Robust Planning and Control for Dynamic Quadrupedal Locomotion with Adaptive Feet

Guiyang Xin[1], Songyan Xin[1], Oguzhan Cebe[1], Mathew Jose Pollayil[2], Franco Angelini[2], Manolo Garabini[2], Sethu Vijayakumar[1], Michael Mistry[1]

arXiv:2010.12326v1 [cs.RO] 23 Oct 2020

***Abstract*— In this paper, we aim to improve the robustness of dynamic quadrupedal locomotion through three aspects: 1) fast model predictive foothold planning, 2) LQR control for robust motion tracking and 3) adaptive feet for terrain adaptation. In our proposed planning and control framework, foothold plans are updated at 400 Hz considering the current robot state and an LQR controller generates optimal feedback gains for motion tracking. The LQR optimal gain matrix with non-zero off-diagonal elements leverages the coupling of dynamics to compensate for system underactuation, such as a quadruped robot with passive ankles. The specially designed foot with adaptive sole aims at improving the traversability of rough terrains with rocks, loose gravel and rubble by enlarging the contact surfaces with ground. Experiments on the quadruped ANYmal demonstrate the effectiveness of the proposed method for robust dynamic locomotion given external disturbances and environmental uncertainties.**

Fig. 1. ANYmal with adaptive feet stepping on rough terrain.

I. INTRODUCTION

Legged robots have evolved quickly in recent years. Although there are robots, such as Spot from Boston Dynamics, which have been deployed in real industrial scenarios, researchers continue to explore novel techniques to improve locomotion performance. A popular technique is the staged approach which divides the larger problem into sub-problems and chains them together. Typically the pipeline is composed of state estimation, planning and control, which may be running at different frequencies. The motion planner typically runs at a slower frequency comparing to controller due to model nonlinearities and long planning horizons. The lower-level feedback controller runs at a higher frequency to resist model discrepancies and external disturbances. After years of evolution, optimization becomes the core approach for motion planning and control of legged robots.

A. Related planning methods

Legged robot motion planning is a trade off between several criteria: formulation generality, model complexity, the planning horizon and computational efficiency. While the goal is to maximize all at once, this is not realistic given current available computational resources. As a result, different design choices lead to different formulations.

To generate motions in a more general and automated fashion, trajectory optimization (TO) has been used. In [1], a ZMP-based TO formulation is presented to optimize body motion, footholds and center of pressure simultaneously. It can generate different motion plans with multiple steps in less than a second. In a later work [2], a phase-based TO formulation is proposed to automatically determine the gait-sequence, step timings, footholds, body motion and contact forces. Motion for multiple steps can be still generated in few seconds. In these two works, the TO formulations are both extremely versatile in terms of motion types that can be generated, however, online Model Predictive Control (MPC) has not been demonstrated yet.

A linearized, single rigid-body model has been proposed [3][4] to formulate the ground reaction force as a QP optimization problem and which can be solved in an MPC fashion. In both works, the footstep locations are provided by simple heuristics. Online TO based on a nonlinear single rigid-body model has been given in [5], and can generate stable dynamic motion for quadruped robots based on a given contact sequence. A whole-body dynamic model has been considered in [6] to generate robot motion in a MPC fashion. Crocoddyl [7] improves the computation speed further more. In both works, the contact planning problem has been decoupled from the whole-body motion planning problem.

*This work is supported by the following grants: EPSRC UK RAI Hubs NCNR (EPR02572X/1), ORCA(EP/R026173/1), FAIR-SPACE (EP/R026092/1) and THING in the EU Horizon 2020 (ICT-2017-1 780883).

[1]Guiyang Xin, Songyan Xin, Oguzhan Cebe, Sethu Vijayakumar and Michael Mistry are with the School of Informatics, Institute of Perception, Action and Behaviour, University of Edinburgh, EH8 9AB, 10 Crichton Street, Edinburgh, United Kingdom guiyang.xin@ed.ac.uk

[2]Mathew Jose Pollayil, Franco Angelini and Manolo Garabini are with the Department of Information Engineering of the University of Pisa and with the Research Center "E. Piaggio" of the University of Pisa, Italy mathewjose.pollayil@phd.unipi.it

Footstep optimization on biped robots has been proposed in [8] [9]. An underactuated linear inverted pendulum model (LIPM) has been used to formulate the footstep optimization problem. The idea has been extended and generalized to both biped and quadruped robots in our previous work [10]. In this work, we realize real-time footstep optimization that can be executed in a MPC fashion and test it on the hardware ANYmal.

B. Related control methods

In recent years, there has been a convergence among legged robot researchers to formulate the control problem as a Quadratic Program (QP) with constraints. The problem can be further decomposed into hierarchies to coordinate multiple tasks within whole-body control [11][12][13][14]. These optimization-based controllers usually rely on manually tuned diagonal feedback gain matrices. Also, these controllers only compute the best commands for the next control cycle, and therefore are not suitable for dynamic gaits with underactuation. Classical optimal control theory, such as LQR, can consider long or even infinite time horizons and generate optimal non-diagonial gain matricies exploiting dynamic coupling effects which benefit underactuated systems such as a cart-pole [15].

Classical LQR does not consider any constraints except system dynamics. However, for legged robots, we have to satisfy inequality constraints such as torque limits and friction cones on contact feet. The works [16][17] proposed to use the classical LQR controller for bipedal walking, but inequality constraints were not enforced. In this paper, we propose an LQR controller for dynamic gait control under the framework of projected inverse dynamics [18][19]. Projected inverse dynamic control enables us to control motion in a constraint-free subspace while satisfying inequality constraints in an orthogonal subspace. In our previous work, we used Cartesian impedance controllers within the constraint-free subspace to control both the base and swing legs during static walking of a quadruped robot. Here, we use LQR in the constraint-free subspace to replace the Cartesian impedance controller for base motion control and to handle the underactuation in the trotting gait.

