2022 IEEE International Conference on Robotics and Automation (ICRA)
May 23-27, 2022. Philadelphia, PA, USA

A Linearization of Centroidal Dynamics for the Model-Predictive Control of Quadruped Robots

Wanchao Chi, Xinyang Jiang, Yu Zheng[†], *Senior Member, IEEE*

***Abstract*— Centroidal dynamics, which describes the overall linear and angular motion of a robot, is often used in locomotion generation and control of legged robots. However, the equation of centroidal dynamics contains nonlinear terms mainly caused by the robot's angular motion and needs to be linearized for deriving a linear model-predictive motion controller. This paper proposes a new linearization of the robot's centroidal dynamics. By expressing the angular motion with exponential coordinates, more linear terms are identified and retained than in the existing methods to reduce the loss from the model linearization. As a consequence, a model-predictive control (MPC) algorithm is derived and shows a good performance in tracking angular motions on a quadruped robot.**

I. Introduction

Being able to traverse various terrains, quadruped robots are ardently desired to offer services in industrial and social environments, such as patrol, surveillance, security, delivery, and etc. Nowadays, many famous quadruped robots have emerged, including BigDog, WildCat and Spot series from Boston Dynamics [1], HyQ series from IIT [2], ANYmal from ETH [3], Cheetah series from MIT [4], Jueying series from DeepRobotics [5], and Laikago and Aliengo from Unitree [6], which all contributed a great deal to the development of quadruped robots. To realize locomotion, almost all current quadruped robots use a high-level motion planner to generate reference motions including trajectories of every degree of freedom (DoF) and walking gaits followed by a real-time motion controller for the robots to execute the reference motions. Such a control architecture is also used by biped robots. Therefore, real-time motion control is a core problem in the research of legged locomotion.

Since most modern quadruped and biped robots are torque-controlled, a real-time motion controller is aimed at computing required joint torques for a robot to follow the reference motion. In doing this, several relations and constraints need to be satisfied for the computed joint torques to be practically reasonable. First, the robot's dynamics, which relates the robot's motion to its joint torques and other external forces, typically the forces at the contacts between the robot and the environment, must be considered. Besides, joint torques have limits and contact forces should satisfy the friction constraint to avoid causing slip at contact. In addition, some kinematic constraints at contact may have to be considered as well. Overall, the problem usually is underdetermined and written as a quadratic programming (QP) problem, which is a type of optimization problem that can be easily solved in real time to obtain optimal joint torques for controlling a robot.

[†] denotes the corresponding author. e-mail: petezheng@tencent.com
The authors are with Tencent Robotics X, Shenzhen, Guangdong, China.

A. Related Work

Whole-body dynamics describes the motion of every DoF of a robot with relation to the joint torques and contact forces. Particularly for humanoid robots whose legs and arms have nonnegligible masses, motion controllers considering whole-body dynamics have been developed. In such controllers, while the desired acceleration of every DoF consists of the reference acceleration and a feedback term determined using PD control law, the joint torques and contact forces are the unknowns in the resulting QP problem with the primary objective to realize the desired acceleration as much as possible. Tracking of desired center of pressure for balance control [7], [8], [9], [10] or task-space motion [11], [12], [13], [14] and regulation of joint torques and contact forces are often added as additional objective terms in the QP. Although computing joint torques and contact forces on the whole-body model simultaneously results in a large QP problem, it can still be solved in real time on a modern PC.

To have smaller problems and less computation time, some methods choose to eliminate contact forces from the equation of motion by using a projection matrix [15], [16], [17], [18], [19], [20], [21], [22]. Other methods first compute the contact forces by much smaller QP problems and then map them to the joint torques through the contact Jacobian directly or a linear equality constrained least-squares problem [23], [24], [25], [26], [27], [28], [29], [4], [30], [31], [32]. To do this, researchers often resort to the centroidal dynamics of a robot [23], [24], [25], which describes the motion of the robot's CoM and the change of its angular momentum with respect to the contact forces and does not involve the joint torques. Similar motion controllers have also been popular for quadruped robots [27], [28], [29], [4]. Compared with biped robots, the leg's mass of quadruped robots is usually negligible and their centroidal dynamics involves only a single body, which is much easier to compute. Such a torque-free relation between the robot's motion and contact forces can also be derived using decoupled dynamics [19], [30] or the equations of motion in the whole-body dynamics corresponding to the robot's floating base [31], [32], [33]. Furthermore, by omitting the product term of angular velocity and assuming constant inertia tensor, the equation of centroidal dynamics gives a linear relation between the robot's acceleration and contact forces, which enables the development of model-predictive control (MPC) of both biped and quadruped locomotion [34], [35], [36], [37]. MPC algorithms can also be derived using other linearization techniques [38] or from the whole-body dynamics [39], [40].

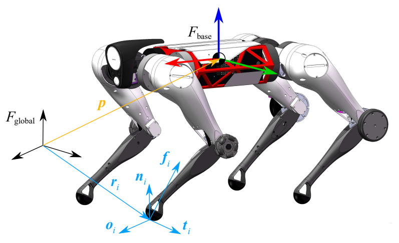


Fig. 1. A quadruped robot developed by Tencent Robotics X.

B. Our Work

Following the state of the art, we derive a model-predictive motion controller for quadruped robots based on the robot's centroidal dynamics. Instead of simply omitting the terms in the centroidal dynamics that seem to be nonlinear over a long time as in [35], [36], for a short prediction horizon in the MPC we decompose those terms and discover several components that are still linear with contact forces over the prediction horizon and can be retained to reduce the loss in the linearization of centroidal dynamics. During the horizon, the robot's state is described by its change from the initial value at the beginning of the horizon and the change of the robot's orientation is approximated with the exponential coordinate of rotation. By identifying and omitting truly nonlinear components, a new model-predictive controller is formulated as a QP problem and tested on a real quadruped robot with severe motions.

The rest of this paper is organized as follows. Section II introduces the robot's centroidal dynamics, for which a linear approximation is derived, leading to a new MPC controller, in Section III. Section IV shows the experimental validation of the proposed controller. Section V concludes this paper.

II. Robot Centroidal Dynamics

We assume negligible mass for the legs of a quadruped robot. Then, the center of mass (CoM) of the robot coincides with the CoM of its body, which is selected as the robot's floating base. We set the body frame F_{base} at the robot's CoM, while the global frame F_{global} can be anywhere in space, as depicted in Fig. 1. Let $\boldsymbol{p} \in \mathbb{R}^3$ and $\boldsymbol{R} \in SO(3)$ denote the position and orientation of F_{base}, respectively. Furthermore, let $\dot{\boldsymbol{p}}$ and $\ddot{\boldsymbol{p}}$ denote the linear velocity and acceleration and $\boldsymbol{\omega}$ and $\dot{\boldsymbol{\omega}}$ the angular velocity and acceleration of the robot's base, respectively. Unless otherwise indicated, all the quantities in this paper are expressed in frame F_{global}.

