Safe Motion Planning for Quadruped Robots using Density Functions

Andrew Zheng*[1], Sriram S.K.S Narayanan*[1]

***Abstract*— This report presents a motion planning algorithm for quadruped robots based on density functions. We decompose the planning problem into a high-level global planner and a model predictive controller (MPC). Density functions are a physically intuitive way to represent the environment. It is a scalar positive valued function that takes zero values in the obstacle set and infinite values in the target set. The center of mass of the robot is modeled as an integrator system, and the high-level plan is obtained as a gradient of the density function. Further, the proposed planner can be implemented as a feedback controller for the system. We then use the MPC to follow the desired reference plan. Finally, a low-level PID controller is used to obtain the joint torques. The overall framework is implemented in simulation using ROS and Gazebo. Our project page with implementation is available at `https://github.com/AndrewZheng-1011/legged_planner`**

I. Introduction

Quadruped research has seen great success in the research community over the past few years [1]. This is highlighted through the DARPA Subterranean Challenge, where the top two teams had either ANYmal from ANYbotics or Spot from Boston Dynamics as a core component to their underground exploration challenge [2], [3]. This can be attributed to advancements in both the algorithmic component, researchers finding clever ways to deal with the hybrid dynamics, and the hardware component, hardware having real-time capabilities for nonlinear optimization problems.

In general, the quadruped locomotion problem can be decomposed into a high-level global planner which provides collision-free paths for the floating base and a low-level controller for designing the required joint torques for each leg to track the desired reference trajectory (see Figure 1). In this report, we use density functions to design a high-level plan for quadruped locomotion. The density function has a physical interpretation, where the measure associated with the density is a measure of occupancy of the system trajectories for any set in state space. We exploit this occupancy-based physical interpretation of the density function in the construction for designing a collision-free plan. This plan is obtained as the gradient of the density function and can be implemented as a feedback controller. We then use a nonlinear model predictive controller (NMPC) as a low-level planner and obtain the desired reference foot forces for each foot required to track the high-level reference plan. Finally, the desired foot forces are converted to joint torques using a low-level PID controller.

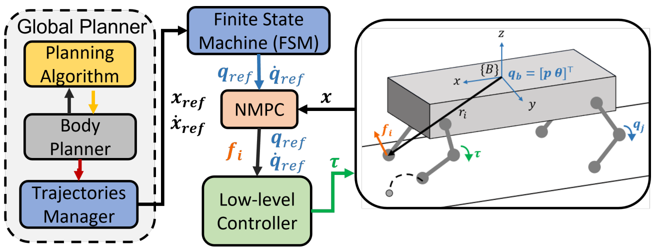

Fig. 1. Hierarchical planning and control structure for quadruped locomotion

The rest of the report is organized as follows. In Section II, we introduce the preliminaries of quadruped dynamics and density functions. In Section III, we provide simulation results of the proposed density-based planner fpr a single integrator system. In Section IV, we integrate the density-based planner with the quadruped locomotion framework using ROS and Gazebo. Finally, in Section VI, we conclude this work with some limitations and future directions.

II. Preliminaries

A. Rigid Body Dynamics

Quadruped locomotion can be described as a hybrid system switching between swing and stance phase dynamics. The switching logic is determined by a contact detection algorithm. This system is under-actuated since there is no direct actuation along the direction of motion. The robot must exert ground reaction forces (GRFs) at each foot in contact to propel its base forward to follow a reference trajectory. Most state-of-the-art approaches use a reduced-order model to represent quadruped dynamics. In this report, we use the centroidal model which describes the evolution of the floating base as a result of the net change of momentum of the full-order system. The state and input vectors are defined by

$$\boldsymbol{x} := [\boldsymbol{h}_{com}\ \boldsymbol{q}_b\ \boldsymbol{q}_j]^\top \in \mathbb{R}^{12+n}$$
$$\boldsymbol{u} := [\boldsymbol{f}_{c_i}\ \boldsymbol{v}_j] \in \mathbb{R}^{3n_c+n_j}$$