C. Contributions

This paper focuses on improving the robustness of dynamic quadrupedal gaits. The trotting gait of quadruped robots will be primarily studied and demonstrated in simulation and real experiments. The main contributions lie in the computation speed of the MPC and the optimal feedback control. As an additional contribution, our approach is shown to be valid both with spherical and adaptive feet with flexible soles. Our contributions are listed as follows:

1) We propose to formulate foothold planning as a QP problem subject to LIPM dynamics, which can be solved within the control cycle of 2.5 ms. Running re-planning at high frequency allows the robot to be responsive to disturbances and control commands. The higher the updating frequency of the MPC, the better the reactivity achieved by the robot.
2) We use unconstrained infinite-horizon LQR to generate optimal gains for base control in order to improve the robustness of the controller and cope with underactuation. Meanwhile, we use the projected inverse dynamic framework to satisfy the inequality constraints in an orthogonal subspace.
3) We validate through experiments that our approach aids locomotion both for robots using traditional spherical as well as novel adaptive feet with passive ankles. Additionally, we show that the use of adaptive feet with flexible soles improves foot adaptation on rough terrains, increases the foot-terrain contact surface, and reduces slippage.

D. Paper organization

The paper is organized in accordance with the hierarchical structure of the whole system, which is shown in Fig. 2. The top level user command is a simple reference velocity for the robot to track. It consists of three components: sagittal speed, lateral speed and angular yaw speed. Given these velocities, the foothold planner plans future footsteps based on the current robot state: this motion generation is explained in Section II. Only the first optimized footstep from MPC will be sent to the next layer, the trajectory generator. According to this, the desired CoM and swing trajectories will be updated and sent to the controller. Details about the MPC formulation and the Cartesian trajectory generation as well as the leg coordination mechanism for quadruped robots are given respectively in Sections II-A and II-B. Section III describes the derivation of the LQR for base control including the linearization of the dynamics and the entire control framework with constraint satisfaction. Details about the adaptive feet employed for experimental validation are presented in Section IV. Experiments and discussions are given in Section V. Finally, Section VI draws the relevant conclusions.

II. Motion generation

When considering dynamic gaits such as trotting, two contact points cannot supply enough constraints to the floating base. The system becomes underactuated as one DOF around the support line is not directly controlled. Researchers have been using the LIPM as an abstract model for balance control in this situation. The CoM position and velocity can be predicted by solving the forward dynamics of the passive inverted pendulum. In order keep long term balance, the next ZMP point has to be carefully selected to capture the falling CoM. For trotting, the ZMP point always lays on the support line formed by the supporting leg pair. As a result, the footholds optimization problem can be transformed to a ZMP optimization problem.

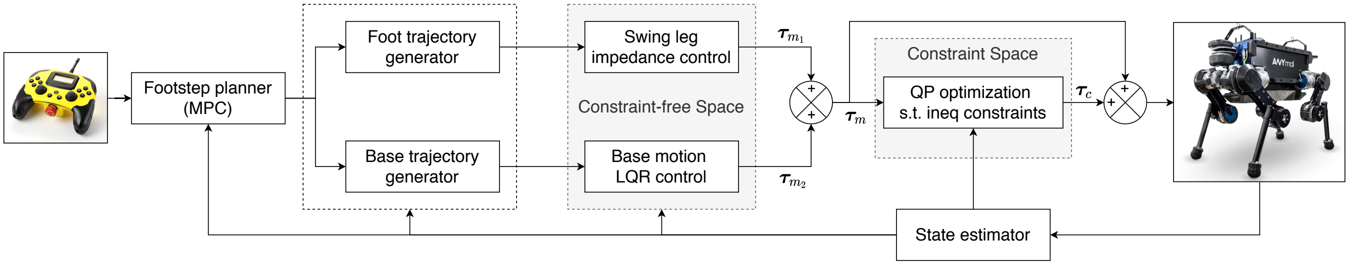

Fig. 2. Control framework block diagram. All the modules are running at 400 Hz.

A. MPC formulation

The dynamics of the linear inverted pendulum is as follows:

$$\begin{aligned} \ddot{x}_{CoM} &= \frac{g}{z_{CoM}}(x_{CoM} - p_x) \\ \ddot{y}_{CoM} &= \frac{g}{z_{CoM}}(y_{CoM} - p_y) \end{aligned} \tag{1}$$

where x_{CoM}, y_{CoM} and z_{CoM} are the CoM position coordinates, p_x and p_y are the coordinates of ZMP, g represents the acceleration of gravity. Considering z_{CoM} as constant, the dynamics become linear and result in the following solution:

$$\begin{aligned} \mathbf{x}_{CoM}(t) &= \mathbf{A}(t)\mathbf{x}^0_{CoM} + \mathbf{B}(t)p_x \\ \mathbf{y}_{CoM}(t) &= \mathbf{A}(t)\mathbf{y}^0_{CoM} + \mathbf{B}(t)p_y \end{aligned} \tag{2}$$

where $\mathbf{x}_{CoM} = [x_{CoM} \quad \dot{x}_{CoM}]^\top$, $\mathbf{y}_{CoM} = [y_{CoM} \quad \dot{y}_{CoM}]^\top$, are the state vectors, and $\mathbf{x}^0_{CoM}$ and $\mathbf{y}^0_{CoM}$ are the initial state vectors. $\mathbf{A}(t)$ and $\mathbf{B}(t)$ are defined as

$$\mathbf{A}(t) = \begin{bmatrix} \cosh(\omega t) & \omega^{-1}\sinh(\omega t) \\ \omega\sinh(\omega t) & \cosh(\omega t) \end{bmatrix} \tag{3}$$

$$\mathbf{B}(t) = \begin{bmatrix} 1 - \cosh(\omega t) \\ -\omega\sinh(\omega t) \end{bmatrix} \tag{4}$$

while $\omega = \sqrt{g/z_{CoM}}$.