Omitting the mass of each leg, we can simplify the angular momentum $\boldsymbol{\mathcal{L}}$ of the robot as

$$\boldsymbol{\mathcal{L}} = \boldsymbol{\mathcal{I}}\boldsymbol{\omega} \tag{1}$$

where $\boldsymbol{\mathcal{I}}$ is the instantaneous inertia tensor of the robot's base relative to the frame at the origin of frame F_{base} aligned with frame F_{global} and its relation to the body inertia tensor $\boldsymbol{\mathcal{I}}_B$ relative to frame F_{base} is given by

$$\boldsymbol{\mathcal{I}} = \boldsymbol{R}\boldsymbol{\mathcal{I}}_B\boldsymbol{R}^T. \tag{2}$$

The time derivative of $\boldsymbol{\mathcal{L}}$ is

$$\dot{\boldsymbol{\mathcal{L}}} = \boldsymbol{\mathcal{I}}\dot{\boldsymbol{\omega}} + \hat{\boldsymbol{\omega}}\boldsymbol{\mathcal{I}}\boldsymbol{\omega} \tag{3}$$

where the symbol $\wedge$ represents the conversion of a vector in $\mathbb{R}^3$ into the skew-symmetric matrix for calculating its cross product with another vector.

Assume that the robot makes N contacts with the environment and the position of contact i ($i = 1, 2, \ldots, N$) in frame F_{global} is $\boldsymbol{r}_i \in \mathbb{R}^3$. The centroidal dynamics of the robot can be described by the following two equations:

$$\sum_{i=1}^{N} \boldsymbol{f}_i = m(\ddot{\boldsymbol{p}} - \boldsymbol{g}) \tag{4a}$$

$$\sum_{i=1}^{N} (\hat{\boldsymbol{r}}_i - \hat{\boldsymbol{p}})\boldsymbol{f}_i = \dot{\boldsymbol{\mathcal{L}}} \tag{4b}$$

where $\boldsymbol{f}_i \in \mathbb{R}^3$ is the contact force at contact i, m is the robot's mass, and $\boldsymbol{g} \in \mathbb{R}^3$ is the gravitational acceleration with respect to frame F_{global}. Actually, (4a) is Newton's law describing the linear motion of the robot's CoM or its base, while (4b) is Euler's equation describing the overall angular motion of the robot or its base. In the above discussion, $\boldsymbol{p}$, $\dot{\boldsymbol{p}}$, $\ddot{\boldsymbol{p}}$, $\boldsymbol{R}$, $\boldsymbol{\omega}$, $\dot{\boldsymbol{\omega}}$, $\boldsymbol{\mathcal{I}}$, $\boldsymbol{\mathcal{L}}$, $\dot{\boldsymbol{\mathcal{L}}}$, and $\boldsymbol{f}_i$'s are time-varying, while $\boldsymbol{r}_i$'s also change during locomotion but they change intermittently and can be deemed constant over the finite-length prediction horizon of the MPC.

The contact force $\boldsymbol{f}_i \in \mathbb{R}^3$ should satisfy the constraint

$$\boldsymbol{N}_i^T\boldsymbol{f}_i \le \boldsymbol{b}_i \tag{5}$$

where $\boldsymbol{N}_i = -[\mu_i\boldsymbol{n}_i - \boldsymbol{o}_i \quad \mu_i\boldsymbol{n}_i + \boldsymbol{o}_i \quad \mu_i\boldsymbol{n}_i - \boldsymbol{t}_i \quad \mu_i\boldsymbol{n}_i + \boldsymbol{t}_i \quad \boldsymbol{n}_i \quad -\boldsymbol{n}_i] \in \mathbb{R}^{3\times 6}$, $\boldsymbol{b}_i = [0 \quad 0 \quad 0 \quad 0 \quad -f_i^L \quad f_i^U]^T$, μ_i is the coefficient of friction, $\boldsymbol{n}_i, \boldsymbol{o}_i, \boldsymbol{t}_i \in \mathbb{R}^3$ are the unit normal and two orthonormal tangent vectors described with respect to the global frame, and f_i^L and f_i^U are the nonnegative lower and upper bounds on the normal contact force, respectively. The first four linear inequalities given in (5) represent the friction constraint, which defines a pyramidal cone limiting the direction of $\boldsymbol{f}_i$, while the last two linear inequalities further limit the magnitude of $\boldsymbol{f}_i$.

III. Model-Predictive Control

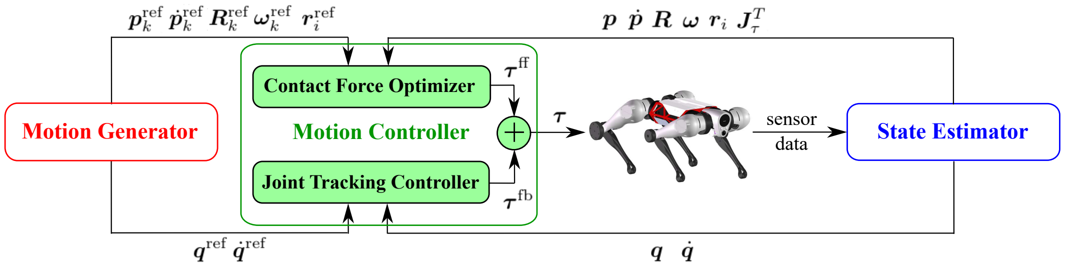
The control framework of our robot is depicted in Fig. 2. Taking a reference motion generated by the motion generator and the current robot state acquired by its on-board sensors as inputs, the motion controller computes the required joint torques for the robot to execute the reference motion. Similarly to many existing controllers for legged robots [27], [41], [28], [29], [4], [36], [9], [42], [10], [22], the joint torques $\boldsymbol{\tau}$ used to control our robot consist of a feed-back component $\boldsymbol{\tau}^{\text{fb}}$ for tracking every joint trajectory and a feed-forward component $\boldsymbol{\tau}^{\text{ff}}$ for realizing the reference body motion, i.e.,

$$\boldsymbol{\tau} = \boldsymbol{\tau}^{\text{fb}} + \boldsymbol{\tau}^{\text{ff}}. \tag{6}$$

The component $\boldsymbol{\tau}^{\text{fb}}$ is obtained by the PD control law:

$$\boldsymbol{\tau}^{\text{fb}} = \boldsymbol{K}_p^q\left(\boldsymbol{q}^{\text{ref}} - \boldsymbol{q}\right) + \boldsymbol{K}_d^q\left(\dot{\boldsymbol{q}}^{\text{ref}} - \dot{\boldsymbol{q}}\right) \tag{7}$$

where $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ with or without the superscript "ref" are the reference or current joint angles and velocities, respectively, and $\boldsymbol{K}_p^q$ and $\boldsymbol{K}_d^q$ are the PD gains.



