusted to cancel out additional rotational momentum delivered by the impact.

After the calculation of required vertical impulse and rotational impulse, α_x^f and α_z^f were modified for force profiles $u_x(s)$ and $u_z(s)$ to provide required impulses.

VI. Results

With our framework the quadruped is able to autonomously jump over obstacles up to 40 cm in height. This maximum height represents 80% of the quadruped's nominal leg length.

A. Setup

The algorithms described in the previous sections were applied for online validation of the obstacle clearance framework. All high-level control for obstacle detection, approach adjustment, and trajectory optimization was performed using a single core of a 2.6 GHz Intel Core i5-54250U processor on a Nuc Mini-PC. Lower-level bounding control was carried out on a 2.16 GHz Intel Core2Duo SpeedGoat board running compiled Simulink. The base control frequency for the lower-level bounding controller was 3 KHz with further details provided in [17].

The quadruped was brought to a steady state 2.4 m/s bounding gait before the running jump algorithms were tested. During normal running, a wireless link to a host computer was used to provide a desired running speed as well as a heading set point from an operator to address lateral drift on the treadmill. All other control actions were completed on board. To clear the obstacle, no information regarding the placement timing or obstacle height was provided to the system in advance. Figure 8 shows snapshots of the obstacle clearance jump in response to a 27.5 cm high obstacle. Video results demonstrate the capabilities of this planning framework to clear 34 cm and 40 cm obstacles without any retuning of controller parameters.

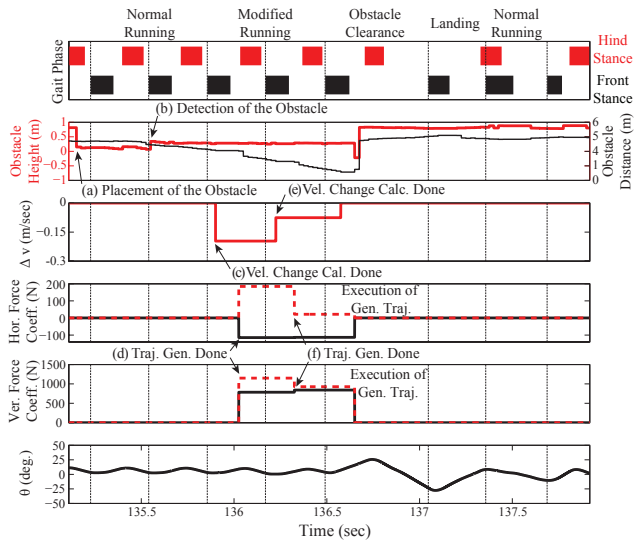

Fig. 9. Algorithm outputs over time for an obstacle clearance jump at 2.4 m/s over a 27.5 cm obstacle. Section VI-B provides additional details.

B. Algorithm Outputs

Figure 9 displays many of the algorithm outputs over time as the quadruped detected, approached, and cleared the obstacle. At data point **(a)** the obstacle is moved onto the treadmill. During this process, the gap between the obstacle and the ground produces errant height readings. A small amount of logic was applied to disregard static obstacles at a far distance from the quadruped. At **(b)** the obstacle is first correctly detected. Between this time, and the final execution of the jump, the detection algorithm provides an average height of 28.1 cm with a standard deviation of 1.6 cm. Use of this detected obstacle occurs at the following step, when the approach adjustment algorithm is run. The algorithm finished at **(c)** and commands a velocity decrease to position the cheetah before the obstacle. Note that the velocity change commanded in the following step at **(e)** is smaller due to the discounting strategy applied by (7).

As shown in the fourth and fifth subplots, trajectory optimization was applied during both modified running steps. In all cases, a safety margin of 3 cm was added to the sensed height to provide additional clearance. The jump optimization process began immediately after **(c)** and completed at **(d)** in under half a gait cycle. This first optimization was precautionary, in case the approach adjustment algorithm determined that a clearance jump was required at the following step. Note however, that both horizontal and vertical scaling factors are larger for this precautionary jump (in comparison to the following step) since the emergency jump would need to be performed from a farther distance. Here, the system evolves as predicted by the the MPC approach controller, and an additional step is possible before the obstacle. The clearance trajectory computed at **(f)** is applied 100 ms later, at the beginning of the next step.