For a periodic trotting gait with fixed swing duration T_s, assuming instant switching between single support phases, the states of N future steps along x direction can be predicted given step duration T_{s_i}

$$\begin{aligned} \mathbf{x}_{CoM_1} &= \mathbf{A}(T_{s_1})\mathbf{x}_{CoM_0} + \mathbf{B}(T_{s_1})p_{x_1} \\ \mathbf{x}_{CoM_2} &= \mathbf{A}(T_{s_2})\mathbf{x}_{CoM_1} + \mathbf{B}(T_{s_2})p_{x_2} \\ &\vdots \\ \mathbf{x}_{CoM_N} &= \mathbf{A}(T_{s_N})\mathbf{x}_{CoM_{N-1}} + \mathbf{B}(T_{s_N})p_{x_N} \end{aligned} \tag{5}$$

where $\mathbf{x}_{CoM_0}$ is the state at the moment of first touchdown, which can be computed from

$$\mathbf{x}_{CoM_0} = \mathbf{A}(t_0)\mathbf{x}^0_{CoM} + \mathbf{B}(t_0)p_{x_0} \tag{6}$$

where t_0 is the remaining period of the current swing phase. $\mathbf{x}^0_{CoM}$ is the current state given by the state estimator which also runs at 400 Hz.

The state along y direction has the same evolution as shown in Eq. (5). Regarding ZMPs as the system inputs, we define the cost function of the MPC as follows

$$J_x = \sum_{i=1}^{N}[\frac{1}{2}Q_i(\dot{x}_{CoM_i} - \dot{x}_{CoM_d})^2 + \frac{1}{2}R_i(p_{x_i} - p_{x_{i-1}})^2] \tag{7}$$

where $\dot{x}_{CoM_d}$ is the desired CoM velocity in the x direction, Q_i and R_i are weight factors. The cost function for the y direction has the same form as Eq. (7). The MPC is formulated as a QP minimising Eq. (7) subject to Eq. (5). Solving the QP results in the optimal ZMPs for the future N steps $\mathbf{p}^*_x = [p^*_{x_1} \quad p^*_{x_2} \quad \dots \quad p^*_{x_N}]^\top$. Similarly, solving another QP for y direction yields the coordinate $\mathbf{p}^*_y = [p^*_{y_1} \quad p^*_{y_2} \quad \dots \quad p^*_{y_N}]^\top$ for the optimal ZMPs in this direction. We only use the first pair $\mathbf{p}^*_1 = (p^*_{x_1} \quad p^*_{y_1})$ to generate the swing trajectory. Since the MPC is running in the same loop of controller, the position $\mathbf{p}^*_1 = (p^*_{x_1} \quad p^*_{y_1})$ keeps updating during a swing phase given the updated CoM state $(\mathbf{x}^0_{CoM} \quad \mathbf{y}^0_{CoM})$ and desired CoM velocity $(\dot{x}_{CoM_d} \quad \dot{y}_{CoM_d})$.

B. Reference trajectories

This section explains the algorithms to generate the desired trajectories for swing feet and the CoM based on the results of the MPC. The MPC provides the optimal ZMP that should be on the line connecting the next pair of support legs. We choose the ZMP to be the middle point of the support line. We keep the distance from the ZMP to each support foot location to be a fixed value r. Then we use the following equations to compute the desired footholds when the feet are swing (depicted in Fig. 3):

$$\begin{aligned} \text{LF}: \quad \begin{bmatrix} p^{\text{LF}}_x \\ p^{\text{LF}}_y \end{bmatrix} &= \begin{bmatrix} p^*_{x_1} \\ p^*_{y_1} \end{bmatrix} + r\begin{bmatrix} \cos(\theta_0 + \Delta\theta) \\ \sin(\theta_0 + \Delta\theta) \end{bmatrix} \\ \text{RH}: \quad \begin{bmatrix} p^{\text{RH}}_x \\ p^{\text{RH}}_y \end{bmatrix} &= \begin{bmatrix} p^*_{x_1} \\ p^*_{y_1} \end{bmatrix} + r\begin{bmatrix} -\cos(\theta_0 + \Delta\theta) \\ -\sin(\theta_0 + \Delta\theta) \end{bmatrix} \\ \text{RF}: \quad \begin{bmatrix} p^{\text{RF}}_x \\ p^{\text{RF}}_y \end{bmatrix} &= \begin{bmatrix} p^*_{x_1} \\ p^*_{y_1} \end{bmatrix} + r\begin{bmatrix} \cos(\theta_0 - \Delta\theta) \\ -\sin(\theta_0 - \Delta\theta) \end{bmatrix} \\ \text{LH}: \quad \begin{bmatrix} p^{\text{LH}}_x \\ p^{\text{LH}}_y \end{bmatrix} &= \begin{bmatrix} p^*_{x_1} \\ p^*_{y_1} \end{bmatrix} + r\begin{bmatrix} -\cos(\theta_0 - \Delta\theta) \\ \sin(\theta_0 - \Delta\theta) \end{bmatrix} \end{aligned} \tag{8}$$

where LF, RH, RF and LH are the abbreviations for left-fore, right-hind, right-fore and left-hind feet. θ_0 is a constant angle measured in the default configuration while $\Delta\theta$ is the rotation command sent by the users. Although Eq. (8) is

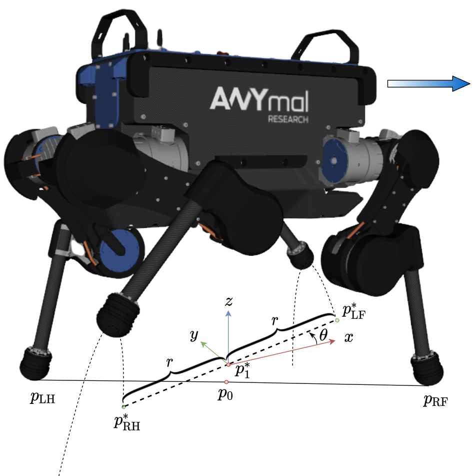

Fig. 3. Geometrical relationship between footholds and ZMPs. We assign the current ZMP (p_0) to be the middle point of the support line at the touchdown moment. The desired footholds (p^*_{LF}, p^*_{RH}) are calculated from the desired ZMP (p^*_1) and two foot pair parameters r and θ. r determine the distance between the foot pair and θ determines the orientation of the foot pair with respect to the robot heading direction. Nominal values are used for these two parameters. If there is a given steering command ω_z, the orientation can be updated $\theta = \theta_0 + \omega_z \cdot dt$.

specified for a trotting gait, this strategy of assigning the ZMP as the middle point can be applied to other gaits as well.