where $\boldsymbol{h}_{com} = [\boldsymbol{h}_{lin}\ \boldsymbol{h}_{ang}] \in \mathbb{R}^6$ is the centroidal momentum and $\boldsymbol{h}_{lin}$, $\boldsymbol{h}_{ang}$ are the linear and angular momentum respectively. The states $\boldsymbol{q}_b = [\boldsymbol{p}\ \boldsymbol{\theta}] \in \mathbb{R}^6$ represents the position and orientation of the floating base. The states $\boldsymbol{q}_j \in \mathbb{R}^n$ represent the joint angles for each foot. Each leg i in contact generates a ground reaction force $\boldsymbol{f}_{\boldsymbol{c_i}} \in \mathbb{R}^3$, and v_j is the angular velocity of each joint. Here, n_c and n_a are

*These authors contributed equally

[1]Andrew Zheng is a Masters student with the Department of Mechanical Engineering, Clemson University, Clemson, SC 29630, USA `azheng@clemson.edu`

[1]Sriram S.K.S Narayanan is a PhD student with the Department of Mechanical Engineering, Clemson University, Clemson, SC 29630, USA `sriramk@clemsons.edu`

the number of contact points and the number of leg joints respectively. The centroidal dynamics are given by

$$\dot{\boldsymbol{h}}_{com} = \begin{bmatrix} \sum_{i=1}^{n_c} \boldsymbol{f}_{c_i} + m\boldsymbol{g} \\ \sum_{i=1}^{n_c} \boldsymbol{r}_{com} \times \boldsymbol{f}_{c_i} + \boldsymbol{\tau}_{c_i} \end{bmatrix}$$

where m is the mass of the robot, $\boldsymbol{g} = [0 \;\; 0 \;\; g]^\top$ denotes the gravity vector. $\boldsymbol{f}_{c_i}$ and $\boldsymbol{\tau}_{c_i}$ represent the contact forces and torques applied by the environment on the floating base. Further, we use the centroidal momentum matrix $\boldsymbol{A}(\boldsymbol{q}) = [\boldsymbol{A}_b(\boldsymbol{q}) \; \boldsymbol{A}_j(\boldsymbol{q})] \in \mathbb{R}^{6\times(6+n)}$ as introduced in [4] such that

$$\boldsymbol{h}_{com} = \begin{bmatrix} \boldsymbol{A}_b(\boldsymbol{q}) & \boldsymbol{A}_j(\boldsymbol{q}) \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_b \\ \dot{\boldsymbol{q}}_j \end{bmatrix}$$

Finally, the dynamics of floating base $\boldsymbol{q}_b$ is augmented with the kinematics of the joints $\boldsymbol{q}_j$ to obtain

$$\dot{\boldsymbol{q}}_b = \boldsymbol{A}_b^{-1}\Big(\boldsymbol{h}_{com} - \boldsymbol{A}_j\dot{\boldsymbol{q}}_j\Big) \tag{1a}$$

$$\dot{\boldsymbol{q}}_j = \boldsymbol{v}_j \tag{1b}$$

Note that the dynamics presented in (1) is nonlinear and under-determined (since $\boldsymbol{A}(\boldsymbol{q})$ is a fat matrix). The system has multiple feasible solutions for a given input $\boldsymbol{u}$.

