2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Agile and Safe Trajectory Planning for Quadruped Navigation with Motion Anisotropy Awareness

Wentao Zhang[1], Shaohang Xu[1,2], Peiyuan Cai[1] and Lijun Zhu[1]

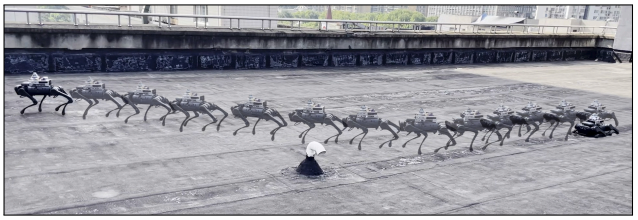
***Abstract*— Quadruped robots demonstrate robust and agile movements in various terrains; however, their navigation autonomy is still insufficient. One of the challenges is that the motion capabilities of the quadruped robot are anisotropic along different directions, which significantly affects the safety of quadruped robot navigation. This paper proposes a navigation framework that takes into account the motion anisotropy of quadruped robots including kinodynamic trajectory generation, nonlinear trajectory optimization, and nonlinear model predictive control. In simulation and real robot tests, we demonstrate that our motion-anisotropy-aware navigation framework could: (1) generate more efficient trajectories and realize more agile quadruped navigation; (2) significantly improve the navigation safety in challenging scenarios. The implementation is realized as an open-source package at https://github.com/ZWT006/agile_navigation.**

I. INTRODUCTION

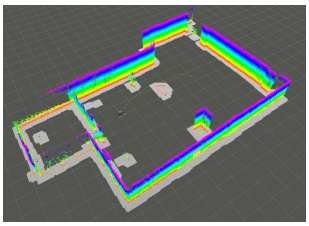
A quadruped robot is a bionic machine mimicking the locomotion of quadruped animals in nature. It is capable of omnidirectional movement and traversing complex terrains. Recent advancements in locomotion control for quadruped robots include model predictive control (MPC) [1]–[5] and learning-based approaches [6]–[10]. These works demonstrate the stable and agile locomotion over various terrains where the reference trajectory to be tracked is given by the user command.

The recent advancements have indeed facilitated robots in achieving a high level of locomotion in single-motion modes, encompassing movement speed, angular rate, jumping, and traversing rough terrains. Nonetheless, realizing agile motion akin to animals (involving multiple-motion modes at high speed) in the real world still poses a significant challenge for current quadruped robots.

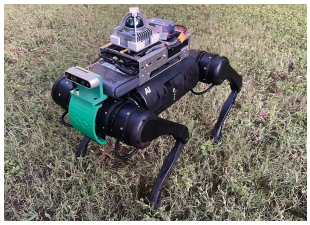
Analyzing data from routine motion experiments on quadruped robots, we identify omnidirectional motion anisotropy (OMA) as a key factor influencing their agile motion. This issue is not only observed in the Unitree A1 quadruped robot but also discussed in [11], which highlights the motion anisotropy of Cassie biped robot. This anisotropy stems from the mechanical design and joint motors, creating a coupling between its direction of motion and the magnitude, direction, and work distance of ground reaction forces (GRF) on the robot's legs. Therefore, we propose a hierarchical real-time navigation system that takes OMA into account, to achieve agile autonomous quadruped navigation in an unknown/known environment, as Fig. 1. Our experiments demonstrate that neglecting OMA awareness during trajectory planning leads to unstable locomotion when feeding the reference trajectory to the low-level controller.



(a) agile movements



(b) planning visualization



(c) system hardware

Fig. 1: (a) showcases our planning system guiding the robot to achieve agile movements in real-world environment. (b) depicts a demonstration of the robot's perception, planning, and control. (c) illustrates the actual hardware deployment of the entire planning system on Unitree A1.

A. Related Work

MPC is prevalent technique in legged robot locomotion control, often incorporating collision-free constraints, such as [12], [13]. The former is an NMPC-WBC (Whole-Body Control) controller incorporating Discrete Control Barrier Functions, while the latter treats both static and dynamic obstacles as soft inequality constraints. Nevertheless, the effectiveness of control is heavily influenced by the predictive horizon length and the optimization problem is prone to fall into local optima. [14] focus on person-following navigation in confined spaces through path search, trajectory optimization, and MPC, considering the robot's orientation for collision-free motion planning in geometric space while overlooking posture planning's impact on motion stability.

Learning-based methods have been proposed as [15]–[18]. [15] demonstrates obstacle avoidance in cluttered and dynamic environments only using depth camera images for learning input. [16] proposes a hierarchical navigation

[1]School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, China, wentaozhang@hust.edu.cn, shaohangxu@hust.edu.cn, m202173177@hust.edu.cn, ljzhu@hust.edu.cn

[2]School of Data Science, City University of Hong Kong, HKSAR, shaohanxu2-c@my.cityu.edu.hk

framework encompassing mapper, global planner, local planner, and command-tracking controller, yet real-world testing remains pending. Both [17] and [18] leverage neural networks to estimate robots' locomotion capabilities, adapting robot trajectories to dynamics and validating performance on the quadruped robot ANYmal in real-world experiments. However, learning-based approaches require pre-training or dataset. Notably, these datasets are predominantly sourced from simulated environments, potentially introducing discrepancies when transitioning to real-world applications.

Both MPC and RL controllers mentioned above require the operator to provide a trajectory. However, due to the lack of consideration for OMA, following the given trajectory may lead to poor tracking performance and even result in robot falls (as demonstrated in our subsequent experiments).

Multiple navigation frameworks based on sample-based or search-based path planning methods and optimization methods are proposed in [19]–[24]. [20] considers legged robot flight phases and utilizes Rapidly-Exploring Random Trees (RRT) to generate Center of Mass (COM) trajectories but it has not been tested on real robots. [21] presents a framework that encompasses visual processing, state estimation, path planning, and locomotion control. Nevertheless, it yields geometrically straight trajectories, lacking smoothness. [22] and [23] incorporate jumping into the planning process and have been tested in the real world, but with slow movement speeds.