Here we do not tackle the height changing issue. We use the current height of support feet to be the desired height of desired footholds for swing feet. The peak height during swing is a fixed relative offset. This technique has to be adapted for some tasks such as climbing stairs. But the robustness of the planner and controller could handle slightly rough terrains. We will show this through experiments of traversing rough terrains. After determining the desired footholds, we use cubic splines to interpolate the trajectories between the initial foot positions and desired footholds for the swing feet, and feed the one forward time step positions, velocities and accelerations to the controller.

The desired positions and velocities for CoM are determined by the LIPM, i.e., Eq. (2) where the initial states $\mathbf{x}^0_{CoM}$ and $\mathbf{y}^0_{CoM}$ are updating with 400 Hz as well. Setting the variable t in Eq. (2) to be a constant value $t = 2.5$ ms results in the desired CoM positions and velocities along x and y for controller. We set the desired height of CoM to be a constant value with respect to the average height of the support feet.

III. LQR FOR BASE CONTROL

We continue to use our projected inverse dynamic control framework [19] as it allows us to focus on designing trajectory tracking controllers without considering physical constraints. The physical constraints will be satisfied in an orthogonal subspace. This framework gives us the opportunity to use the classical LQR without any adaptation.

The dynamics of a legged robot can be projected into two orthogonal subspaces by using the projection matrix $\mathbf{P} = \mathbf{I} - \mathbf{J}_c^+ \mathbf{J}_c$ [20][21] as follows:

Constraint-free space:

$$\mathbf{P}\mathbf{M}\ddot{\mathbf{q}} + \mathbf{P}\mathbf{h} = \mathbf{P}\mathbf{S}\boldsymbol{\tau} \tag{9}$$

Constraint space:

$$(\mathbf{I} - \mathbf{P})(\mathbf{M}\ddot{\mathbf{q}} + \mathbf{h}) = (\mathbf{I} - \mathbf{P})\mathbf{S}\boldsymbol{\tau} + \mathbf{J}_c^\top \boldsymbol{\lambda}_c \tag{10}$$

where $\mathbf{q} = \begin{bmatrix} {}_I\mathbf{x}_b^\top & \mathbf{q}_j^\top \end{bmatrix}^\top \in SE(3) \times \mathbb{R}^n$, where ${}_I\mathbf{x}_b \in SE(3)$ denotes the floating base's position and orientation with respect to a fixed inertia frame I, meanwhile $\mathbf{q}_j \in \mathbb{R}^n$ denotes the vector of actuated joint positions. Also, we define the generalized velocity vector as $\dot{\mathbf{q}} = \begin{bmatrix} {}_I\mathbf{v}_b^\top & {}_B\boldsymbol{\omega}_b^\top & \dot{\mathbf{q}}_j^\top \end{bmatrix}^\top \in \mathbb{R}^{6+n}$, where ${}_I\mathbf{v}_b \in \mathbb{R}^3$ and ${}_B\boldsymbol{\omega}_b \in \mathbb{R}^3$ are the linear and angular velocities of the base with respect to the inertia frame expressed respectively in the I and B frame which is attached on the floating base. $\mathbf{M} \in \mathbb{R}^{(n+6)\times(n+6)}$ is the inertia matrix, $\mathbf{h} \in \mathbb{R}^{n+6}$ is the generalized vector containing Coriolis, centrifugal and gravitational effects, $\boldsymbol{\tau} \in \mathbb{R}^{n+6}$ is the vector of torques, $\mathbf{J}_c \in \mathbb{R}^{3k\times(n+6)}$ is the constraint Jacobian that describes $3k$ constraints, k denotes the number of contact points accounting foot contact and body contact, $\boldsymbol{\lambda}_c \in \mathbb{R}^{3k}$ are constraint forces acting on contact points, and

$$\mathbf{S} = \begin{bmatrix} \mathbf{0}_{6\times6} & \mathbf{0}_{6\times n} \\ \mathbf{0}_{n\times6} & \mathbf{I}_{n\times n} \end{bmatrix} \tag{11}$$

is the selection matrix with n dimensional identity matrix $\mathbf{I}_{n\times n}$.

Note that Eq. (9) together with Eq. (10) provides the whole system dynamics. The sum of the torque commands generated in the two subspaces will be the final command sent to the motors as shown in Fig. 2. In this paper, we focus on trajectory tracking control in the constraint-free subspace. We refer to our previous paper [19] for the inequality constraint satisfaction in the constraint subspace. The swing legs are controlled by impedance controllers proposed in our former paper [19]. In this paper, we propose to replace the impedance controller for base control with an LQR controller, benefiting from the optimal gain matrix instead of the hand-tuned diagonal impedance gain matrices.

A. Linearization in Cartesian space

Based on Eq. (9), we derive the forward dynamics

$$\ddot{\mathbf{q}} = \mathbf{M}_c^{-1}(-\mathbf{P}\mathbf{h} + \dot{\mathbf{P}}\dot{\mathbf{q}}) + \mathbf{M}_c^{-1}\mathbf{P}\mathbf{S}\boldsymbol{\tau} \tag{12}$$

where $\mathbf{M}_c = \mathbf{P}\mathbf{M} + \mathbf{I} - \mathbf{P}$. Eq. (12) could be linearized with respect to the full state vector $\begin{bmatrix} \mathbf{q}^\top & \dot{\mathbf{q}}^\top \end{bmatrix}^\top$. However, the resulting linearized system would not be controllable since the control inputs are much less than the outputs. Instead of resorting to one more projection as done in [16], we linearize the dynamics in the Cartesian space to control only the base states rather than all the states of a whole robot.