B. Nonlinear MPC

In this report, we adopt the nonlinear MPC proposed in [5]. In general, the nonlinear MPC problem with a horizon N can be formulated as

$$\min_{\boldsymbol{x},\boldsymbol{u}} \sum_{i=0}^{N-1} \{||\boldsymbol{x}_{k+1} - \boldsymbol{x}_{k+1,ref}||_{\boldsymbol{Q}_k} + ||\boldsymbol{u}||_{\boldsymbol{K}_k}\}$$

subject to

Dynamics (1)

Friction constraints

Actuator limits

where $\boldsymbol{x}_k$ and $\boldsymbol{u}_k$ are the state and control input respectively at time step k, $\boldsymbol{Q}_k$ and $\boldsymbol{K}_k$ are diagonal positive definite weight matrices. Here, $\mathbf{u}_i$ are the individual foot forces required to track the global plan. The optimization is solved in closed-loop in a receding horizon fashion using the Sequential Linear Quadratic (SLQ) technique [6].

C. Low-level Controller

The low-level controller is designed to track the optimal reference plans. This is formulated through a hierarchical quadratic program (QP) which optimizes the actuator torques under a higher fidelity model, friction, and actuator constraints. To map the MPC output to the QP, extra information is extrapolated such as the swing feet trajectories, which can be obtained through a simple kinematic relation of task to joint space acceleration

$$\boldsymbol{J}_{c_i}\ddot{\boldsymbol{q}}_j + \dot{\boldsymbol{J}}_{c_i}\dot{\boldsymbol{q}}_j = \ddot{\boldsymbol{r}}_{c_i} \tag{2}$$

where $\boldsymbol{J}_{c_i}$ is the swing leg jacobian and $\ddot{\boldsymbol{r}}$ is the end-effector acceleration. The optimized trajectory from the QP is tracked

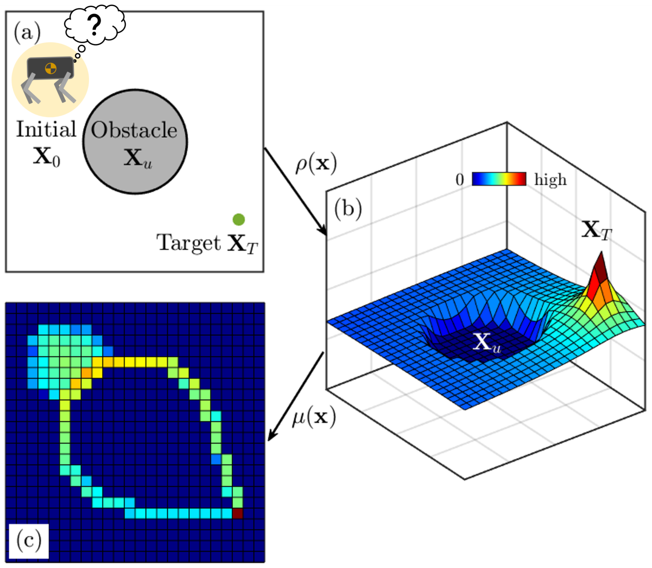

Fig. 2. Motion Planning framework using density where (a) defines the environment, (b) shows the density function, and (c) shows occupancy measure, which physically denotes the duration of system trajectories occupying the set.

using a drive controller where the torque actuated ($\boldsymbol{\tau}_a$) is defined by the following

$$\boldsymbol{\tau}_a = \boldsymbol{\tau}_j + \boldsymbol{K}_p(\boldsymbol{q}^\star - \boldsymbol{q}_j) + \boldsymbol{K}_d(\dot{\boldsymbol{q}}^\star - \dot{\boldsymbol{q}}_j) \tag{3}$$

where $\boldsymbol{\tau}_j$ is the expected joint torque, $\dot{\boldsymbol{q}}^\star$ and $\boldsymbol{q}^\star$ are the optimized joint reference states, More details on the QP formulation can be referenced to [5].

D. Construction of Density Functions

As introduced in [7], the navigation measure has a physical interpretation of occupancy, where the measure of any set is equal to the occupancy of the system trajectories in the set as shown in Figure 2. Hence, zero occupancy in a set implies system trajectories not occupying that particular set. So by ensuring that the navigation measure is zero on the obstacle set and maximum on the target set, it is possible to induce dynamics whereby the system trajectories will reach the desired target set while avoiding the obstacle set. We exploit this occupancy-based interpretation in the construction of analytical density functions.