$\boldsymbol{p}_k^{\text{ref}}\ \dot{\boldsymbol{p}}_k^{\text{ref}}\ \boldsymbol{R}_k^{\text{ref}}\ \boldsymbol{\omega}_k^{\text{ref}}\ \boldsymbol{r}_i^{\text{ref}}$
$\boldsymbol{p}\ \dot{\boldsymbol{p}}\ \boldsymbol{R}\ \boldsymbol{\omega}\ \boldsymbol{r}_i\ \boldsymbol{J}_\tau^T$
Contact Force Optimizer
$\boldsymbol{\tau}^{\text{ff}}$
Motion Generator
Motion Controller
$\boldsymbol{\tau}$
sensor
data
State Estimator
Joint Tracking Controller
$\boldsymbol{\tau}^{\text{fb}}$
$\boldsymbol{q}^{\text{ref}}\ \dot{\boldsymbol{q}}^{\text{ref}}$
$\boldsymbol{q}\ \ \dot{\boldsymbol{q}}$

Fig. 2. Control framework of our quadruped robot. Based on the sensor data including inertia measurement unit (IMU), joint encoder and torque measurement, the position $\boldsymbol{p}$ and velocity $\dot{\boldsymbol{p}}$ of the robot's CoM and the orientation $\boldsymbol{R}$ and angular velocity $\boldsymbol{\omega}$ of its floating base as well as its current contacts $\boldsymbol{r}_i$ with the environment are estimated. The reference body and joint motions are produced by a motion generator. Based on the robot's centroidal dynamics, the optimal contact forces for realizing the reference body motion are calculated and converted to the feed-forward joint torque $\boldsymbol{\tau}^{\text{ff}}$, which is added to the feed-back joint torque $\boldsymbol{\tau}^{\text{fb}}$ determined by a joint PD tracking controller to form the final control command $\boldsymbol{\tau}$ for the robot.

From the contact forces $\boldsymbol{f}^{\text{opt}} = [\boldsymbol{f}_1^T \ \ \boldsymbol{f}_2^T \ \ \cdots \ \ \boldsymbol{f}_N^T]^T \in \mathbb{R}^{3N}$ required to realize the body motion, the component $\boldsymbol{\tau}^{\text{ff}}$ can be computed through whole-body inverse dynamics or simply as below since the robot's legs are considered to be massless and move relatively slowly in stance,

$$\boldsymbol{\tau}^{\text{ff}} = -\boldsymbol{J}_\tau^T \boldsymbol{f}^{\text{opt}} \tag{8}$$

where $\boldsymbol{J}_\tau \in \mathbb{R}^{3N\times N_J}$ is the Jacobian matrix. The rest of this section discusses an MPC method based on a reduced-loss linearization of the centroidal dynamics (4) to compute $\boldsymbol{f}^{\text{opt}}$.

A. Model Linearization

Similarly to the existing MPC methods [35], [36], a linear relation between the robot's state and control inputs needs to be derived first, which is the goal of this subsection. Let K be the horizon length for the MPC problem, which is the number of predicted times, and ΔT the time step between two successive times. At time k ($k = 0, 1, \ldots, K$), let $\boldsymbol{f}_{i,k} \in \mathbb{R}^3$ be the force at contact i and $\boldsymbol{f}_k = [\boldsymbol{f}_{1,k}^T \ \ \boldsymbol{f}_{2,k}^T \ \ \cdots \ \ \boldsymbol{f}_{N_k,k}^T]^T \in \mathbb{R}^{3N_k}$ the concatenation of all the contact forces, where N_k is the number of contacts between the robot and the environment. Furthermore, let $\boldsymbol{f} \in \mathbb{R}^C$ be the concatenation of all the contact forces over the horizon, which provide all the control inputs of the MPC, where $C = 3\sum_{k=0}^{K-1} N_k$.

1) Linear Motion: From (4a) it follows that the relation between the linear acceleration of the robot's base and the contact forces $\boldsymbol{f}_{i,k}$ is naturally linear and it at time $k = 0, 1, \ldots, K-1$ can be rewritten as

$$\ddot{\boldsymbol{p}}_k = \boldsymbol{A}_{\ddot{p},k}\boldsymbol{f} + \boldsymbol{b}_{\ddot{p},k} \tag{9}$$

where the blocks of $\boldsymbol{A}_{\ddot{p},k} \in \mathbb{R}^{3\times C}$ corresponding to $\boldsymbol{f}_{i,k}$ for $i = 1, 2, \ldots, N_k$ are the 3-by-3 identity matrices divided by m and the other entries are all zero, and $\boldsymbol{b}_{\ddot{p},k} = \boldsymbol{g}$. Simply using the forward Euler method, we obtain the linear velocity $\dot{\boldsymbol{p}}$ and position $\boldsymbol{p}$ of the robot at time $k = 1, 2, \ldots, K$ as

$$\dot{\boldsymbol{p}}_k = \Delta T \sum_{j=0}^{k-1} \ddot{\boldsymbol{p}}_j + \dot{\boldsymbol{p}}_0 = \boldsymbol{A}_{\dot{p},k}\boldsymbol{f} + \boldsymbol{b}_{\dot{p},k} \tag{10}$$

$$\boldsymbol{p}_k = \Delta T^2 \sum_{j=0}^{k-1} (k - j - \frac{1}{2})\ddot{\boldsymbol{p}}_j + k\Delta T \dot{\boldsymbol{p}}_0 + \boldsymbol{p}_0 = \boldsymbol{A}_{p,k}\boldsymbol{f} + \boldsymbol{b}_{p,k} \tag{11}$$

where $\boldsymbol{A}_{\dot{p},k}$, $\boldsymbol{b}_{\dot{p},k}$ and $\boldsymbol{A}_{p,k}$, $\boldsymbol{b}_{p,k}$ are derived by substituting (9) into (10) and (11), respectively, and $\boldsymbol{b}_{\dot{p},k} = k\Delta T \boldsymbol{g} + \dot{\boldsymbol{p}}_0$ and $\boldsymbol{b}_{p,k} = \frac{1}{2}(k\Delta T)^2 \boldsymbol{g} + k\Delta T \dot{\boldsymbol{p}}_0 + \boldsymbol{p}_0$.