The aforementioned planning approaches do not explicitly consider OMA, which may result in trajectories that are difficult for actual robots to track, such as fast lateral walking. Although [11] considers lateral stability in reactive planning, this reactive method does not take obstacle avoidance into account.

B. Contribution

We model the OMA of quadruped robots and introduce a real-time hierarchical navigation framework that specifically addresses the OMA in different level. The key contributions of our study can be outlined as follows:

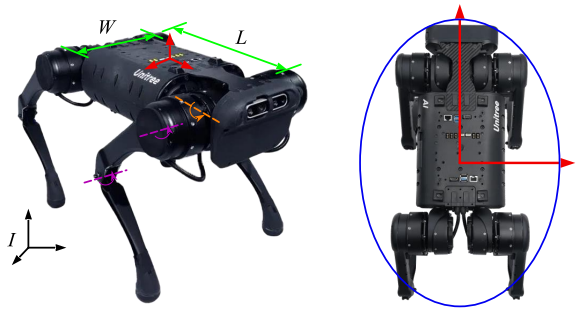
- An efficient kinodynamic trajectory generation method based on LazyPRM* is proposed, which takes the translational motion direction and orientation changes into explicit consideration. The Optimal Boundary Value Problem (OBVP) is employed to kinodynamic connect the state points.
- A spatiotemporal Nonlinear Trajectory Optimization (NTO) problem based on parameterized polynomial trajectory and nonlinear constraint is constructed to further smoothen the trajectory locally and make it dynamically feasible. The OMA is introduced as an elliptical constraint on the maximum translational velocity and motion direction.
- An NMPC with WBC locomotion controller is proposed to track the target COM state trajectory, while the penalty on maximum simultaneous linear and angular velocities are introduced to prevent potential falls.
- The proposed agile navigation framework is deployed on the Unitree A1 robot platform in simulations and real-world experiments with unknown environments. We also released it as ros packages, in order to facilitate research communication in relevant fields.

The remainder of this article is organized as follows. The hierarchical planning is elaborated in detail in Section II. In Section III, we integrate perception into the navigation framework and deploy it on the A1 robot platform for real-world implementation. We analyze and evaluate our method through simulations and experiments. Section IV concludes the paper.

II. HIERARCHICAL PLANNER METHOD

A. Omnidirectional Motion Anisotropy

A quadruped robot relies on the GRF generated through leg-ground contact to enable movement. The resultant force of the GRF can act in any direction, enabling the robot to achieve omnidirectional motion. Additionally, both the turning and translation motions of the robot depend on the combined forces of GRF, leading to a coupling effect between turning and translation. Consequently, high turning speed and high translation speed cannot be achieved simultaneously. The analysis of OMA in a quadruped robot is shown in Fig. 2. Each leg consists of three joints: the shoulder joint provides lateral movement by roll motion, while the hip and knee joints enable pitch motion for forward and backward locomotion. Furthermore, the distance L is larger than the distance W, resulting in unequal work distance in the two directions during one gait contact phase.



(a) leg joint (b) motion capability

Fig. 2: I represents the world inertial coordinate system, and B represents the robot coordinate system. (a) illustrates the drivetrain of a single leg. The orange color represents the shoulder joint, while the purple color indicates the hip and knee joints. The distance between the left and right legs is denoted as W, and the distance between the front and back legs is represented by L. (b) depicts the robot's motion capabilities in different directions using a blue ellipse. It can be interpreted as the maximum acceleration, maximum velocity, and the farthest distance covered in one gait contact cycle. This feature can be used to guide legged robots' trajectory planning, foot placement planning, and dynamics control

B. Kinodynamic Trajectory Generation

This algorithmic aims to rapidly generate a coarse robot COM trajectory, denoted as Γ, taking into account both translational and turning movements of the robot by separately planning the plane position x, y and yaw angle θ. Unlike conventional planning methods that consider $\theta = \text{atan2}(\dot{x}, \dot{y})$ as described in [20], planning $[x, y, \theta]$ separately is crucial to overcome OMA. The algorithm utilizes LazyPRM* [25], which is a variant of PRM* [26], checking connectivity only when those two nodes are part of the optimal solution, increasing search efficiency.

Alg.1 outlines the key steps. $\mathcal{O}$ and $\mathcal{C}$ refer to the open and closed set of A* [27] and *RoadMap* is the set of state sample points. **Expand**$(\cdot)$ searches neighboring nodes n_c around the parent node n_n within a given range, and saves their index sequences in *nearsets*. **CollisionFree**$(\cdot)$ checks the safety of trajectory. **Update**$(\cdot)$ is the normal A* search update process. **Solve**$(\cdot)$ is used to solve the OBVP to connect two state points as explained in Section II-B.2.

Algorithm 1: Kinodynamic LazyPRM*

Input : Obstacle map *Map*, Start pose $\mathbf{p}_{\text{start}}$, Goal pose $\mathbf{p}_{\text{goal}}$
Output: A trajectory Γ from $\mathbf{p}_{\text{start}}$ to $\mathbf{p}_{\text{goal}}$

