Received 4 March 2024, accepted 13 May 2024, date of publication 16 May 2024, date of current version 23 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3401827

RESEARCH ARTICLE

Safe and Robust Motion Planning for Autonomous Navigation of Quadruped Robots in Cluttered Environments

HONGYI LIU[1] **AND QUAN YUAN**[2], **(Member, IEEE)**
[1]Department of Electronic Engineering, School of Information Science and Technology, Fudan University, Shanghai 200433, China
[2]Research Institute of Product Planning and Innovative Automotive Technology (GHY), BYD Company Ltd., Shenzhen 518083, China

Corresponding author: Hongyi Liu (20210720027@fudan.edu.cn)

ABSTRACT Quadruped robots, with their superior terrain adaptability and flexible movement capabilities, demonstrate greater application potential in complex environments compared to traditional ground robots. However, their non-negligible body shape and anisotropic motion characteristics complicate the achievement of high-precision motion planning and autonomous navigation. In this paper, we propose a safe and robust motion planning system tailored for autonomous navigation of quadruped robots in cluttered environments. We adopt a hierarchical architecture and decompose the planning process into front-end searching and back-end optimization. In the front-end searching stage, the robot finds a smooth, feasible, and energy-efficient initial trajectory with safety consideration. In the back-end optimization stage, we leverage B-splines to enhance the trajectory smoothness, safety, and motion stability. Finally, the time allocation is fine-tuned through iterative refinement, ensuring the feasibility of the optimized trajectory. Our method is extensively validated in challenging simulations as well as in real-world testing environments, benchmark comparisons also demonstrate the improved performance of our method.

INDEX TERMS Quadruped robots, autonomous navigation, motion and path planning, B-spline trajectory optimization.

I. INTRODUCTION

In recent years, owing to their superior flexibility, terrain adaptability, and payload capacity in unstructured environments, quadruped robots have gradually become a research focus in the field of robotics. They have achieved commendable outcomes in various applications such as terrain exploration, post-disaster search and rescue, material transportation, and industrial inspection, significantly reducing human labor costs [1].

Compared to traditional wheeled and tracked ground robots, quadruped robots possess significant advantages. On the one hand, they can adjust their gaits in response to different ground structures, enabling stable locomotion similar to quadruped animals in unstructured environments such as rugged terrains. On the other hand, they have approximate omnidirectional mobility to navigate through narrow spaces such as tunnels and caves by adjusting their body postures, thus exhibiting higher flexibility. Therefore, it is meaningful and promising to integrate autonomous navigation technology with quadruped robots to enhance their autonomy and further expand their practical application range.

Existing work on quadruped robots primarily focuses on novel mechanical structures [2], [3] and the study of their gaits motion, including gait generation, foot trajectory planning, as well as body balance and control robustness [4], [5], [6]. These studies ensure stable and efficient low-level motion for quadruped robots across different environments and tasks but they lack the research on their high-level center-of-mass motion behaviors, limiting their autonomous capabilities and range of application in task-oriented missions within complex environments. Moreover, the non-negligible body shape of quadruped robots, along with their movement capability difference in various orientations, introduces additional complexity and challenges to the motion planning and autonomous navigation.

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.

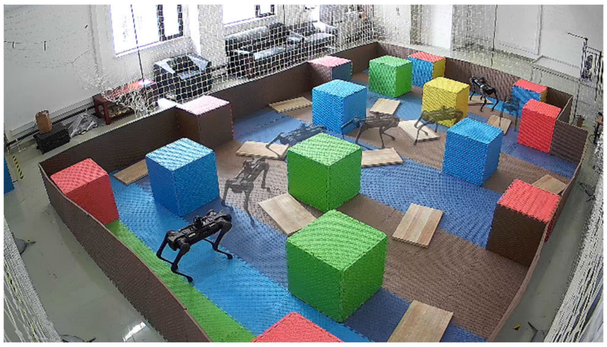


FIGURE 1. Composite image of an Aliengo robot navigating through the test field cluttered with boxes and boards.