2) Angular Motion: Compared with the linear motion, from (3) and (4b) the equation of angular motion contains several nonlinear terms due to the products of time-varying variables and needs to be linearized for deriving a linear MPC. A simple way to do this is omitting the centrifugal term $\hat{\boldsymbol{\omega}}\boldsymbol{\mathcal{I}}\boldsymbol{\omega}$ in (3) by assuming that the angular velocity $\boldsymbol{\omega}$ is small and taking $\boldsymbol{\mathcal{I}}$ and $\boldsymbol{p}$ constantly to be their values at time $k = 0$ [35], [36], which are denoted by $\boldsymbol{\mathcal{I}}_0$ and $\boldsymbol{p}_0$, respectively. However, this linearization is excessive and may be unsuited for violent robot motions. Below, we derive a linear description of the angular motion by omitting fewer nonlinear terms in (4b). Letting $\Delta \boldsymbol{p}_k = \boldsymbol{p}_k - \boldsymbol{p}_0$ and uniting (4a), we first rewrite (4b) at time k as

$$\sum_{i=1}^{N} (\hat{\boldsymbol{r}}_i - \hat{\boldsymbol{p}}_0)\boldsymbol{f}_{i,k} = m(\hat{\boldsymbol{g}} - \hat{\ddot{\boldsymbol{p}}}_k)\Delta \boldsymbol{p}_k + \dot{\boldsymbol{\mathcal{L}}}_k. \tag{12}$$

Instead of omitting the whole term $m(\hat{\boldsymbol{g}} - \hat{\ddot{\boldsymbol{p}}}_k)\Delta \boldsymbol{p}_k$ as in the previous work [35], [36], we just omit $m\hat{\ddot{\boldsymbol{p}}}_k \Delta \boldsymbol{p}_k$ but keep $m\hat{\boldsymbol{g}}\Delta \boldsymbol{p}_k$, which is linear with the control input $\boldsymbol{f}$ from (11).

Prior to linearizing $\dot{\boldsymbol{\mathcal{L}}}_k$, we seek a linear representation of the change in the robot's orientation. Let $\boldsymbol{\theta}_{k,k-1} \in \mathbb{R}^3$ denote the change in the robot's orientation from time $k-1$ to time k such that $\boldsymbol{R}_k = e^{\hat{\boldsymbol{\theta}}_{k,k-1}} \boldsymbol{R}_{k-1}$, where $\hat{\boldsymbol{\theta}}_{k,k-1} \in so(3)$ is known as the exponential coordinates for rotation [43]. The exponential of $\hat{\boldsymbol{\theta}}_{k,k-1}$, denoted by $e^{\hat{\boldsymbol{\theta}}_{k,k-1}} \in SO(3)$, can be calculated by the power series as

$$e^{\hat{\boldsymbol{\theta}}_{k,k-1}} = \boldsymbol{I} + \hat{\boldsymbol{\theta}}_{k,k-1} + \frac{1}{2!}\hat{\boldsymbol{\theta}}_{k,k-1}^2 + \cdots \tag{13}$$

where $\boldsymbol{I} \in \mathbb{R}^{3\times 3}$ is the identity matrix. When $\boldsymbol{\theta}_{k,k-1}$ is small, like in the MPC, the nonlinear terms in the above equation can be omitted and the exponential can be approximated by

$$e^{\hat{\boldsymbol{\theta}}_{k,k-1}} \approx \boldsymbol{I} + \hat{\boldsymbol{\theta}}_{k,k-1}. \tag{14}$$

Furthermore, we can derive

$$e^{\hat{\boldsymbol{\theta}}_{k+1,k}} e^{\hat{\boldsymbol{\theta}}_{k,k-1}} \approx \boldsymbol{I} + \hat{\boldsymbol{\theta}}_{k+1,k} + \hat{\boldsymbol{\theta}}_{k,k-1}. \tag{15}$$

This means that the change in the robot's orientation from time $k-1$ to $k+1$, namely from $\boldsymbol{R}_{k-1}$ to $\boldsymbol{R}_{k+1}$, can be reduced to $\boldsymbol{\theta}_{k+1,k} + \boldsymbol{\theta}_{k,k-1}$. Hence, $\boldsymbol{R}_k$ can be written as

$$\boldsymbol{R}_k \approx (\boldsymbol{I} + \Delta\hat{\boldsymbol{\theta}}_k)\boldsymbol{R}_0 \tag{16}$$

where $\Delta\boldsymbol{\theta}_k$ is the approximated exponential coordinates for $\boldsymbol{R}_k\boldsymbol{R}_0^T$ and can be calculated by

$$\Delta\boldsymbol{\theta}_k \approx \sum_{j=1}^{k} \boldsymbol{\theta}_{j,j-1}. \tag{17}$$

Again, by the forward Euler method, the angular velocity $\boldsymbol{\omega}_k$ of the robot can be written as

$$\boldsymbol{\omega}_k = \boldsymbol{\omega}_{k-1} + \dot{\boldsymbol{\omega}}_{k-1}\Delta T = \Delta T \sum_{j=0}^{k-1} \dot{\boldsymbol{\omega}}_j + \boldsymbol{\omega}_0. \tag{18}$$

Then, the velocity change from the beginning to time k, denoted by $\Delta\boldsymbol{\omega}_k$, is given by

$$\Delta\boldsymbol{\omega}_k = \boldsymbol{\omega}_k - \boldsymbol{\omega}_0 = \Delta T \sum_{j=0}^{k-1} \dot{\boldsymbol{\omega}}_j. \tag{19}$$

In addition, $\boldsymbol{\theta}_{k,k-1}$ can be calculated by

$$\boldsymbol{\theta}_{k,k-1} = \boldsymbol{\omega}_{k-1}\Delta T + \frac{1}{2}\dot{\boldsymbol{\omega}}_{k-1}\Delta T^2. \tag{20}$$

Substituting (18) and (20) into (17), we derive

$$\Delta\boldsymbol{\theta}_k = \Delta T^2 \sum_{j=0}^{k-1} (k - j - 0.5)\dot{\boldsymbol{\omega}}_j + k\Delta T \boldsymbol{\omega}_0. \tag{21}$$

Substituting (16) into (2) and omitting the second-order term of $\Delta\boldsymbol{\theta}_k$, we can derive

$$\begin{aligned} \boldsymbol{\mathcal{I}}_k &= \boldsymbol{R}_k \boldsymbol{\mathcal{I}}_B \boldsymbol{R}_k^T \\ &= (\boldsymbol{I} + \Delta\hat{\boldsymbol{\theta}}_k)\boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_B \boldsymbol{R}_0^T (\boldsymbol{I} - \Delta\hat{\boldsymbol{\theta}}_k) \\ &\approx \boldsymbol{\mathcal{I}}_0 + \Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k \end{aligned} \tag{22}$$

where $\boldsymbol{\mathcal{I}}_0 = \boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_B \boldsymbol{R}_0^T$ is the spatial inertia tensor at time $k = 0$. Then, substituting (22) with (19) into (3) and omitting the nonlinear terms, we can further derive

$$\begin{aligned} \dot{\boldsymbol{\mathcal{L}}}_k &= (\boldsymbol{\mathcal{I}}_0 + \Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k + \\ &\quad (\hat{\boldsymbol{\omega}}_0 + \Delta\hat{\boldsymbol{\omega}}_k)((\boldsymbol{\mathcal{I}}_0 + \Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)(\boldsymbol{\omega}_0 + \Delta\boldsymbol{\omega}_k)) \\ &\approx \boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_k + (\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k + \boldsymbol{Q}\Delta\boldsymbol{\omega}_k - \boldsymbol{P}\Delta\boldsymbol{\theta}_k \\ &\quad + \hat{\boldsymbol{\omega}}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0 \end{aligned} \tag{23}$$