1 **for** $idx \leftarrow 0$ **to** $idx_{\max}$ **do**
2 **for** $idy \leftarrow 0$ **to** $idy_{\max}$ **do**
3 $\mathbf{p}(x) = idx \cdot \delta_g + rand(-1,1) \cdot \delta_b$;
4 $\mathbf{p}(y) = idy \cdot \delta_g + rand(-1,1) \cdot \delta_b$;
5 $RoadMap[idx, idy] \leftarrow \mathbf{p}$;
6 Initialize();
7 **for** $iter \leftarrow 1$ **to** $MaxIter$ **do**
8 **if** $\mathcal{O}$.**empty then**
9 **return**;
10 $n_n \leftarrow \mathcal{O}$.**pop**$()$;$\mathcal{C}$.**insert**$(n_n)$;
11 **if** $n_n.\mathbf{p}$ == $\mathbf{p}_{\text{goal}}$ **then**
12 $\Gamma \leftarrow$ **GetPath**$(\mathcal{C}, n_n)$;
13 **return** Γ;
14 $x_{\text{par}} \leftarrow n_n.\mathbf{p}(x)$; $y_{\text{par}} \leftarrow n_n.\mathbf{p}(y)$;
15 $nearsets \leftarrow$ **Expand**$(RoadMap, n_n.\mathbf{p})$;
16 **for** $idx.idy \leftarrow nearsets$ **do**
17 $\mathbf{p} \leftarrow RoadMap[idx, idy]$;
18 **if** $n_n.\mathbf{p}(x,y)$ == $\mathbf{p}_{\text{goal}}(x,y)$ **then**
19 $\mathbf{p}(\theta)$ = $\mathbf{p}_{\text{goal}}(\theta)$;
20 **else**
21 $\mathbf{p}(\theta) = \text{atan2}\left((y - y_{\text{par}}) / (x - x_{\text{par}})\right)$;
22 $n_c \leftarrow$ **Solve**$(\mathbf{p}, n_n)$;
23 **if CollisionFree**(n_c) **then**
24 **Update**$(n_c, n_n, \mathcal{C}, \mathcal{O})$;

1) RoadMap Sample: A quadruped robot has 6 Degrees of Freedom (DoF), but only the horizontal position and orientation angle are mainly considered during the locomotion, since body height, pitch angle, and roll angle are relevant to the terrain. To enhance efficiency, we replaced random sampling in LazyPRM* by uniformly distributed sampling with random perturbation as described in Alg. 1 lines 1 to 5. In this scheme, the entire plane space is uniformly discretized into $idx_{\max} \times idy_{\max} = n$ grids, each grid associated with a state point stored in the *RoadMap*. The state of grid is represented as $RoadMap[idx, idy] = [x, y, \theta]$, where idx and idy are the index of grid. δ_g is grid size and δ_b is the maximum bias magnitude. Initially, the yaw angle θ is set to zero and is adjusted during the expanding process based on the states of the parent node and the current node (see Alg. 1 lines 18 to 21). By using the grid indices idx and idy to access state points during the search expansion, the time complexity of finding the optimal trajectory is $\mathrm{O}(n)$. In contrast, if a completely random sampling approach is used for the entire space and Dijkstra's method is employed to find the optimal solution, the time complexity increases to $\mathrm{O}(n^2)$.

2) Kinodynamic Connection: The motion of a quadruped robot is dependent on GRF, which provides acceleration, so we express the simplified motion equation of COM as a double integration system as

$$\begin{aligned} \dot{\mathbf{s}} &= \mathbf{A}\mathbf{s} + \mathbf{B}\mathbf{u}, \\ \mathbf{A} &= \begin{bmatrix} 0 & \mathbf{I}_3 \\ 0 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 0 \\ \mathbf{I}_3 \end{bmatrix}, \\ \mathbf{s} &= [\mathbf{p}, \mathbf{v}]^{\mathrm{T}} = [\mathbf{p}, \dot{\mathbf{p}}]^{\mathrm{T}}, \\ \mathbf{u} &= \mathbf{a} = \ddot{\mathbf{p}}, \end{aligned}$$

where $\mathbf{s} = [\mathbf{p}, \mathbf{v}], \mathbf{p} = \left[p_x, p_y, p_\theta\right]$ is state variables, system input $\mathbf{u}$ is fiction force that can be generated by the low-level controller via GRF.

To enhance the consistency between the coarse trajectory and the robot's motion, we formulate the kinodynamic connection between two nodes as an OBVP that minimizes the total trajectory energy $J(T) = \int_0^T \mathbf{a}(t)^2 dt$, with given initial state $\mathbf{s}_i = [\mathbf{p}_i, \mathbf{v}_i]$, fixed final position state $\mathbf{p}_f$, and free final velocity state $\mathbf{v}_f$. We employ Pontryagin's maximum principle [28] to obtain explicit solutions as:

$$\begin{aligned} \begin{bmatrix} p_\mu^*(t) \\ v_\mu^*(t) \end{bmatrix} &= \begin{bmatrix} \frac{\alpha_\mu}{6}t^3 - \frac{\alpha_\mu T}{2}t^2 + v_{\mu i}t + p_{\mu i} \\ \frac{\alpha_\mu}{2}t^2 - \alpha_\mu T t + v_{\mu i} \end{bmatrix}, \\ u_\mu^*(t) &= \alpha_\mu t - \alpha_\mu T, \\ J^*(T) &= \sum_{\mu \in \{x,y,\theta\}} \left(\frac{1}{3}\alpha_\mu^2 T^3\right), \\ \alpha_\mu &= -\frac{p_{\mu f} - v_{\mu i}t + p_{\mu i}}{3}. \end{aligned} \tag{1}$$

To numerically calculate the solution, the value of trajectory duration time T is needed, which we specify as $T = T_{\text{ref}} := \max\left(\left\|[x_i - x_f, y_i - y_f]\right\|_2 / v_{\text{ref}}, |\theta_i - \theta_f| / \omega_{\text{ref}}\right)$. v_{ref} denotes the reference linear velocity and ω_{ref} represents the reference angular rate. It is worth noting that the optimal trajectory in this case is a polynomial trajectory, which we also exploit in the trajectory optimization problem.