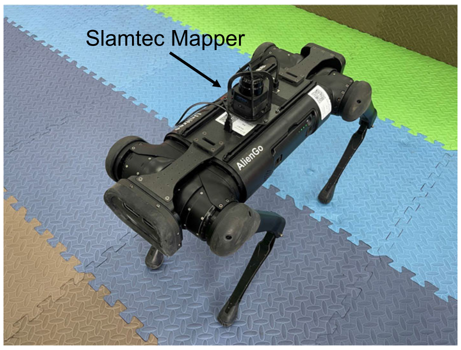


FIGURE 2. The Aliengo robot (weight: 21.5*kg*, standing dimensions: 0.8*m* × 0.35*m* × 0.6*m*). It is equipped with a Slamtec mapper LiDAR for environmental perception, an NVIDIA Jetson TX2 for motion planning, and an onboard miniPC for motion control.

In this paper, we propose a novel motion planning framework for the autonomous navigation of quadruped robots in cluttered environments. Figure 1 shows a composite image of an Aliengo robot navigating through a cluttered test field with our proposed method. Our framework employs a hierarchical planning structure dividing the planning process into front-end and back-end stages. In the front-end stage, we propose a kinodynamic path searching method, named *Improved Kinodynamic A**, that integrates the robot's kinematics into the search process. This method balances the control energy consumption and the path smoothness to quickly search for a guiding global trajectory connecting the initial and terminal states. In the back-end stage, the guiding trajectory is parameterized into a B-spline trajectory, and we optimize the control points of the trajectory by leveraging the local controllability and convex hull properties of B-splines, thus enhancing its smoothness and improving clearance with obstacles. Finally, to ensure the feasibility of the generated trajectory, iterative refinement is employed, with the motion characteristics of quadruped robots taking into account, thereby eliminating infeasible velocities and accelerations within the trajectory.

We compare our method with an existing state-of-the-art work employing a similar planning method in both simulated and real-world environments. Benchmark comparisons demonstrate that our method has lower energy consumption and higher stability, with higher success rates in the autonomous navigation tasks. The main contributions of this paper are summarized as follows:

1) We propose a novel path searching method *Improved Kinodynamic A**, which generates a guiding initial trajectory conforming to the motion characteristics of quadruped robots.
2) Incorporating the body shape and motion characteristics of quadruped robots, we propose a trajectory optimization method based on B-splines to generate smooth, safe, and feasible trajectories.
3) Based on the optimized trajectory and the motion characteristics of quadruped robots, we propose an iterative refinement method that ensures the feasibility of the final trajectory.

The remainder of this article is organized as follows. Section II summarizes the related works. Section III introduces the quadruped robot hardware platform and the system architecture. Section IV details the *Improved Kinodynamic A** path searching algorithm, while Section V presents the gradient-based trajectory optimization algorithm. Section VI validates and analyzes the proposed method through both simulation and real-world experiments. Finally, Section VII concludes the article.

II. RELATED WORKS

As stated before, only a limited number of works have explored autonomous navigation of quadruped robots, including navigation in unstructured terrains or unknown environments [7], [8], [9], [10], [11], [12], [13], [14], the navigation with multi-gait combinations (e.g. walking and jumping) [15], [16], [17], and multi-robot cooperative exploration [18], [19]. For the motion planning and autonomous navigation of robotic systems, the prevalent approach is to utilize a two-stage hierarchical architecture, which can be decomposed into front-end searching and back-end optimization.