Just using a selection matrix, we can derive the forward dynamics with respect to ${}_I\mathbf{x}_b$

$$\ddot{\mathbf{x}}_b = \mathbf{J}_b\mathbf{M}_c^{-1}(-\mathbf{P}\mathbf{h} + \dot{\mathbf{P}}\dot{\mathbf{q}}) + \mathbf{J}_b\mathbf{M}_c^{-1}\mathbf{P}\mathbf{S}\boldsymbol{\tau} = f({}_I\mathbf{x}_b, \dot{\mathbf{x}}_b, \boldsymbol{\tau}) \tag{13}$$

where $\mathbf{J}_b = [\mathbf{I}_{6\times6} \quad \mathbf{0}_{6\times n}]_{6\times(n+6)}$, $\dot{\mathbf{x}}_b = \begin{bmatrix} {}_I\mathbf{v}_b^\top & {}_B\boldsymbol{\omega}_b^\top \end{bmatrix}^\top$.

By using Euler angles for the orientation in ${}_I\mathbf{x}_b$, we can define the state vector as

$$\mathbf{X} = \begin{bmatrix} {}_I\mathbf{x}_b \\ \dot{\mathbf{x}}_b \end{bmatrix}_{12\times1} \tag{14}$$

We linearize Eq. (13) to state space dynamics around a configuration $(\mathbf{q}_0, \dot{\mathbf{q}}_0, \boldsymbol{\tau}_0)$

$$\dot{\mathbf{X}} = \mathbf{A}_0^b\mathbf{X} + \mathbf{B}_0^b\boldsymbol{\tau} \tag{15}$$

where $\boldsymbol{\tau}_0$ is the gravity compensation torques. Eq. (15) is detailed as

$$\begin{bmatrix} \dot{\mathbf{x}}_b \\ \ddot{\mathbf{x}}_b \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} {}_I\mathbf{x}_b \\ \dot{\mathbf{x}}_b \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{B}_2 \end{bmatrix} \boldsymbol{\tau} \tag{16}$$

where $\mathbf{A}_{21}$, $\mathbf{A}_{22}$ and $\mathbf{B}_2$ are defined as

$$\mathbf{A}_{21} = \frac{\partial f({}_I\mathbf{x}_b, \dot{\mathbf{x}}_b, \boldsymbol{\tau})}{\partial {}_I\mathbf{x}_b}|_{\mathbf{q}_0,\dot{\mathbf{q}}_0,\boldsymbol{\tau}_0} \tag{17}$$

$$\mathbf{A}_{22} = \frac{\partial f({}_I\mathbf{x}_b, \dot{\mathbf{x}}_b, \boldsymbol{\tau})}{\partial \dot{\mathbf{x}}_b}|_{\mathbf{q}_0,\dot{\mathbf{q}}_0} \tag{18}$$

$$\mathbf{B}_2 = \frac{\partial f({}_I\mathbf{x}_b, \dot{\mathbf{x}}_b, \boldsymbol{\tau})}{\partial \boldsymbol{\tau}} = \mathbf{M}_c^{-1}\mathbf{PS}|_{\mathbf{q}_0} \tag{19}$$

For simplicity, we use a central finite difference method to compute the partial derivatives of Eq. (17) and Eq. (18). The deviation factor for finite difference we used for the experiments is 1×10^{-5}.

B. LQR controller

We consider Eq. (15) as a linear time-invariant system and solve the infinite horizon LQR problem to compute the optimal feedback gain matrix $\mathbf{K}$. The cost function to be minimized is defined as

$$J = \int_0^\infty (\mathbf{X}^\top\mathbf{Q}\mathbf{X} + \boldsymbol{\tau}^\top\mathbf{R}\boldsymbol{\tau})dt \tag{20}$$

and the resulting controller for the base control is

$$\boldsymbol{\tau}_{m_2} = \mathbf{K}(\mathbf{X}_d - \mathbf{X}) + \boldsymbol{\tau}_0 \tag{21}$$

We use ADRL Control Toolbox (CT) [22] to solve the infinite-horizon LQR problem and obtain the $\mathbf{K}$ matrix. It should be noted that the linearization is computed in every control cycle based on the current configuration $(\mathbf{q}_0, \dot{\mathbf{q}}_0, \boldsymbol{\tau}_0)$. The $\mathbf{K}$ matrix is updated at 400 Hz, which is different to [17] where they only compute the $\mathbf{K}$ matrices corresponding to few key configurations. We think linearization should be updated around current configuration in order to improve computation accuracy if the computation is fast enough.

In practice, we reduce the weights in $\mathbf{R}$ of Eq. (20) for swing legs, relying more on the support legs for base control. Otherwise, the torque commands of Eq. (21) can affect the tracking of swing trajectories too much.

In addition, the motion planner in Section II feeds the desired CoM trajectory to the controllers, whereas the LQR controller controls the base pose. In theory, we should replace ${}_I\mathbf{x}_b$ with $\mathbf{x}_{CoM}$ in Eqs. (9)(10) and transform the dynamic equations to be with respect to CoM variables as in [23].

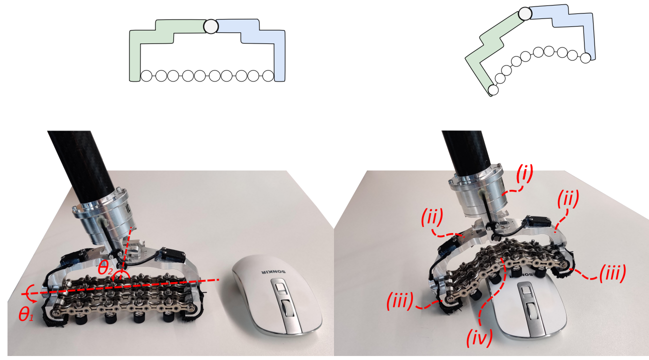

Fig. 4. The SoftFoot-Q, an adaptive foot for quadrupeds.