3) Trajectory Cost: The yaw angle and linear velocity direction have a significant impact on the stability of robot motion. Therefore, we incorporate yaw angle cost to prevent short translation with huge direction change situation. The trajectory cost is

$$t_c = \lambda_{yaw} \cdot |\theta_i - \theta_f|^2 + \int^T \sqrt{\delta x^2 + \delta y^2}, \tag{2}$$

where the first term represents yaw cost with weight coefficient λ_{yaw} and the second term is the length of the trajectory which is calculate by the differentiation of state variables δx and δy. We utilize a quadratic form to promote smoother yaw angle variations. The effect of trajectory cost is shown in Fig. 3. The reference linear/angular velocity v_{ref} and ω_{ref} impact the maximum velocity. The reference time T_{ref} and angle cost weight λ_{yaw} influences trajectory smooth. It is essential to highlight that, unlike many existing approaches, our method specifically incorporates yaw information in the trajectory calculation, enabling the consideration and enforcement of the OMA.

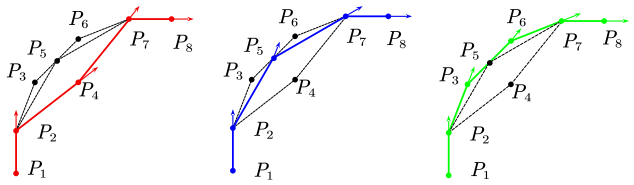


Fig. 3: Three trajectories are illustrated connecting P_1 and P_8. Arrows indicate the yaw angle direction. The translational distance is consistent between (a) and (b), but (b) has a smaller angular variation, demonstrating the effect of introducing an angle cost in trajectory cost. The angular variation is consistent between (b) and (c), but the quadratic angle cost results in a smaller angular cost for (c), demonstrating the effect of using a quadratic angle cost in trajectory cost. In conclusion, (2) guides the generation of smooth trajectories.

C. Nonlinear Trajectory Optimization

The rough trajectory Γ may be dynamically infeasible, as we use double integrate system to connect waypoints. Moreover, the trajectory duration T in (1) is determined by the reference velocity, which may not fully exploit the robot's motion capabilities. To address these limitations, we introduce an NTO problem to further improve the quality of Γ. Taking inspiration from [29]–[31], we still parameterize trajectory as polynomial but increase its order to improve its flexibility and accuracy.

1) Optimization Problem: The optimization problem is formulated with the polynomial parameters $\mathbf{c}$ and the duration time $\mathbf{T}$ serving as the decision variables, as below:

$$\min_{\mathbf{c},\mathbf{T}} \quad \sum_{j=1}^{N} \begin{bmatrix} \lambda_s \int_0^{T_j} \mathbf{u}_j(\mathbf{c}_j)^T R \mathbf{u}_j(\mathbf{c}_j) dt + \\ \lambda_c \phi(\mathbf{s}_j(t)) + \lambda_t \rho(T_j) \end{bmatrix} \tag{3a}$$

$$\text{s.t.} \quad \mathbf{s}(t) = \sum_{i=0}^{n} \mathbf{c}_{ji} \cdot t^i, t \geqslant 0, \forall t \in [0, T_j], j = 0 \sim N \tag{3b}$$

$$G(\mathbf{s}(t), \dot{\mathbf{s}}(t)) \preceq 1 \tag{3c}$$

$$H(\dot{\mathbf{s}}(t), \ddot{\mathbf{s}}(t)) \preceq 0 \tag{3d}$$

$$\mathbf{s}_{j-1}^{(m)}(T_{j-1}) = \mathbf{s}_j^{(m)}(0), m = 0 \sim M, j = 1 \sim N \tag{3e}$$

where $\mathbf{s}(t)$ is the nth orders polynomial trajectory of $[x, y, \theta]$, N represents the segment number of trajectories, which is one less than the amount of waypoints, j denotes the jth segment, R is a positive definite diagonal matrix judge states cost, $\mathbf{T}$ is the duration time vector, and $\mathbf{c}_{ji}$ represents the ith order coefficient of the jth segment polynomial.

2) Objective Function: The nonlinear optimization problem considers trajectory energy, obstacle avoidance, and time optimality by incorporating cost functions (3a) with weights λ_s, λ_c, and λ_t for each factor, respectively. The jth segment trajectory energy cost is described as

$$\int_0^{T_j} \mathbf{u}_j(\mathbf{c}_j)^T R \mathbf{u}_j(\mathbf{c}_j) dt. \tag{4}$$

Note that $\mathbf{c}_j = \{\mathbf{c}_j 1, \cdots, \mathbf{c}_{jn}\}$, $\mathbf{c} = \{\mathbf{c}_1, \cdots, \mathbf{c}_N\}$, and the controller input $\mathbf{u}_j$ is written as a function of variables $\mathbf{c}_j$.

We discretize the spatial trajectory and calculate the obstacle cost to ensure obstacle avoidance as

$$\begin{aligned} \phi(s_j(t)) &= \int_0^{T_j} f_d(d(p(t))) ds \\ &\approx \sum_{l=0}^{T_j/\delta t} f_d(d(p(T_{jl}))) v(T_{jl}) \delta t, \ T_{jl} = l \cdot \delta t, \end{aligned} \tag{5}$$

where d_s is the differentials of the arc lengths of the $[x, y]$ trajectories, and $f_d(\cdot)$ is the distance penalty function. For instance, $f_d(d(p(t))) = 1/d(p(t))$. Where $d(p(t)) < d_{th}$, $d(p(t))$ is the Euclidean Distance Field (EDF) [32] at point $p(t)$, and d_{th} is the threshold of obstacle clearance.

The time penalty function is defined as

$$\begin{aligned} \rho(T_j) &= |T_j|, T_j > 0 \\ &= e^{\tau_j}, \tau_j \in \mathbb{R}, \end{aligned} \tag{6}$$

where the time vector $\mathbf{T}$ is positive real numbers. We introduce intermediate variable τ as real number to transform time into an unconstrained optimization variable.