where $\boldsymbol{Q} = \hat{\boldsymbol{\omega}}_0 \boldsymbol{\mathcal{I}}_0 - (\widehat{\boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0})$ and $\boldsymbol{P} = \hat{\boldsymbol{\omega}}_0((\widehat{\boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0}) - \boldsymbol{\mathcal{I}}_0 \hat{\boldsymbol{\omega}}_0) = \hat{\boldsymbol{\omega}}_0 \boldsymbol{Q}^T$. One interesting term in (23) is $(\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k$, which has to be omitted in order for the final relation of $\dot{\boldsymbol{\omega}}_k$ to $\boldsymbol{f}$ to be linear. From (2) we have $\boldsymbol{\mathcal{I}}_0 = \boldsymbol{R}_0 \boldsymbol{\mathcal{I}}_B \boldsymbol{R}_0^T$, where $\boldsymbol{\mathcal{I}}_B = \mathrm{diag}(\mathcal{I}_\mathrm{x}, \mathcal{I}_\mathrm{y}, \mathcal{I}_\mathrm{z})$ is assumed to be diagonal without loss of generality and $\mathcal{I}_x, \mathcal{I}_y, \mathcal{I}_z$ are the principal moments of inertia. Ideally, when $\mathcal{I}_x, \mathcal{I}_y, \mathcal{I}_z$ are equal, $\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k$ is zero and so is $(\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k$. In another case where the robot body rotates only about one of the x, y, and z axes of frame F_body from the beginning, which are given by the three columns of $\boldsymbol{R}_0 = [\boldsymbol{r}_x \;\; \boldsymbol{r}_y \;\; \boldsymbol{r}_z]$, we can also derive $(\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k = \boldsymbol{0}$. For example, assume that $\Delta\boldsymbol{\theta}_k = \alpha\boldsymbol{r}_x$ and $\dot{\boldsymbol{\omega}}_k = \beta\boldsymbol{r}_x$. Then, we can derive

$$\begin{aligned} (\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k &= \alpha\hat{\boldsymbol{r}}_x \boldsymbol{\mathcal{I}}_0 \beta\boldsymbol{r}_x - \boldsymbol{\mathcal{I}}_0 \alpha\hat{\boldsymbol{r}}_x \beta\boldsymbol{r}_x \\ &= \alpha\beta[\boldsymbol{0} \;\; \boldsymbol{r}_z \;\; -\boldsymbol{r}_y]\boldsymbol{\mathcal{I}}_B[1 \;\; 0 \;\; 0]^T \\ &= \alpha\beta[\boldsymbol{0} \;\; \boldsymbol{r}_z \;\; -\boldsymbol{r}_y][\mathcal{I}_x \;\; 0 \;\; 0]^T = \boldsymbol{0}. \end{aligned}$$

Therefore, it is relatively safe to omit $(\Delta\hat{\boldsymbol{\theta}}_k \boldsymbol{\mathcal{I}}_0 - \boldsymbol{\mathcal{I}}_0 \Delta\hat{\boldsymbol{\theta}}_k)\dot{\boldsymbol{\omega}}_k$. As a result, $\dot{\boldsymbol{\mathcal{L}}}_k$ can be simplified as

$$\dot{\boldsymbol{\mathcal{L}}}_k \approx \boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_k + \boldsymbol{Q}\Delta\boldsymbol{\omega}_k - \boldsymbol{P}\Delta\boldsymbol{\theta}_k + \hat{\boldsymbol{\omega}}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0. \tag{24}$$

Substituting (24) into (12) and omitting $m\hat{\ddot{\boldsymbol{p}}}_k \Delta\boldsymbol{p}_k$, we have

$$\begin{aligned} \sum_{i=1}^{N} (\hat{\boldsymbol{r}}_i - \hat{\boldsymbol{p}}_0)\boldsymbol{f}_{i,k} &\approx m\hat{\boldsymbol{g}}\Delta\boldsymbol{p}_k + \boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_k \\ &\quad + \boldsymbol{Q}\Delta\boldsymbol{\omega}_k - \boldsymbol{P}\Delta\boldsymbol{\theta}_k + \hat{\boldsymbol{\omega}}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0. \end{aligned} \tag{25}$$

Substituting (19) and (21) into (25) followed by some mathematical manipulation, we further derive

$$\begin{aligned} \boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_k &= \sum_{i=1}^{N} (\hat{\boldsymbol{r}}_i - \hat{\boldsymbol{p}}_0)\boldsymbol{f}_{i,k} \\ &\quad + \sum_{j=0}^{k-1} \begin{bmatrix} (\frac{2j+1}{2} - k)\Delta T^2 m\hat{\boldsymbol{g}}^T \\ (k - \frac{2j+1}{2})\Delta T^2 \boldsymbol{P}^T - \Delta T \boldsymbol{Q}^T \end{bmatrix}^T \begin{bmatrix} \ddot{\boldsymbol{p}}_j \\ \dot{\boldsymbol{\omega}}_j \end{bmatrix} \\ &\quad - mk\Delta T \hat{\boldsymbol{g}}\dot{\boldsymbol{p}}_0 - (\boldsymbol{I} + k\Delta T \hat{\boldsymbol{\omega}}_0)\hat{\boldsymbol{\omega}}_0 \boldsymbol{\mathcal{I}}_0 \boldsymbol{\omega}_0. \end{aligned} \tag{26}$$

In comparison with the previous work [35], [36], which has $\boldsymbol{\mathcal{I}}_0 \dot{\boldsymbol{\omega}}_k = \sum_{i=1}^{N} (\hat{\boldsymbol{r}}_i - \hat{\boldsymbol{p}}_0)\boldsymbol{f}_{i,k}$, we retain more linear terms in (26). Uniting (9), (26) gives a recursive way to compute the relation of $\dot{\boldsymbol{\omega}}_k$ to $\boldsymbol{f}$, which is linear and can be written as

$$\dot{\boldsymbol{\omega}}_k = \boldsymbol{A}_{\dot{\omega},k}\boldsymbol{f} + \boldsymbol{b}_{\dot{\omega},k} \tag{27}$$

where $\boldsymbol{A}_{\dot{\omega},k} \in \mathbb{R}^{3\times C}$ and $\boldsymbol{b}_{\dot{\omega},k} \in \mathbb{R}^3$ depend only on the initial state of the robot, the prediction horizon, and the time step and they can be calculated by (26). Substituting (27) into (18) and (21) yields the linear expressions of $\boldsymbol{\omega}_k$ and $\Delta\boldsymbol{\theta}_k$ with respect to $\boldsymbol{f}$, respectively, i.e.,

$$\boldsymbol{\omega}_k = \boldsymbol{A}_{\omega,k}\boldsymbol{f} + \boldsymbol{b}_{\omega,k} \tag{28}$$

$$\Delta\boldsymbol{\theta}_k = \boldsymbol{A}_{\theta,k}\boldsymbol{f} + \boldsymbol{b}_{\theta,k}. \tag{29}$$

Due to the space limit and the complexity, the detailed forms of matrices $\boldsymbol{A}_{\dot{\omega},k}$, $\boldsymbol{A}_{\omega,k}$, $\boldsymbol{A}_{\theta,k}$ and vectors $\boldsymbol{b}_{\dot{\omega},k}$, $\boldsymbol{b}_{\omega,k}$, $\boldsymbol{b}_{\theta,k}$ in the above equations are not provided here.