Although density functions can represent any arbitrarily shaped obstacles, we restrict the focus of this report to circular obstacles. For k obstacles, we start with constructing the unsafe set $\mathbf{X}_{u_k}$, where the boundary of the unsafe set is described in terms of the zero-level set of a function. For a circular obstacle, the unsafe set $\mathbf{X}_{u_k}$ is defined as follows

$$\mathbf{X}_{u_k} = \{\mathbf{x} \in \mathbf{X} : \|\mathbf{x} - \mathbf{c}_k\| \leq r_k\}$$

Next, for each obstacle, we define a transition region $\mathbf{X}_{s_k}$ that encloses the unsafe set $\mathbf{X}_{u_k}$ such that the set $\{\mathbf{x} \in \mathbf{X} : s_k(\mathbf{x}) = 0\}$ defines the boundary of this transition region. A circular transition region can be defined by the following set

$$\mathbf{X}_{s_k} = \{\mathbf{x} \in \mathbf{X} : \|\mathbf{x} - \mathbf{c}_k\| \leq s_k\} \setminus \mathbf{X}_{u_k}$$

The physical significance of this transition region is to act as a sensing region inside which the robot trajectories start to react to the unsafe set. Now we can define a positive scalar-valued function $\rho(\mathbf{x})$, which takes the following form

$$\rho(\mathbf{x}) = \frac{\prod_{k=1}^{L} \Phi_k(\mathbf{x})}{V(\mathbf{x})^{\alpha}} \tag{4}$$

Here, the function $V(\mathbf{x})$ is the distance function that measures the distance from state $\mathbf{x}$ to the target set, i.e. the origin, and α is a positive scalar. In this paper, we assume $V(\mathbf{x})$ to be of the form $V(\mathbf{x}) = \|\mathbf{x}\|^2$. The inverse bump function $\Phi_k(\mathbf{x})$ is a smooth $\mathcal{C}^\infty$ function that captures the geometry of the unsafe set $\mathbf{X}_{u_k}$ and can be constructed using the following sequence of functions. We first define an elementary $\mathcal{C}^\infty$ function f as follows [8]

$$f(\tau) = \begin{cases} \exp\left(\frac{-1}{\tau}\right), & \tau > 0 \\ 0, & \tau \leq 0 \end{cases}$$

where $\tau \in \mathbb{R}$. Next, we construct a smooth version of a step function $\bar{f}$ from f as follows

$$\bar{f}(\tau) = \frac{f(\tau)}{f(\tau) + f(1-\tau)}$$

To incorporate the geometry of the environment, we define a change of variables such that $\phi_k(\mathbf{x}) = \bar{f}\left(\frac{\|\mathbf{x}-\mathbf{c}_k\|^2 - r_k^2}{s_k^2 - r_k^2}\right)$. The resulting function $\Phi_k(\mathbf{x})$ take the following form,

$$\Phi_k(\mathbf{x}) = \begin{cases} 0, & \mathbf{x} \in \mathbf{X}_{u_k} \\ \phi_k(\mathbf{x}), & \mathbf{x} \in \mathbf{X}_{s_k} \\ 1, & \text{otherwise} \end{cases}$$

Note that α and $\mathbf{s}_k$ are scalar tuning parameters that can be used to obtain trajectories with the desired behavior.

III. Planning for a Holonomic Single Integrator

Given the construction of $\rho(\mathbf{x})$ in (4), we design a controller for navigation as the positive gradient of the density function $\rho(\mathbf{x})$, i.e.,

$$\dot{\mathbf{x}} = \nabla\rho(\mathbf{x}) \tag{5}$$

The controller defined in equation (5) will converge to the goal if the density function $\rho(\mathbf{x})$ satisfies almost everywhere navigation properties [9]. In this report, we show that the density function proposed in equation (4) can be used for obstacle avoidance while also satisfying the convergence properties. We demonstrate this using a simple example as shown in Figure3. The environment is defined with the target set at $\mathbf{X}_T = (4, -3)$ and a circular unsafe set $\mathbf{X}_u$ and circular sensing region $\mathbf{X}_s$.