C. Discussion

During the experiment, six jumping trials were carried out. The robot could jump over the obstacles successfully four times out of six trials, but two failures were observed. The first failure was caused when there was not enough distance between the obstacle and the robot when the obstacle was introduced. The treadmill is only 4 meters long, so the robot did not have enough steps to adjust the approach and failed to initiate the jump with appropriate distance from the obstacle. The other failure was caused when the obstacle fell over, as our algorithm is not yet capable of adapting to dynamic obstacles.

During one of the four successful trials, we observed that the robot's front left foot slightly collided with the obstacle. In this experiment, the obstacle was tilted so not perfectly perpendicular to the running direction. Because the sensor only scans in the sagittal plane for the setup, the robot could not detect that the part of the obstacle in front of the leg was closer than the body. This would not have happened if the obstacle was perpendicular to the running direction.

VII. Conclusion

This paper has developed new algorithms to perform a dynamic obstacle clearance movement in response to a sensed obstruction up to 80% of the quadruped's leg length. The quadruped reasons about its dynamics over multiple time frames to select appropriate actions to prepare for and execute the running jump. A model-predictive approach controller places the system in an optimal location relative to the obstacle to perform the running jump. By focusing on the the most important characteristics of the approach, this strategy is able to be computed in 250 μs, and also determines the optimal number of step to be taken before the obstacle. In the final steps of the approach, a more detailed trajectory optimization problem is solved which determines ground force profiles to propel the system up and over the obstacle while maintaining clearance for its swing legs. These algorithms working in tandem have shown viability in an experimental quadruped, providing dynamic running jumps online during a 2.4 m/s gait.

Future research will build upon this work to integrate reactive modification of leg trajectories in response to properties of the sensed obstacle. Given the additional computation time available with the current algorithms, this addition appears computationally feasible. Additional work will also study the performance of this framework with terrain obstacles in outdoor environments. Earlier detection will provide additional availability to modify the approach, potentially resulting in more favorable final states to begin the running jump. However, this early detection also provides the possibility to consider obstacle avoidance maneuvers such as a sharp turn. With these capabilities, solid foundational dynamic movement controllers will provide a basis to pursue dynamic motion planning for legged machines in yet further unstructured environments.

VIII. Acknowledgements

The authors would like the acknowledge support from the DARPA M3 program and from the Korean Agency for Defense Development under the contract UD140073ID.

References

[1] Stéphane Bazeille, Victor Barasuol, Michele Focchi, Ioannis Havoutis, Marco Frigerio, Jonas Buchli, Darwin G. Caldwell, and Claudio Semini. Quadruped robot trotting over irregular terrain assisted by stereo-vision. *Intelligent Service Robotics*, 7(2):67–77, 2014.

[2] Stelian Coros, Andrej Karpathy, Ben Jones, Lionel Reveret, and Michiel van de Panne. Locomotion Skills for Simulated Quadrupeds. In *ACM SIGGRAPH 2011*, pages 59:1–12, New York, NY, USA, 2011. ACM.

[3] C.M. Dellin and S.S. Srinivasa. A framework for extreme locomotion planning. In *IEEE Int. Conf. on Robotics and Automation*, pages 989–996, May 2012.

[4] Dimitar Dimitrov, Alexander Sherikov, and Pierre-Brice Wieber. A sparse model predictive control formulation for walking motion generation. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2292 –2299, Sept. 2011.

[5] 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*, 24(4):559 – 565, 2011.

[6] H.J. Ferreau, H.G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.