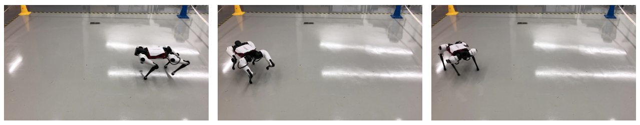
B. QP Formulation

Like many other MPC problems, with the above linear relations between the robot's state and control inputs we write our MPC controller as a QP problem:

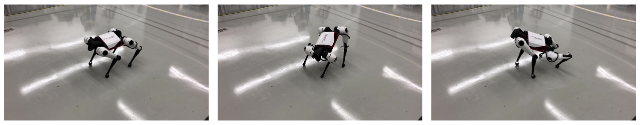
$$\begin{cases} \text{minimize} & \sum_{k=1}^{K} \|\boldsymbol{e}_k\|^2_{\boldsymbol{W}_k} + \|\boldsymbol{f}\|^2_{\boldsymbol{W}_f} \\ \text{subject to} & (5) \text{ for all } i, k \end{cases} \tag{30}$$

where $\|\boldsymbol{e}_k\|_{\boldsymbol{W}_k}$ is the weighted norm of the tracking error $\boldsymbol{e}_k$ of the robot's state at time k and $\|\boldsymbol{f}\|_{\boldsymbol{W}_f}$ is the weighted norm of the control input $\boldsymbol{f}$. The state error $\boldsymbol{e}_k$ is defined as

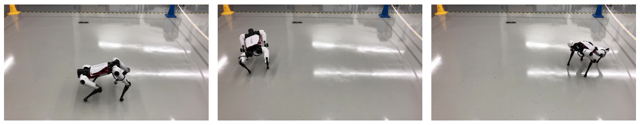
$$\boldsymbol{e}_k = \begin{bmatrix} \boldsymbol{p}_k^\mathrm{ref} - \boldsymbol{p}_k \\ \dot{\boldsymbol{p}}_k^\mathrm{ref} - \dot{\boldsymbol{p}}_k \\ \Delta\boldsymbol{\theta}_k^\mathrm{ref} - \Delta\boldsymbol{\theta}_k \\ \boldsymbol{\omega}_k^\mathrm{ref} - \boldsymbol{\omega}_k \end{bmatrix} \in \mathbb{R}^{12} \tag{31}$$



(a)



(b)



(c)

Fig. 3. Experiments on our quadruped robot. (a) Trotting forward/backward and turning. (b) Attitude tracking while standing. (c) Attitude tracking while trotting in a circle. Realized motions are shown in the accompanying video.

TABLE I. Model and Control Parameters

m (kg)	$\boldsymbol{I}_{xx}$ (kgm^2)	$\boldsymbol{I}_{yy}$ (kgm^2)	$\boldsymbol{I}_{zz}$ (kgm^2)	μ
12.4	0.067	0.272	0.324	0.5
$\boldsymbol{W}_p$	$\boldsymbol{W}_{\dot{p}}$	$\boldsymbol{W}_\theta$	$\boldsymbol{W}_\omega$	$\boldsymbol{W}_f$
[10, 10, 12]	[0.05, 0.05, 0.2]	[2, 2, 4]	[0, 0, 0.2]	4×10^{-5}

where $\Delta\boldsymbol{\theta}_k^{\text{ref}} = (\boldsymbol{R}_k^{\text{ref}}\boldsymbol{R}_0^T)^\vee$ and $\vee$ represents the inverse of the operator $\wedge$ to convert a skew-symmetric matrix to the corresponding vector in $\mathbb{R}^3$. From (10), (11), (28), and (29), $\boldsymbol{e}_k$ is linear with $\boldsymbol{f}$ and the objective function of (30) is a quadratic function of $\boldsymbol{f}$. Among several off-the-shelf solvers for QP problems, we use qpOASES [44] in this paper.

IV. Experiments

Here we test the proposed MPC method on our quadruped robot, as shown in Fig. 3 and the accompanying video.

A. Experimental Setup

The proposed MPC method has been implemented on our robot and examined in three different locomotion scenarios, namely trotting in straight lines and turning around, attitude tracking while standing, and attitude tracking while trotting in a circle. To better evaluate the tracking performance of the proposed method, the robot's state estimator integrates the motion capture data of high accuracy with the IMU data of high frequency and uses a low-pass filter to reduce the high frequency noise in the estimated velocity. Similarly to the work [4], a heuristic online foot trajectory planner was implemented, which calculates the foot trajectories based on the actual position, attitude, and velocity of the robot. A leg impedance controller was implemented as well. The state estimator, the foot trajectory planner, and the leg impedance controller run at 500 Hz, while the MPC prediction happens at 100 Hz. The prediction window length of the controller is set to 10, totaling 0.5 s which is equal to the duration of a full gait cycle, except for trotting in straight lines in which case it is set to 5 to achieve a better tracking performance.

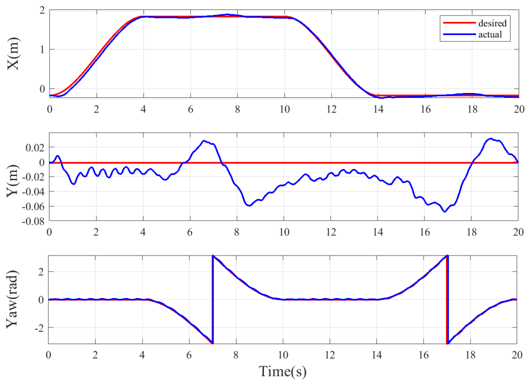


Fig. 4. X, Y, and yaw of the robot during trotting in a straight line. Curves in the red and blue colors represent the desired reference trajectories and the realized trajectories estimated by the robot's state estimator, respectively.

The reference trajectories of the robot's CoM and attitude were generated offline so that results of repetitive trials on the same locomotion can be compared and thorough comparison with existing control methods can also be made. Other model and control parameters are listed in Table I.

B. Straight Trotting and Turning

The first experiment is intended to verify the performance of our MPC method in realizing the normal trotting gait, as shown in Fig. 3a and the accompanying video. We require the robot to trot forward for 2 m, turn around clockwise, trot backward to the starting point, and finally turn around counterclockwise. The reference roll and pitch for the robot are set to zero during this experiment.