Figure 3a illustrates the convergence properties of the proposed controller. We can see that trajectories starting from almost all initial conditions starting at the boundary of the environment converge to the target while avoiding the obstacle. However, trajectories starting on the black line, which is the polar opposite of the target set, converge to a saddle point (shown in Figure 3b). This set of initial conditions constitutes a measure zero set $m\mathbf{X} = 0$.

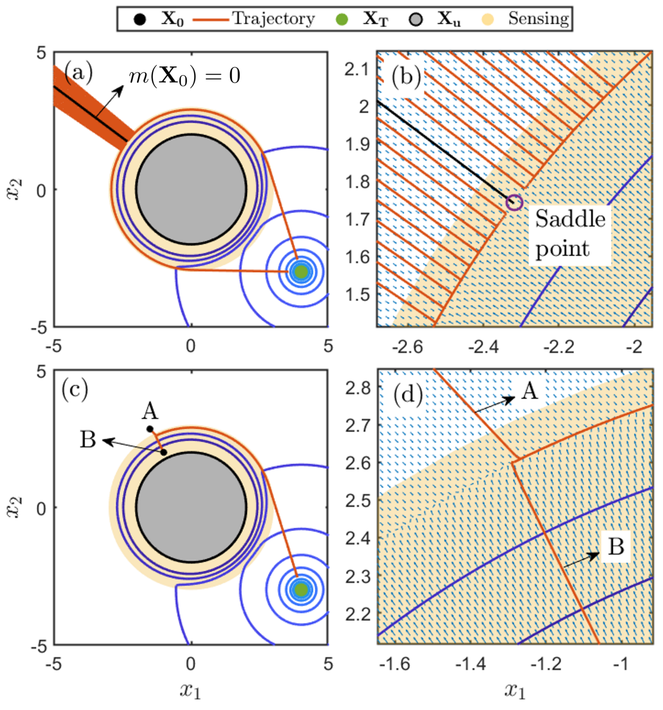

Fig. 3. (a) Trajectories converge to the target set (green) while avoiding the unsafe set (gray) with a.e. convergence, (b) Initial conditions along the zero-measure set (black) converge to a saddle point (purple), (c) Trajectories starting at A ($s(\mathbf{x}) > 0$) and B (in $\mathbf{X}_{s_k}$) converge to the target set, (d) Trajectories starting from A and B follow the same path near the boundary of $s(\mathbf{x})$.

In Figure 3c, we look at the characteristics of initial conditions starting outside the sensing region (trajectory A), and within the sensing region (trajectory B). The gradients of the density function $\rho(\mathbf{x})$ are such that trajectory A starts to react as it enters the sensing region. In contrast, trajectory B is repelled outward towards the boundary of the sensing region before converging to the target set (see Figure 3d).

TABLE I
Density Parameters

Parameter	Value
α	0.2
r	1
s	2
Control Gain	25

IV. Planning for Quadruped Locomotion

This section discusses the planning architecture to map our feedback density planner for holonomic systems to quadruped locomotion. We show planning architecture using the density planner and show convergence and obstacle avoidance properties of the controller for a quadruped robot in simulation.