Then the LQR controller will directly track the desired CoM trajectory. In this paper, we approximately consider the translation of base along x and y aligned with CoM since the base dominates the mass of the whole robot.

IV. Adaptive Feet

Traditionally, the feet of quadruped robots are designed to be spherical. The contact between this kind of feet and the ground is regarded as point contact, which has three rotational degrees of freedom. With a very small contact area, spherical feet are easy to slip on rough terrains. Moreover, penetration easily happens when walking on sandy and muddy terrains. Motivated by these issues, we test our approach employing an adaptive soft foot called SoftFoot-Q [24] (see Fig. 4).

This foot is made of four main components (with reference to Fig. 4): (i) an ankle base is the connection interface with the tip of the robot leg; (ii) two arch links create the plantar arch and provide pitching motions to the foot by means of a single revolute joint, which connects the two arches to the ankle; (iii) two roll links, connected to the lower extremities of the two arches by two revolute joints, enable rolling motions of the sole along the frontal plane; (iv) the foot sole is composed of four paddled chains, which allow adaptation to the profile of the terrain.

The mechanically adaptive sole of SoftFoot-Q enables a stiffening by compression behaviour: as downward forces are applied to the feet, the chains tend to adapt to the shape of the terrain and guarantee an increase in the support area. This adaptation leads to a reduction in the phenomenon of slippage and eventually to a more stable robot locomotion.

The passive revolute joints of the feet have the aim to favour the adaptability of the sole to the terrain profile. Assuming that the play between the links of the chains is negligible, the two roll joints can be seen as moving specularly. The range of motions of the pitch and roll joints are respectively ±45 deg and ±30 deg. It is noteworthy that the adaptive foot does not support any yaw motion; this leads to the change of constraint Jacobian $\mathbf{J}_c$ in Eq. (10). However, we ignore that change since rotation of Yaw at ankle is merely used during trotting.

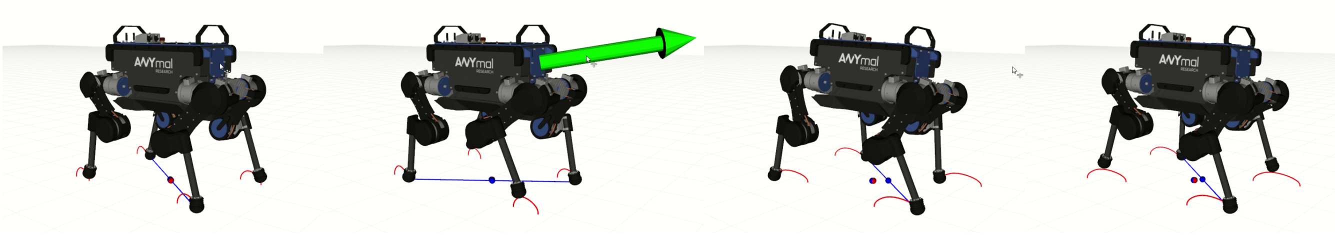

Fig. 5. Footstep and swing trajectory replanning under disturbances. The robot is walking forward and an external force (green arrow) is applied to the base of the robot. The push results in a sudden change of the CoM position and velocity. The footstep planner uses the updated state to replan the footholds. The swing trajectories (red lines) are updated accordingly.

Fig. 6. Kicking the robot during trotting. Benefiting from the 400 Hz MPC update frequency, the robot can quickly update the optimal footholds to recover from disturbances.

V. Experiments

We use a torque controllable quadruped robot ANYmal made by ANYbotics[1] to conduct our experiments. The onboard computer has an Intel 4th generation (HaswellULT) i7-4600U (1.4 GHz-2.1 GHz) processor and two HX316LS9IBK2/16 DDR3L memory cards. The robot weights approximately $35\,\mathrm{kg}$ and has 12 joints actuated by Series Elastic Actuators (SEAs) with maximum torque of $40\,\mathrm{N}\cdot\mathrm{m}$. The real-time control cycle is $2.5\,\mathrm{ms}$. The control software is developed based on Robot Operating System 1 (ROS 1). We use the dynamic modeling library Pinocchio [25] to perform the model linearization of Section III. An active set method based QP solver provided by ANYbotics is used to solve the QPs for the MPC planner and the controller. A video of the experimental results can be found at: `https://www.youtube.com/playlist?list=PLapqoVVJ_B7I3VtneBAK520-RvLOurbFq`.

A. Trotting speed

We first tested the fastest walking speed when using the proposed algorithms. Figure 7 shows the recorded speeds along x direction in real robot experiment and in simulation. In simulation, the robot could stably trot forward with maximum speed 1.2 m/s. On real robot, the maximum speed reached 0.6 m/s. The results are reasonable since the trotting gait does not have a flying phase. The fact that the real robot cannot achieve as fast motion as in simulation is also reasonable considering model errors and other uncertainties. Model errors also cause drifting on the real robot which is difficult to resolve without external control loops. Constant values for the parameters of the gait planner were employed. They are $T_s = 0.3$, $z_{CoM} = 0.42$, $g = -9.8$, $N = 3$, $Q_i = 1000$, $R_i = 1$, $\theta_0 = 0.56$, $r = 0.41$. It should be noted that the prediction number N in the MPC does not need to be as large as possible with the concern of computation efficiency. We tested $N = 2 \sim 5$, and they showed similar performance.

[1] https://www.anybotics.com

B. Push recovery

In this subsection, we demonstrate the benefit of high frequency replanning for disturbance rejection. We first use simulation to show the replanned footholds and trajectories as shown in Fig. 5. The disturbance is added when RF and LH feet are swinging. The disturbance results in sharp state changes. The MPC computed the new footholds after receiving the updated state. Figure 6 presents snapshot photos of the push-recovery experiment on the real robot during trotting while recorded state data is shown in Fig. 8. The robot was kicked four times roughly along the y direction. We can see the peak velocity of y reached -1 m/s during the last two kicks, but it was quickly regulated back to normal using one or two steps. The orientation did not change too much after kicking, which also indicates the robustness of the method.