3) System Constraints: The OMA is considered as constraint terms in (3c). As shown in Fig. 2(b), we use **elliptical** constraint to cater this feature, which is explicit and easily analyzed without introducing too much complexity into the optimization problem. The maximum translational velocity constraint is

$$\begin{bmatrix} v_{Bx} \\ v_{By} \end{bmatrix} = R(\theta) \begin{bmatrix} v_{Ix} \\ v_{Iy} \end{bmatrix}, \\ \frac{v_{Bx}^2}{v_{mx}} + \frac{v_{By}^2}{v_{my}} \leqslant 1, \tag{7}$$

where θ is body yaw angle and $R(\theta)$ is rotation matrix from world inertial frame I to robot body frame B. The maximum frontal and lateral velocity are denoted as v_{mx} and v_{my} respectively, with $v_{\mathrm{my}} < v_{\mathrm{mx}}$ indicating that frontal movement capability is stronger than lateral. (3d) enforces velocity and acceleration limits $|\dot{\mathbf{s}}(t)| - \mathbf{v}_{\max} \leqslant 0, |\ddot{\mathbf{s}}(t)| - \mathbf{a}_{\max} \leqslant 0$. Continuity constraints (3e) ensure smooth transitions between waypoints, where M is the continuous state order.

D. NMPC Trajectory Tracking

We use the optimized trajectory as the reference trajectory in the NMPC framework and introduce velocity safety constraints to ensure the stability during fast motion.

1) NMPC Problem: The general formulation of NMPC problem is derived from [33] as

$$\min_{\mathbf{u}(t)} \quad \Phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t), t)\mathrm{d}t \tag{8a}$$

$$\text{s.t.} \quad \mathbf{x}(0) = \hat{\mathbf{x}} \tag{8b}$$

$$\dot{\mathbf{x}} = \mathbf{f}^c(\mathbf{x}, \mathbf{u}, t) \tag{8c}$$

$$\mathbf{g}(\mathbf{x}, \mathbf{u}, t) = \mathbf{0} \tag{8d}$$

$$\mathbf{h}(\mathbf{x}, \mathbf{u}, t) < \mathbf{0} \tag{8e}$$

where $\mathbf{x}(t)$ and $\mathbf{u}(t)$ are the state and input at time t, $\Phi(\cdot)$ is final state cost and $L(\cdot)$ is quadratic cost function regarding the tracking error of the state and input. $\hat{\mathbf{x}}$ represent current measured state. $\mathbf{f}^c(\cdot)$ represents system dynamic function with further details provided in [34] as the floating base dynamic equation. $\mathbf{g}(\cdot)$ and $\mathbf{h}(\cdot)$ represent equality constraints and inequality constraints, respectively. The robot state vector is

$$\mathbf{x} = \left[\theta_{\mathrm{B}}^{\mathrm{T}}, \mathbf{p}_{\mathrm{B}}^{T}, \omega_{\mathrm{B}}^{T}, \mathbf{v}_{\mathrm{B}}^{T}, \mathbf{q}_j^{\mathrm{T}}\right]^{\mathrm{T}}, \mathbf{u} = \left[\mathbf{f}_{\mathrm{c}}^{\mathrm{T}}, \mathbf{v}_j^{\mathrm{T}}\right]^{\mathrm{T}},$$

where B represents the robot body frame. In particular, the system state $\mathbf{x}$ contains centroidal orientation as Euler angle $\theta_{\mathrm{B}} \in \mathbb{R}^3$, position $\mathbf{p}_{\mathrm{B}} \in \mathbb{R}^3$, angular rate ω_{B}, linear velocity $\mathbf{v}_{\mathrm{B}}$ and joint position $\mathbf{q}_j \in \mathbb{R}^{12}$. System input $\mathbf{u}$ consists of four feet contact forces $\mathbf{f}_{\mathrm{c}} \in \mathbb{R}^4$ and joint velocity $\mathbf{v}_j$.

2) Safety Constraint: During high-speed motion, we introduce a constraint to prevent simultaneous large values of translational and angular velocity. This constraint is imposed to avoid instability and potential falls as follows:

$$h_{\mathrm{s}} = \lambda_{\mathrm{l}} \cdot \left(v_{\mathrm{Bx}}^2 + v_{\mathrm{By}}^2\right) + \lambda_\theta \cdot \omega_{\mathrm{B}\theta}^2 - h_{\mathrm{s}}^{th} < 0,$$

where the weights λ_{l} and λ_θ govern the trade-off between translational and angular velocity, while h_{s}^{th} represents safety constraint threshold.

III. Experiments and Results

Our planning method is subjected to numerical computations, physical simulations, and real-world experiments. The search and optimization components are numerical computation tested in MATLAB. Our planning framework is implemented in C++ with ROS as a communication middleware.

A. Planning System Implementation

The overall navigation system for real-time navigation experiments is illustrated in Fig. 4. LiDAR-inertial odometry [35] is a fast, robust, and versatile framework, that provides 10 Hz high-accuracy odometry and global pointcloud data