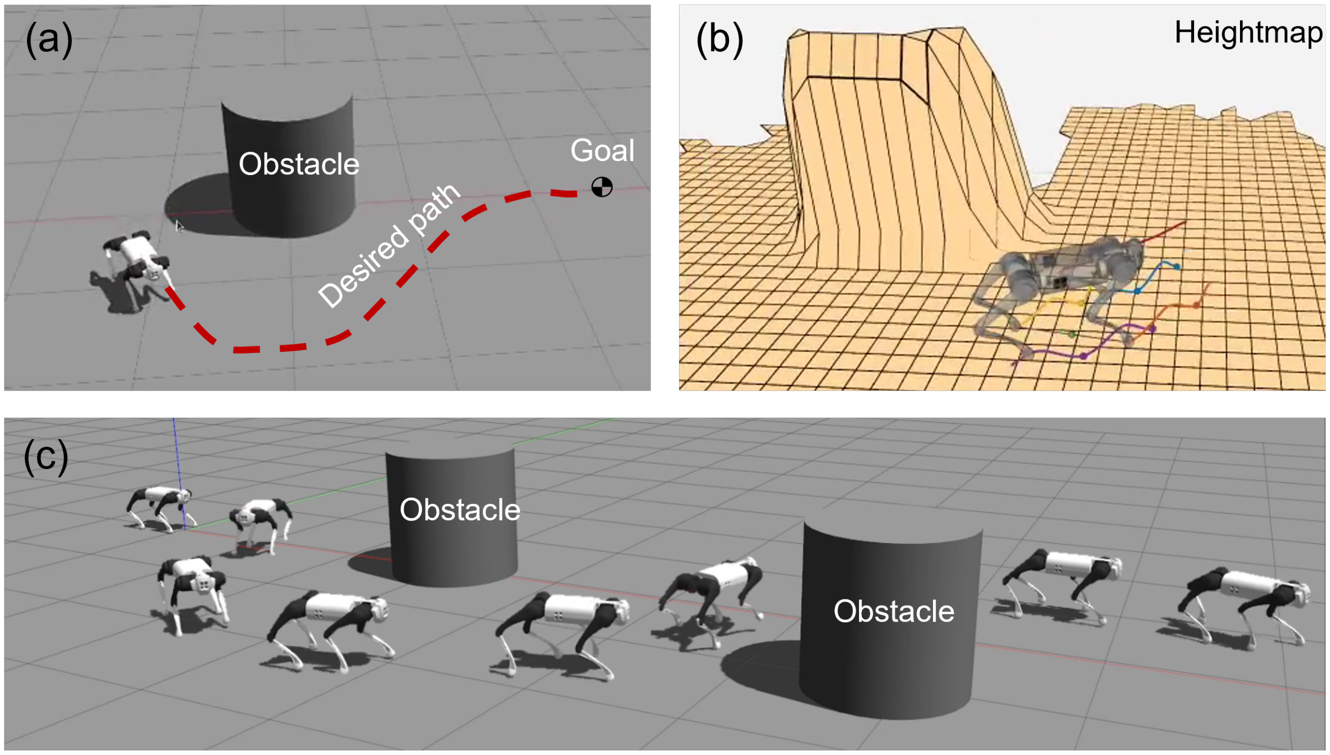

Fig. 4. Quadruped robot executing the proposed global plan. (a) Environment with single obstacle and (b) the corresponding heightmap representation. (c) Environment with two obstacles

A. *Global Planner Architecture*

The main contribution of this work is the integration of a generic motion planning architecture as seen in Figure 1 into the OCS2 framework [10]. The goal of the motion planning architecture is to create an interface that allows for any planner (e.g. A*, RRT, reactive planners, trajectory optimizer, etc.) to plan for a simplified robot model and convert it into a trajectory that is suitable to for the quadruped motion planning problem.

The architecture relies mainly on the body planner to handle the logic of receiving plans, mapping the plan to a centroidal model, and sending the lifted plan to a trajectory manager to handle trajectories as a reference to the finite state machine (FSM) and model predictive controller (MPC).

Additionally, due to the architecture of the body planner, where the body planner communicates to the planning algorithm using a ROS interface, it is simple to interchange different algorithms and transform the plan to locomotion for quadrupeds. In this work, we use the density navigation planner, which gives us a feedback planner for almost everywhere convergence while remaining inside the safe region. Details of the implementation in available on GitHub[1].