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

[8] Michele Focchi, Victor Barasuol, Ioannis Havoutis, Jonas Buchli, Claudio Semini, and Darwin G. Caldwell. Local Reflex Generation for Obstacle Negotiation in Quadrupedal Locomotion. In *Proc. of CLAWAR*, 2013.

[9] C. Gehring, S. Coros, M. Hutter, M. Bloesch, M.A. Hoepflinger, and R. Siegwart. Control of dynamic gaits for a quadrupedal robot. In *IEEE Int. Conf. on Robotics and Automation*, pages 3287–3292, May 2013.

[10] P. Gill, W. Murray, and M. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Review*, 47(1):99–131, 2005.

[11] A. Herdt, N. Perrin, and P.-B. Wieber. Walking without thinking about it. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 190–195, Oct 2010.

[12] Dong Jin Hyun, Sangok Seok, Jongwoo Lee, and Sangbae Kim. High speed trot-running: Implementation of a hierarchical controller using proprioceptive impedance control on the MIT Cheetah. *The International Journal of Robotics Research*, 33(11):1417–1445, 2014.

[13] Mrinal Kalakrishnan, Jonas Buchli, Peter Pastor, Michael Mistry, and Stefan Schaal. Learning, planning, and control for quadruped locomotion over challenging terrain. *The International Journal of Robotics Research*, 30(2): 236–258, 2011.

[14] D.P. Krasny and D.E. Orin. Evolution of dynamic maneuvers in a 3D galloping quadruped robot. In *IEEE Int. Conf. on Robotics and Automation*, pages 1084–1089, 2006.

[15] Viet Nguyen, Stefan Gächter, Agostino Martinelli, Nicola Tomatis, and Roland Siegwart. A comparison of line extraction algorithms using 2D range data for indoor mobile robotics. *Autonomous Robots*, 23(2):97–111, 2007.

[16] Hae-Won Park, Meng Yee Chuah, and Sangbae Kim. Quadruped bounding control with variable duty cycle via vertical impulse scaling. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3245–3252, Sept 2014.

[17] Hae-Won Park, Sangin Park, and Sangbae Kim. Variable-speed quadrupedal bounding using impulse planning: Untethered high-speed 3D running of MIT Cheetah 2. In *IEEE Int. Conf. on Robotics and Automation*, 2015.

[18] Theodosios Pavlidis and S.L. Horowitz. Segmentation of Plane Curves. *IEEE Transactions on Computers*, C-23 (8):860–870, Aug 1974.

[19] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, Rob Playter, and the BigDog Team. BigDog, the Rough-Terrain Quaduped Robot. In *Proceedings of the 17th World Congress*, pages 10822–10825, 2008.

[20] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244 – 256, 1972.

[21] C Semini, N G Tsagarakis, E Guglielmino, M Focchi, F Cannella, and D G Caldwell. Design of HyQ – a hydraulically and electrically actuated quadruped robot. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 225 (6):831–849, 2011.

[22] Sangok Seok, A. Wang, Meng Yee Chuah, D. Otten, J. Lang, and Sangbae Kim. Design principles for highly efficient quadrupeds and implementation on the MIT Cheetah robot. In *IEEE Int. Conf. on Robotics and Automation*, pages 3307–3312, May 2013.

[23] B. Ugurlu, I. Havoutis, C. Semini, and D.G. Caldwell. Dynamic trot-walking with the hydraulic quadruped robot - HyQ: Analytical trajectory generation and active compliance control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 6044–6051, Nov 2013.

[24] Yang Wang and S. Boyd. Fast Model Predictive Control Using Online Optimization. *IEEE Transactions on Control Systems Technology*, 18(2):267–278, March 2010.

[25] P.M. Wensing and D.E. Orin. Development of high-span running long jumps for humanoids. In *IEEE Int. Conf. on Robotics and Automation*, pages 222–227, May 2014.

[26] Ho Cheung Wong and David E. Orin. Control of a quadruped standing jump over irregular terrain obstacles. *Autonomous Robots*, 1:111–129, 1995.