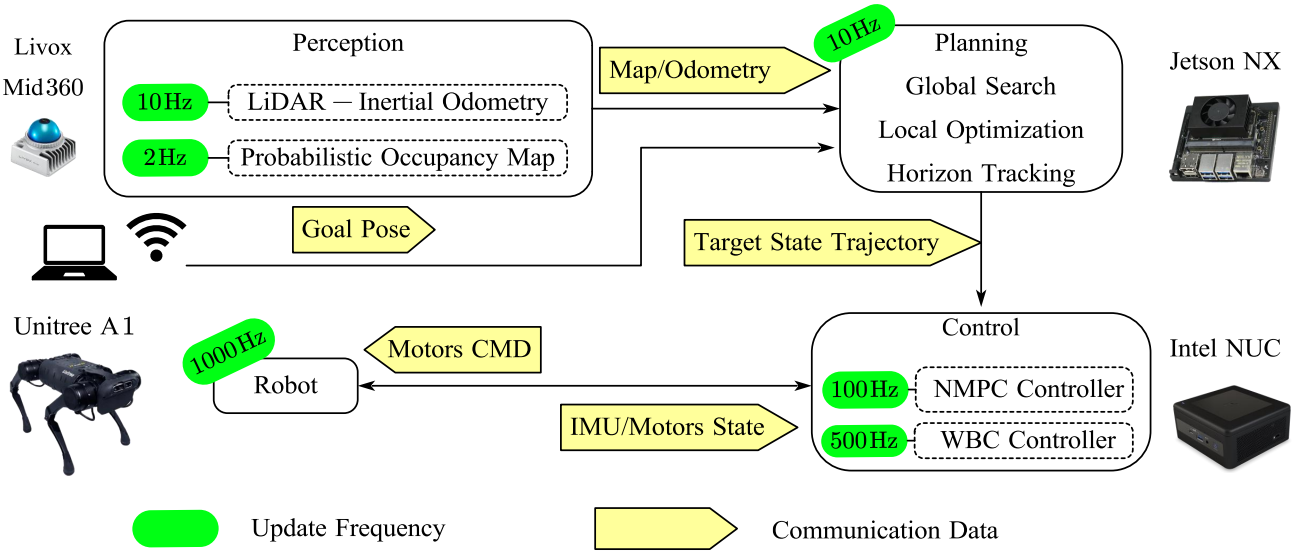


Fig. 4: This figure summarizes the proposed planning system. The perception and planning board is Jetson Xavier NX with Graphics Processing Unit (GPU). Hybrid solid-state lidar Livox Mid-360 can provide 360° horizontal, 52° vertical FOV and 100 m detection range. The control board is Intel NUC 11PAHi7-1165G7 with 16G RAM, which runs the locomotion controller NMPC-WBC. Quadruped robot Unitree A1 is the experimental platform. The hardware devices are all connected within the same Local Area Network (LAN) and communicate by the User Datagram Protocol (UDP).

with lower computation. We use [32] to update the local probabilistic occupancy map centered on the robot at 2 Hz, which is not sensitive to dynamic obstacles and reduces the impact of dynamic point clouds on planning. We use OpenCV to calculate 2D EDF map from global occupancy map for path search and trajectory optimization. The locomotion controller[1] is responsible for robot locomotion control and trajectory tracking, and is an NMPC-WBC controller implemented by OCS2 [36], operates at 100 Hz for NMPC and 500 Hz for WBC.

B. Optimization Solving

For numerical solving of NTO problem, we utilize the general optimization solver NLopt[2] and employ the gradient-based local optimization method MMA(Method of Moving Asymptotes) [37]. We utilize the parameters of Γ as an initial guess to facilitate solving the NTO problem. In practical testing, it is nearly impossible to solve the problem without an initial guess. We also manually solve the equality constraints during the solving iterative process and replace inequality constraints with penalty function, transforming NOT into an unconstrained optimization problem to improve solving efficiency. This approach enables the application in real-time planning for solving as 10 Hz.

C. Simulation

We conduct planning simulations in Gazebo, using a corridor obstacle map with size of 30[m]×8[m] and a local square perception radius of 6 m. Our planner is set to $v_{\text{ref}} = 1.5\,\text{m/s}$ and $\omega_{\text{ref}} = 1.0\,\text{rad/s}$ for comparison with champ [38], which employs GMapping [39] for SLAM and OMPL [40] for planning. As depicted in Tab. I, our proposed method achieves significantly higher linear velocity and angular rate, while reducing navigation time by 36%.

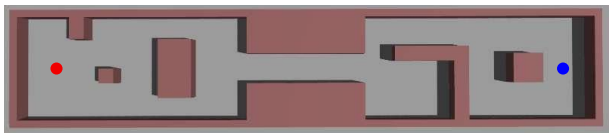


Fig. 5: The simulation map in Gazebo. It is a cluttered corridor with random obstacles. The robot is required to navigate through the corridor from the red point to the

TABLE I: Simulation Result

Method	Trajectory State				
	time[s]	v_{ave}[m/s]	ω_{ave}[rad/s]	v_{max}[m/s]	ω_{max}[rad/s]
baseline	111.1	0.2468	0.0905	0.6012	0.7785
proposed	**40.1**	**1.0024**	**0.5673**	**1.6819**	**3.0731**

[1]legged_control:https://github.com/qiayuanliao/legged_control
[2]Steven G. Johnson NLopt: http://github.com/stevengj/nlopt

D. Legged Robot Omnidirectional Constraint

We simulate a typical scenario in which the robot must pass through a doorway laterally to evaluate OMA for legged robots. Four experiments are conducted: optimized trajectory planning (NO-T), optimized trajectory planning without yaw angle (NOY-T), search trajectory planning (KD-T), and search trajectory planning without yaw angle (KDY-T). As shown in Fig. 6, our optimization techniques effectively improve trajectory quality. It is evident from Fig. 6(f) that the (KD-T) exhibits significant tracking errors. Nevertheless, the optimized trajectory demonstrates superior tracking performance and reaches the destination more quickly in Fig. 6(d). Notably, neglecting planning for the yaw angle renders the robot highly susceptible to falling down under the same dynamic bounds, as evidenced by Fig. 6(e) and Fig. 6(g). Further experimental details can be found in the supplementary video.

E. General Planning Experiments

We conduct tests in three different real-world scenarios, as depicted in Fig. 7. The robot successfully navigates through obstacles and efficiently reaches the desired positions and orientations, demonstrating its flexibility. Furthermore, the use of probabilistic obstacle maps enhances the robustness of the planning system against the influence of moving obstacles. The results of our real-world planning experiments are presented in Tab. II. Our planning framework proves to be adaptable to uneven grasslands while maintaining stability.