Figure 4 shows the tracking performance of our MPC method in x, y, and yaw, respectively. The maximum tracking error in x direction is 0.109 m, which is reached during the accelerating phase, while it is 0.068 m in y direction during the turning phase. It makes sense that starting tracking from the standing state may delay the acceleration response. It can be seen that the error in x is gradually reduced during forward trotting and maintained on a minimum level afterwards. The mean tracking errors in x and y directions are 0.035 m and 0.023 m, respectively. Tracking of yaw angle has a maximum error of $4.5°$. The discontinuity in yaw is simply due to angle wrapping at $\pm\pi$. Overall, our MPC controller enables the robot to track the desired trajectories with a good accuracy. Note that the tracking performance is also subject to the parameter tuning of the controller and could be further improved with finer parameter tuning.

Our MPC method takes less than 0.18 ms every time and averagely 0.089 ms during the experiment, where 83.5% of its computations (1859 out of 2225) are less than 0.1 ms. Thus, it can be readily applied to the real-time control of quadruped robots at 1 kHz. The average computation time of the convex MPC algorithm [36] under the same parameter setting for the same locomotion is about 0.097 ms, showing that our MPC method achieves comparable efficiency despite of its higher model complexity.

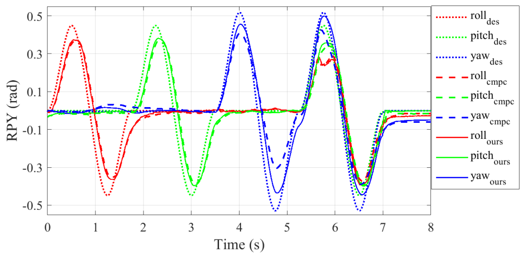


Fig. 5. Roll (red), pitch (green), and yaw (blue) of the robot during standing and tracking reference angular motions. The reference trajectories and the realized ones by the convex MPC [36] and our method are plotted in the dotted, dashed, and solid curves, respectively.

C. Attitude Tracking while Standing

As the linear model derived for our MPC method retains more terms contributing to the robot's angular motion, its performance in robot attitude control is further verified by the case where the robot is required to follow a reference attitude trajectory while standing, as shown in Fig. 3b. It is compared with the convex MPC algorithm [36], which assumes zero roll and pitch angles. The parameters of the convex MPC are tuned with the same technique as used for our controller to obtain the best overall performance in tracking position and Euler angles at the same time. The reference attitude trajectory is generated offline and comprises sequential individual roll, pitch, and yaw oscillations followed by a mixed oscillation in the three directions. The limits of roll and pitch are $\pm\frac{\pi}{7}$ and that of yaw is $\pm\frac{\pi}{6}$. Figure 5 depicts the realized trajectories by the two controllers in comparison with the reference. It can be seen that the performance of the two controllers in tracking individual roll and pitch oscillations is very close, but ours can better track the individual yaw and mixed oscillations with a faster tracking response and smaller errors in the peak and valley values. For example, the error in tracking the yaw oscillation at the valley by our controller is 0.088 rad, while that by the convex MPC controller is 0.22 rad. The results show that retaining extra linear terms in the dynamics model helps the derived controller achieve a better attitude control of the robot.

D. Attitude Tracking while Trotting

To further test our MPC controller in dynamic motions, we let the robot trot in a circle of radius 1 m while imposing a periodic mixed oscillation of roll and pitch (see Fig. 3c). The periods of both roll and pitch oscillations are 2 s and the magnitudes are $\frac{\pi}{10}$. Figures 6 and 7 show the tracking results in time domain and in xy plane, respectively. It can be seen that the mean tracking errors of the proposed controller in x and y are both around 0.1 m with a response delay, while the convex MPC outperforms ours with a mean error of 0.05 m. However, the peak and valley position values achieved by our controller agree well with the reference values. Consequently, our controller tracks the 2-D trajectory in xy plane with a smaller spatial error (see Fig. 7). Meanwhile, our controller tracks the pitch angle with a maximum error of 0.13 rad, compared with 0.17 rad by the convex MPC. Differences in

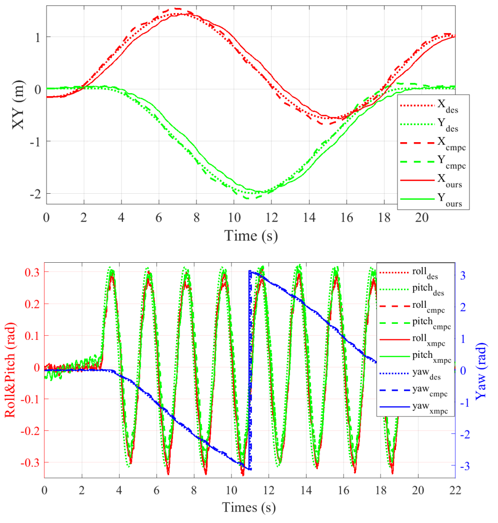


Fig. 6. Position (upper) and attitude (lower) of the robot during trotting in a circle and tracking reference angular motions.

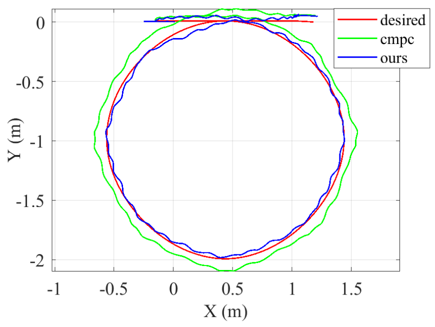


Fig. 7. Planar trajectory of the robot during trotting in a circle and tracking reference angular motions.

tracking performance of roll and yaw angles between the two controllers are trivial. The reasons could be that the magnitudes of the reference roll, pitch and the corresponding angular velocities are not large enough due to the physical joint limits of our robot. Nevertheless, comparable tracking performance with the convex MPC validates the effectiveness of the proposed MPC method. Besides, a slight oscillation with a period of 2 s is observed in yaw, implying a coupling effect with roll and pitch oscillations. As the yaw tracking performance is not debased compared with trotting in straight line, it verifies the robustness of our method to some extent.

V. Conclusion

In this paper, we propose a new linearization of the robot's centroidal dynamics, which identifies and retains more linear terms, leading to a more accurate linear dynamics model for deriving a model-predictive controller. Experiments with three locomotion scenarios have been conducted on our robot and shown that the MPC controller is competent for tracking both translational and rotational motions and the tracking error in the robot's attitude is small, which benefits from the more accurate model linearization. Besides the motion controller, the linear model can also be used in the locomotion generation, which we will explore in the future.

REFERENCES

[1] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "BigDog, the rough-terrain quadruped robot," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 10 822–10 825, 2008.

[2] 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*, vol. 225, no. 6, pp. 831–849, 2011.

[3] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "ANYmal – a highly mobile and dynamic quadrupedal robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Daejeon, South Korea, October 2016, pp. 38–44.

[4] G. Bledt, M. J. Powell, B. Katz, J. Di 'Carlo, P. M. Wensing, and S. Kim, "MIT Cheetah 3: Design and control of a robust, dynamic quadruped robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, October 2018, pp. 2245–2252.