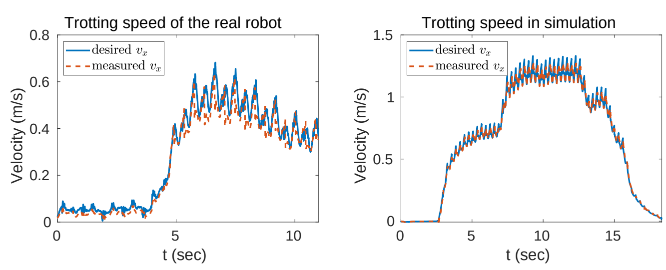

Fig. 7. Recorded fastest trotting speeds on real robot and in simulation. The desired velocities are generated from LIPM dynamics, i.e. Eq. (2).

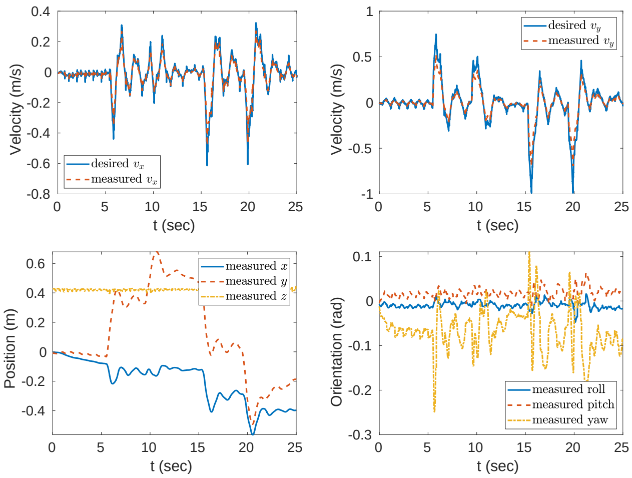

Fig. 8. Recorded state when the robot was kicked four times. The desired Euler angles were 0.

C. Balance control

Most of the trotting gait control algorithms rely on quick switching of swing and stance phases to achieve dynamic balance. Recently researchers demonstrated that quadruped robots with point feet can stand on two feet to maintain balance [26][27]. Although we did not manage balancing on two feet on our robot, we compared the longest period of swing phase of trotting when using our proposed LQR controller versus the default trotting controller of ANYmal [28]. The longest swing phase when using LQR is 0.63s whereas the default controller only achieved 0.42s. This verifies the improved performance of our LQR controller in terms of balance control. Fig. 9 shows two gain matrices for the two phases of trotting during this experiment. It should be noted that the gain matrices are updated in every control cycle (but the changes are small). We can see that the elements of the first 6 rows are 0 because of the existence of selection matrix $\mathbf{S}$. The $\mathbf{Q}_{12\times12}$ we used in the experiment was $\mathbf{Q} = \text{diag}(\text{diag}(1500)_{6\times6},\ \text{diag}(1)_{6\times6})$. The $\mathbf{R}_{18\times18}$ was switched depending on the phase. The diagonal elements corresponding to the two swing legs in $\mathbf{R}$ are 10 times larger than the other diagonal elements for stance legs and the base, reducing the efforts of swing legs in balance control. The elements in $\mathbf{R}$ for stance legs and the base we used in this experiment are 0.03.

D. Outdoor test

In this subsection, we test the versatility of of our approach also with adaptive feet. We perform experiments consisting in trotting locomotion on two kinds of terrains outside our lab.

Firstly, we took the robot to an environment with gravel where the spherical feet suffer from soil penetration and easily slip. The adaptive feet showed considerable advantage over the spherical feet since the sole has a larger contact surface and disperses the contact forces. The robot successfully walked for approximately 5 meters with a rough speed around 0.1 m/s on the gravel terrain as shown in Fig. 10.

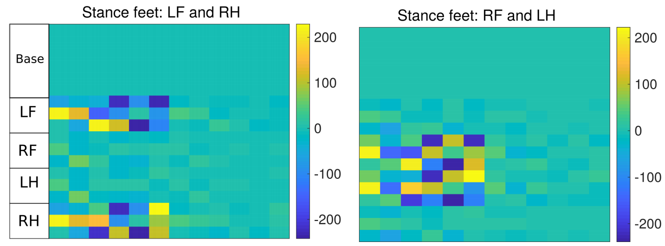

Fig. 9. Visualization of the gain matrix as computed by the LQR controller during trotting. The size of the gain matrix is 18 by 12. The first 6 columns correspond to the position and orientation while the remaining 6 columns are for velocity control. Off-diagonal gains demonstrate that dynamic coupling effects may be exploited for control.

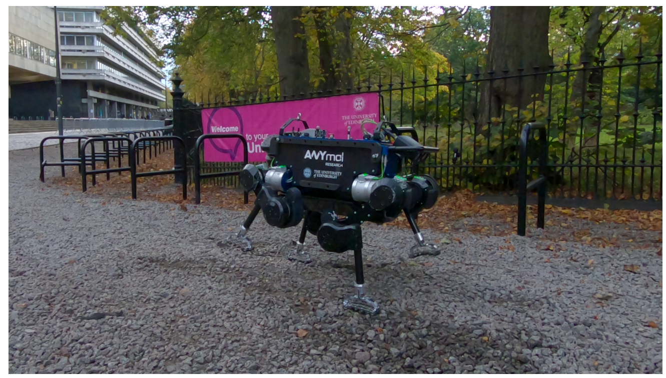

Fig. 10. Trotting out of the lab with adaptive feet on rubble terrain.

Fig. 11. Trotting out of the lab with adaptive feet on bumpy terrain.

Additionally, it also can be noted that the terrain is inclined, which indicates the planner could handle inclined terrains as long as the stance feet are not diverging too much in height.