TABLE II: General Planning Result

Scenario	$\mathbf{p}_{\text{goal}}$	Trajectory State				
		time[s]	v_{ave}[m/s]	ω_{ave}[rad/s]	v_{max}[m/s]	ω_{max}[rad/s]
square	[4.0, -4.0, 0.0] [2.0, -0.5, 0.0]	19.3	0.56	0.50	1.44	-2.07
grass	[0.0, -8.0, 0.0] [2.0, 0.0, 0.0]	31.3	0.56	0.44	1.91	-1.62
corridor	[16.0, 0.0, 0.0]	40.7	0.45	0.29	2.87	-2.22

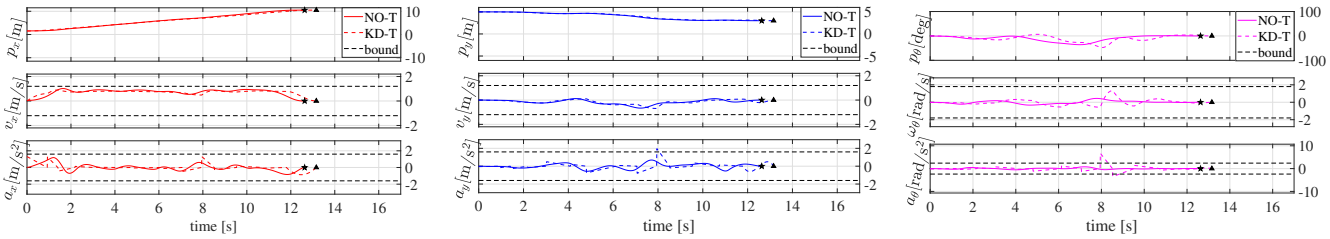
IV. Conclusion

In this paper, we propose a real-time hierarchical planning system for quadruped robots. We consider the anisotropy in the omnidirectional motion at different planning levels. Our navigation system enables agile motion in unknown environments and has been tested in simulations and real-world on the Unitree A1 robot platform.

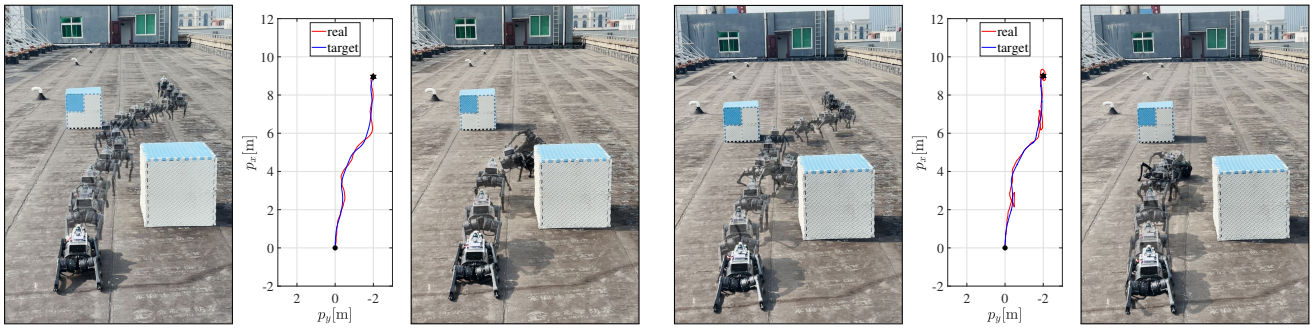
In the future, we plan to expand the map of perception from 2D to 2.5D or even 3D, enabling planner to tackle high-dimensional navigation challenges. Additionally, we aim to incorporate footsteps and swinging legs in local planning, while enhancing the performance of the NMPC controller and addressing uneven terrain. Our ultimate objective is to achieve more flexible and adaptable planning and control in highly complex scenarios through ongoing research efforts.

ACKNOWLEDGMENT

This work is supported under the National Natural Science Foundation of China under Grant 62173155 and the Program for HUST Academic Frontier Youth Team.

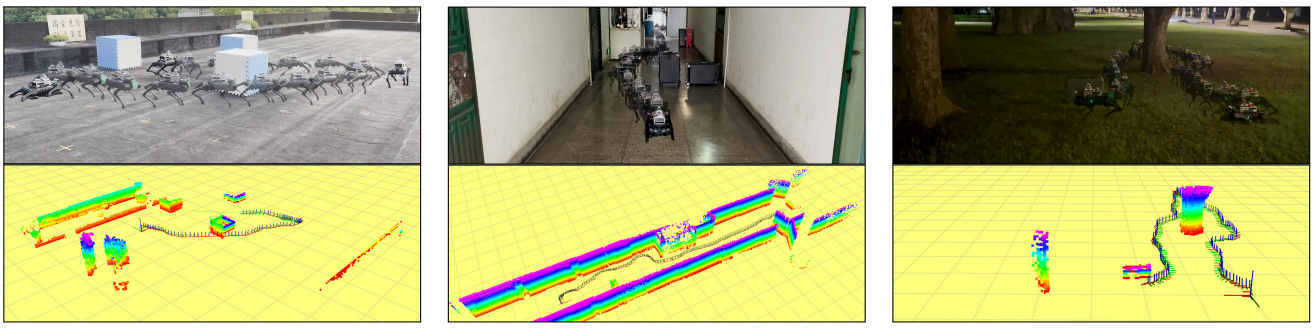


(a) x state trajectory (b) y state trajectory (c) θ state trajectory



(d) NO-T (e) NOY-T (f) KD-T (g) KDY-T

Fig. 6: (a), (b) and (c) are COM state trajectories. Both the KD-T and the NO-T exhibit continuous and smooth state transitions. Furthermore, the NO-T demonstrates a reduced time duration while satisfying the ellipse constraints. The five-pointed star serves as the destination for NO-T, while the triangle represents the endpoint for KD-T. Goal pose is [9.0,-2.0,0.0] and the ellipse velocity constraint v_{mx} is 2.4 m/s and v_{my} is 1.2 m/s. The v_{ref} is 0.8 m/s and ω_{ref} is 1.2 rad/s, with $v_{\max}$ bound at $\pm$1.2 m/s and $\omega_{\max}$ bound at $\pm$1.8 rad/s. The $a_{\max}$ is $\pm$1.6 m/s^2 and $\dot{\omega}_{\max}$ is $\pm$2.4 rad/s^2. (d) NO-T trajectory time duration is 12 s, while (f) KD-T is 21.26 s, (e) and (g) planning without yaw both falling over.