[5] DeepRobotics, 2017. [Online]. Available: http://www.deeprobotics.cn/

[6] Unitree, 2017. [Online]. Available: http://www.unitree.cc/

[7] K. Yamane and J. Hodgins, "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, St. Louis, 2009, pp. 2510–2517.

[8] ——, "Control-aware mappping of human motion data with stepping for humanoid robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, 2010, pp. 726–733.

[9] S. Y. Feng, E. Whitman, X. Xinjilefu, and C. Atkeson, "Optimization-based full body control for the DARPA robotics challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[10] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. K. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the Atlas humanoid robot," *Autonomous Robots*, vol. 40, no. 3, pp. 429–455, 2016.

[11] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, CA, September 2011, pp. 4414–4419.

[12] K. Bouyarmane, J. Vaillant, F. Keith, and A. Kheddar, "Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Osaka, Japan, 2012, pp. 337–342.

[13] P. M. Wensing and D. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, Karlsruhe, Germany, May 2013, pp. 3103–3109.

[14] ——, "High-speed humanoid running through control with a 3D-SLIP model," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Tokyo, Japan, November 2013, pp. 5134–5140.

[15] L. Sentis and O. Khatib, "Control of free-floating humanoid robots through task prioritization," in *Proc. IEEE Int. Conf. Robot. Automat.*, Barcelona, Spain, April 2005, pp. 1718–1723.

[16] F. Aghili, "A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: Applications to control and simulation," *IEEE Trans. Robot.*, vol. 21, no. 5, pp. 834–849, 2005.

[17] M. Mistry, J. Buchli, and S. Schaal, "Inverse dynamics control of floating base systems using orthogonal decomposition," in *Proc. IEEE Int. Conf. Robot. Automat.*, Anchorage, Alaska, 2010.

[18] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Inverse dynamics control of floating-base robots with external constraints: A unified view," in *Proc. IEEE Int. Conf. Robot. Automat.*, Shanghai, China, 2011, pp. 1085–1090.

[19] N. Mansard, "A dedicated solver for fast operational-space inverse dynamics," in *Proc. IEEE Int. Conf. Robot. Automat.*, Saint Paul, Minnesota, May 2012, pp. 4943–4949.

[20] L. Righetti, J. Buchli, M. Mistry, M. Kalakrishnan, and S. Schaal, "Optimal distribution of contact forces with inverse-dynamics control," *International Journal of Robotics Research*, vol. 32, no. 3, pp. 280–298, 2013.

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

[22] Y. Zheng, S. W. Liao, and K. Yamane, "Humanoid locomotion control and generation based on contact wrench cones," *International Jounral of Humanoid Robotics*, vol. 16, no. 5, p. 1950021, 2019.

[23] S.-H. Lee and A. Goswami, "Ground reaction force control at each foot: A momentum-based humanoid balance controller for non-level and non-stationary ground," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, October 2010, pp. 3157–3162.

[24] C. Ott, M. A. Roa, and G. Hirzinger, "Posture and balance control for biped robots based on contact force optimization," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Bled, Slovenia, 2011, pp. 26–33.

[25] T. Koolen, J. Smith, G. Thomas, S. Bertrand, J. Carff, N. Mertins, D. Stephen, P. Abeles, J. Englsberger, S. McCrory, J. van Egmond, M. Griffioen, M. Floyd, S. Kobus, N. Manor, S. Alsheikh, D. Duran, L. Bunch, E. Morphis, L. Colasanto, K.-L. Ho Hoang, B. Layton, P. Neuhaus, M. Johnson, and J. Pratt, "Summary of Team IHMC's Virtual Robotics Challenge entry," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Atlanta, GA, October 2013, pp. 307–314.

[26] B. Henze, M. A. Roa, and C. Ott, "Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios," *International Journal of Robotics Research*, vol. 35, no. 12, pp. 1522–1543, 2016.

[27] C. Gehring, S. Coros, M. Hutter, M. Bloesch, M. A. Hoepflinger, and R. Siegwart, "Control of dynamic gaits for a quadrupedal robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 2013, pp. 3287–3292.

[28] M. Focchi, A. del Prete, I. Havoutis, R. Featherstone, D. G. Caldwell, and C. Semini, "High-slope terrain locomotion for torque-controlled quadruped robots," *Auton. Robots*, vol. 41, no. 1, pp. 259–272, 2017.

[29] C. Mastalli, M. Focchi, I. Havoutis, A. Radulescu, S. Calinon, J. Buchli, D. G. Caldwell, and C. Semini, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," in *Proc. IEEE Int. Conf. Robot. Automat.*, Singapore, May 2017, pp. 1096–1103.

[30] O. E. Ramos, N. Mansard, O. Stasse, and P. Souères, "Walking on non-planar surfaces using an inverse dynamic stack of tasks," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Osaka, Japan, November 2012, pp. 829–834.

[31] Y. Zheng and K. Yamane, "Human motion tracking control with strict contact force constraints for floating-base humanoid robots," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, Atlanta, GA, 2013, pp. 34–41.

[32] M. X. Liu, R. Lober, and V. Padois, "Whole-body hierarchical motion and force control for humanoid robots," *Autonomous Robots*, vol. 40, no. 3, pp. 493–504, 2016.

[33] A. Herzog, S. Rotella, N. amd Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.

[34] B. Henze, C. Ott, and M. A. Roa, "Posture and balance control for humanoid robots in multi-contact scenarios based on model predictive control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Chicago, IL, 2014, pp. 3253–3258.

[35] G. Bledt, P. M. Wensing, and S. Kim, "Policy-regularized model predictive control to stabilize diverse quadrupedal gaits for the MIT Cheetah," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, September 2017, pp. 4012–4019.

[36] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, October 2018, pp. 1–9.

[37] J. Carpentier and N. Mansard, "Multicontact locomotion of legged robots," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1441–1460, 2018.

[38] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback MPC for torque-controlled legged robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Macau, China, November 2019, pp. 4730–4737.

[39] M. Neunert, M. Stäuble, M. Giftthaler, C. D. Bellicoso, C. Gehring, M. Hutter, and J. Buchli, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1458–1465, 2018.

[40] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocoddyl: An effcient and versatile framework for multi-contact optimal control," in *Proc. IEEE Int. Conf. Robot. Automat.*, Paris, France, 2020, pp. 2536–2542.

[41] H.-W. Park, P. M. Wensing, and S. Kim, "Online planning for autonomous running jumps over obstacles in high-speed quadrupeds,"

in *Proceedings of Robotics: Science and Systems*, Rome, Italy, July 2015.

[42] S. Caron, Q.-C. Pham, and Y. Nakamura, "Leveraging cone double description for multi-contact stability of humanoids with applications to statics and dynamics," in *Robotics: Science and System*, 2015.

[43] R. M. Murray, Z. X. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL, USA: CRC Press, 1994.

[44] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Math. Prog. Comp.*, vol. 6, no. 4, pp. 327–363, 2014. [Online]. Available: https://github.com/coin-or/qpOASES