We also tested the robot on a bumpy terrain where the adaptive feature of the feet is highlighted as shown in Fig. 11. Usage of the adaptive feet effectively avoids slipping on such terrain. On the other hand, as the planner and controller did not take the passive joints of the adaptive feet into account, we did not achieve the fastest speed of 0.6 m/s. Without considering the displacement between the two passive joints of the foot, more model errors arise and cause extra internal forces and torques at the ankle. In the future, we will adapt the planner and controller to handle the presence of these passive joints.

VI. CONCLUSIONS

This paper presents a full control framework for dynamic gaits where all the modules are running with the same frequency. The robustness of the dynamic walking is improved significantly by two factors. The first factor is the MPC planner, which mostly contributes to rejecting large disturbances, such as kicking the robot, because the MPC uses footsteps to regulate the state of the robot. The second factor is the LQR controller for balancing control, which also undertakes the duty of trajectory tracking. The method is general and shown to able to work both with spherical and adaptive feet. The latter were seen to reduce the slipping chance on rough terrains. The outdoor experiments demonstrate the robustness of locomotion after adopting the proposed methods and assembling the adaptive feet.

Future work will focus on adapting the current planner to consider terrain information to handle large slopes and stairs. Also, the new feet can be used to measure the local inclination of the ground which can improve the accuracy of the terrain information.

ACKNOWLEDGMENT

The authors thank Dr. Quentin Rouxel and Dr. Carlos Mastalli for introduction to the use of the Pinocchio toolbox.

REFERENCES

[1] A. W. Winkler, F. Farshidian, D. Pardo, M. Neunert, and J. Buchli, "Fast trajectory optimization for legged robots using vertex-based zmp constraints," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2201–2208, 2017.

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

[3] 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 *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[4] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[5] O. Cebe, C. Tiseo, G. Xin, H.-c. Lin, J. Smith, and M. Mistry, "Online dynamic trajectory optimization and control for a quadrupedrobot," *arXiv preprint arXiv:2008.12687*, 2020.

[6] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard, "Whole-body model-predictive control applied to the hrp-2 humanoid," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 3346–3351.

[7] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard, "Crocoddyl: An efficient and versatile framework for multi-contact optimal control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2536–2542.

[8] S. Faraji, S. Pouya, C. G. Atkeson, and A. J. Ijspeert, "Versatile and robust 3d walking with a simulated humanoid robot (atlas): A model predictive control approach," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 1943–1950.

[9] S. Feng, X. Xinjilefu, C. G. Atkeson, and J. Kim, "Robust dynamic walking using online foot step optimization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 5373–5378.

[10] S. Xin, R. Orsolino, and N. G. Tsagarakis, "Online relative footstep optimization for legged robots dynamic walking using discrete-time model predictive control." in *IROS*, 2019, pp. 513–520.

[11] M. De Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," in *ACM Transactions on Graphics (TOG)*, vol. 29, no. 4. ACM, 2010, p. 131.

[12] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Soueres, and J. Y. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 346–362, 2013.

[13] A. Del Prete, F. Nori, G. Metta, and L. Natale, "Prioritized motion–force control of constrained fully-actuated robots:"task space inverse dynamics"," *Robotics and Autonomous Systems*, vol. 63, pp. 150–157, 2015.

[14] A. Herzog, N. Rotella, S. 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.

[15] P. Reist and R. Tedrake, "Simulation-based lqr-trees with input and state constraints," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 5504–5510.

[16] S. Mason, L. Righetti, and S. Schaal, "Full dynamics lqr control of a humanoid robot: An experimental study on balancing and squatting," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 374–379.

[17] S. Mason, N. Rotella, S. Schaal, and L. Righetti, "Balancing and walking using full dynamics lqr control with contact constraints," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 63–68.

[18] G. Xin, H.-C. Lin, J. Smith, O. Cebe, and M. Mistry, "A model-based hierarchical controller for legged systems subject to external disturbances," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4375–4382.

[19] G. Xin, W. Wolfslag, H.-C. Lin, C. Tiseo, and M. Mistry, "An optimization-based locomotion controller for quadruped robots leveraging cartesian impedance control," *Frontiers in Robotics and AI*, vol. 7, p. 48, 2020.

[20] F. Aghili, "Quadratically constrained quadratic-programming based control of legged robots subject to nonlinear friction cone and switching constraints," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2469–2479, 2017.

[21] M. Mistry and L. Righetti, "Operational space control of constrained and underactuated systems," *Robotics: Science and systems VII*, pp. 225–232, 2012.

[22] M. Giftthaler, M. Neunert, M. Stäuble, and J. Buchli, "The control toolbox — an open-source c++ library for robotics, optimal and model predictive control," *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pp. 123–129, 2018.

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

[24] M. G. Catalano, M. J. Pollayil, G. Grioli, G. Valsecchi, H. Kolvenbach, M. Hutter, A. Bicchi, and M. Garabini, "Adaptive Feet for Quadrupedal Walkers," *IEEE Transactions on Robotics*, 2020, [Under Review]. [Online]. Available: https://www.dropbox.com/s/84ry3we72c7kt6s/adaptive_feet_preprint.pdf

[25] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The pinocchio c++ library – a fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *IEEE International Symposium on System Integrations (SII)*, 2019.

[26] M. Chignoli and P. M. Wensing, "Variational-based optimal control of underactuated balancing for dynamic quadrupeds," *IEEE Access*, vol. 8, pp. 49 785–49 797, 2020.

[27] C. Gonzalez, V. Barasuol, M. Frigerio, R. Featherstone, D. G. Caldwell, and C. Semini, "Line walking and balancing for legged robots with point feet," *arXiv preprint arXiv:2007.01087*, 2020.

[28] C. Gehring, S. Coros, M. Hutter, M. Bloesch, M. A. Hoepflinger, and R. Siegwart, "Control of dynamic gaits for a quadrupedal robot," in *2013 IEEE international conference on Robotics and automation*. IEEE, 2013, pp. 3287–3292.