B. *Simulation Results*

Simulation results are shown using Gazebo with RVIZ as the visual interface, as shown in Figure 4. In the following experiment, we define a target set $\mathbf{X}_T = (10, 0)$. In Figure 4a, we set up an environment with a cylindrical obstacle $\mathbf{X}_u$ centered at (5,0) with a radius of $r = 1$, while in Figure 4c, we use two cylindrical obstacles centered at (3,0.1) and (7,-1) respectively. Using these parameters, a density function is constructed. The reference trajectory is obtained as a feedback control using (5). By forward integrating the feedback controller for a fixed horizon, our feedback controller is fed into the NMPC as a reference trajectory. Note that for an increase in performance, we use a first-order filter and a moving average filter to smoothen the trajectories generated by the density planner.

[1]Legged Planner Github Link

V. Limitations

Some limitations of the current framework are listed below

- The density-based planner works only for binary environments. There is no distinction for the degree of traversability and hence it cannot be easily extended to off-road navigation.
- This formulation only works for holonomic systems. Although, there are several ways to extend for non-holonomic cases.
- The formulation relies on global information for the environment and hence cannot incorporate local sensing information to form density functions online.
- The quality of trajectories obtained from the global planner can be improved by passing it through a trajectory optimizer.

VI. CONCLUSIONS

In this work, we develop a motion planning architecture for quadruped locomotion. We use density functions to design a safe reference trajectory for the robot. The trajectories are obtained as a positive gradient of this density function and can be implemented in closed loop. The proposed algorithm is integrated with a nonlinear MPC and low-level controller from the OCS2 framework. Simulation results show that the robot is able to track safe reference trajectories provided by the density-based motion planning framework. Future works will integrate a trajectory optimizer and extend this framework to non-holonomic systems and off road navigation.

REFERENCES

[1] Patrick M Wensing, Michael Posa, Yue Hu, Adrien Escande, Nicolas Mansard, and Andrea Del Prete. Optimization-based control for dynamic legged robots. *arXiv preprint arXiv:2211.11644*, 2022.

[2] Marco Tranzatto, Frank Mascarich, Lukas Bernreiter, Carolina Godinho, Marco Camurri, Shehryar Khattak, Tung Dang, Victor Reijgwart, Johannes Loeje, David Wisth, et al. Cerberus: Autonomous legged and aerial robotic exploration in the tunnel and urban circuits of the darpa subterranean challenge. *arXiv preprint arXiv:2201.07067*, 2022.

[3] Timothy H Chung, Viktor Orekhov, and Angela Maio. Into the robotic depths: Analysis and insights from the darpa subterranean challenge. *Annual Review of Control, Robotics, and Autonomous Systems*, 6, 2022.

[4] David E Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous robots*, 35:161–176, 2013.

[5] Jean-Pierre Sleiman, Farbod Farshidian, Maria Vittoria Minniti, and Marco Hutter. A unified mpc framework for whole-body dynamic locomotion and manipulation. *IEEE Robotics and Automation Letters*, 6(3):4688–4695, 2021.

[6] Moritz Diehl, Hans Georg Bock, and Johannes P Schlöder. A real-time iteration scheme for nonlinear optimization in optimal feedback control. *SIAM Journal on control and optimization*, 43(5):1714–1736, 2005.

[7] Umesh Vaidya. Optimal motion planning using navigation measure. *International Journal of Control*, 91(5):989–998, 2018.

[8] Loring W Tu. Manifolds. In *An Introduction to Manifolds*, pages 47–83. Springer, 2011.

[9] Anders Rantzer. A dual to Lyapunov's stability theorem. *Systems & Control Letters*, 42(3):161–168, 2001.

[10] Farbod Farshidian et al. OCS2: An open source library for optimal control of switched systems. [Online]. Available: `https://github.com/leggedrobotics/ocs2`.