(a) square with obstacles (b) cluttered corridor (c) grass with trees

Fig. 7: Online planning experiments in three different scenarios. The figure above represents the actual navigation, while the figure below illustrates the information about SLAM and planning. The three-axis coordinates depict the robot's odometry.

REFERENCES

[1] 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.

[2] 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.

[3] D. Kang, F. De Vincenti, N. C. Adami, and S. Coros, "Animal motions on legged robots using nonlinear model predictive control," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 955–11 962.

[4] M. Bjelonic, R. Grandia, M. Geilinger, O. Harley, V. S. Medeiros, V. Pajovic, E. Jelavic, S. Coros, and M. Hutter, "Offline motion libraries and online mpc for advanced mobility skills," *The International Journal of Robotics Research*, vol. 41, no. 9-10, pp. 903–924, 2022.

[5] S. Xu, L. Zhu, H.-T. Zhang, and C. P. Ho, "Robust convex model predictive control for quadruped locomotion under uncertainties," *IEEE Transactions on Robotics*, 2023.

[6] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.

[7] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *arXiv preprint arXiv:2004.00784*, 2020.

[8] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[9] J. Wu, G. Xin, C. Qi, and Y. Xue, "Learning robust and agile

legged locomotion using adversarial motion priors," *IEEE Robotics and Automation Letters*, 2023.

[10] K. Caluwaerts, A. Iscen, J. C. Kew, W. Yu, T. Zhang, D. Freeman, K.-H. Lee, L. Lee, S. Saliceti, V. Zhuang, *et al.*, "Barkour: Benchmarking animal-level agility with quadruped robots," *arXiv preprint arXiv:2305.14654*, 2023.

[11] J.-K. Huang and J. W. Grizzle, "Efficient anytime clf reactive planning system for a bipedal robot on undulating terrain," *IEEE Transactions on Robotics*, 2023.

[12] Q. Liao, Z. Li, A. Thirugnanam, J. Zeng, and K. Sreenath, "Walking in narrow spaces: Safety-critical locomotion control for quadrupedal robots with duality-based optimization," *arXiv preprint arXiv:2212.14199*, 2022.

[13] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-free mpc for legged robots in static and dynamic scenes," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8266–8272.

[14] Z. Zhang, J. Yan, X. Kong, G. Zhai, and Y. Liu, "Efficient motion planning based on kinodynamic model for quadruped robots following persons in confined spaces," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 4, pp. 1997–2006, 2021.

[15] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter, "Learning a state representation and navigation in cluttered and dynamic environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5081–5088, 2021.

[16] Y. Kim, C. Kim, and J. Hwangbo, "Learning forward dynamics model and informed trajectory sampler for safe quadruped navigation," *arXiv preprint arXiv:2204.08647*, 2022.

[17] L. Wellhausen and M. Hutter, "Rough terrain navigation for legged robots using reachability planning and template learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 6914–6921.

[18] B. Yang, L. Wellhausen, T. Miki, M. Liu, and M. Hutter, "Real-time optimal navigation planning using learned motion costs," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9283–9289.

[19] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1184–1189.

[20] J. Norby and A. M. Johnson, "Fast global motion planning for dynamic legged robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 3829–3836.

[21] T. Dudzik, M. Chignoli, G. Bledt, B. Lim, A. Miller, D. Kim, and S. Kim, "Robust autonomous navigation of a small-scale quadruped robot in real-world environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 3664–3671.

[22] S. Gilroy, D. Lau, L. Yang, E. Izaguirre, K. Biermayer, A. Xiao, M. Sun, A. Agrawal, J. Zeng, Z. Li, *et al.*, "Autonomous navigation for quadrupedal robots with optimized jumping through constrained obstacles," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 2132–2139.

[23] M. Chignoli, S. Morozov, and S. Kim, "Rapid and reliable quadruped motion planning with omnidirectional jumping," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6621–6627.

[24] E. Jelavic, K. Qu, F. Farshidian, and M. Hutter, "Lstp: Long short-term motion planning for legged and legged-wheeled systems," *IEEE Transactions on Robotics*, 2023.

[25] K. Hauser, "Lazy collision checking in asymptotically-optimal motion planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2951–2957.

[26] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[27] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[28] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[29] Z. Wang, X. Zhou, C. Xu, and F. Gao, "Geometrically constrained trajectory optimization for multicopters," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3259–3278, 2022.

[30] P. Foehn, A. Romero, and D. Scaramuzza, "Time-optimal planning for quadrotor waypoint flight," *Science Robotics*, vol. 6, no. 56, p. eabh1221, 2021.

[31] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3529–3536, 2019.

[32] L. Han, F. Gao, B. Zhou, and S. Shen, "Fiesta: Fast incremental euclidean distance fields for online motion planning of aerial robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4423–4430.

[33] R. Grandia, F. Jenelten, S. Yang, F. Farshidian, and M. Hutter, "Perceptive locomotion through nonlinear model-predictive control," *IEEE Transactions on Robotics*, 2023.

[34] C. D. Bellicoso, C. Gehring, J. Hwangbo, P. Fankhauser, and M. Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 558–564.

[35] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[36] F. Farshidian *et al.*, "OCS2: An open source library for optimal control of switched systems," [Online]. Available: https://github.com/leggedrobotics/ocs2.

[37] K. Svanberg, "A class of globally convergent optimization methods based on conservative convex separable approximations," *SIAM Journal on Optimization*, vol. 12, no. 2, pp. 555–573, 2002. [Online]. Available: https://doi.org/10.1137/S1052623499362822

[38] J. M. Jimeno *et al.*, "CHAMP: Quadruped robot based on mit cheetah i," [Online]. Available: https://github.com/chvmp/champ.

[39] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[40] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.