A. FRONT-END SEARCHING

Front-end searching, also known as global planning in some works, aims to search for a guiding global path in a known or partially known environment. The majority of existing works utilize search-based methods such as A* [7], [8], [9], [10], [15], D* Lite [13], or sampling-based methods such as RRT* [14] and RRT-Connect [16], [17] for global searching. However, while these approaches can find a guiding path geometrically, they lack the robot's kinematics, which may lead to infeasible trajectories for the actual robot, or increased energy consumption in the final trajectory [20]. To address this shortcoming, an intuitive solution is employing the kinodynamic searching. For instance, Hybrid-state A* [21] generates dynamically feasible trajectories for autonomous vehicles in the searching stage. Inspired by this, Zhou et al. [22] designed a kinodynamic path searching method for quadrotors. Considering the approximate omni-directional mobility of quadruped robots within their motion plane, we propose the *Improved Kinodynamic A** approach,

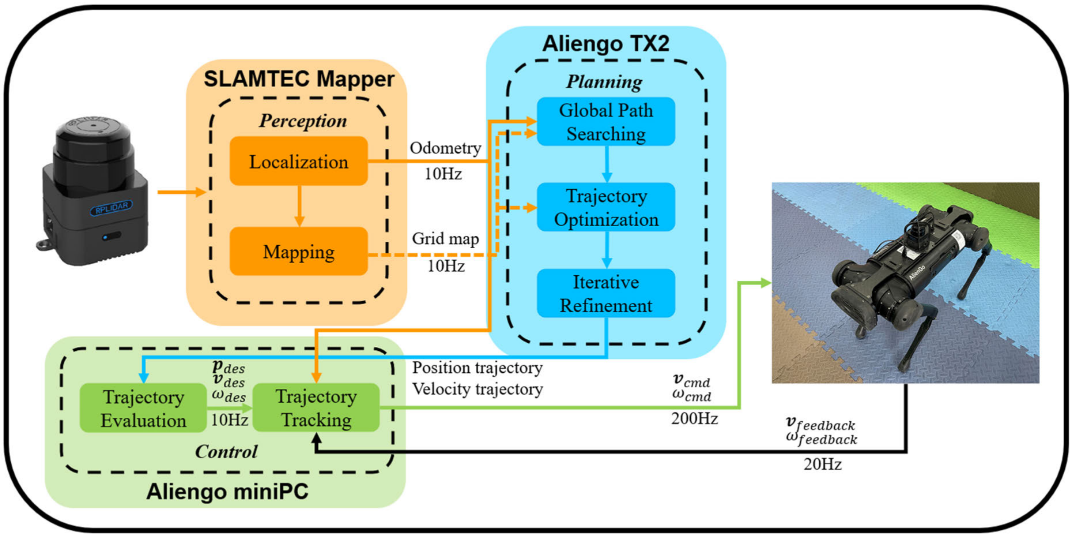


FIGURE 3. System architecture. The orange part represents the perception module, which computes within the LiDAR's internal processor. The blue part denotes the planning module, deployed on the TX2. The green part signifies the control module, running on the onboard miniPC.

which generates a smooth and feasible initial trajectory in the front-end searching stage.

B. BACK-END OPTIMIZATION

Back-end optimization aims at optimizing the global path into an executable trajectory. Some works [8], [15], [17] refine the gait of quadruped robots, directly controlling leg movements to track the global path. Others focus on iteratively optimizing the local trajectory based on the robot's current position [11], [12], [14], [15]. For instance, Wang et al. [12] utilizes the Timed Elastic Band (TEB) for local trajectory optimization. Feng et al. [11] generates multiple local trajectories expressed by Bézier curves, and selects the optimal local trajectory using a scoring strategy. The method proposed in [10] formulates the generation of the executable trajectory into a numerical optimization problem which aims at minimizing the integral of the squared acceleration over time. Wooden et al. [7] fits the global path into a time-parametrized polynomial curve to obtain a smooth and continuous trajectory directly. Such methods are more beneficial to optimizing the trajectory holistically and have demonstrated excellent performance in the autonomous navigation of other types of mobile robots [22], [23], [24]. We parameterize the initial global trajectory into a B-spline and leverage the convex hull as well as the local control properties of B-splines to enhance the smoothness, safety, and motion stability of the trajectory. Our method not only ensures the generated trajectory is smooth and feasible, but also offers the advantage of low computation cost.

III. SYSTEM OVERVIEW

A. EXPERIMENT PLATFORM

We use the Unitree Aliengo[1] quadruped robot as our experiment platform (Figure 2). It is equipped with a Slamtec

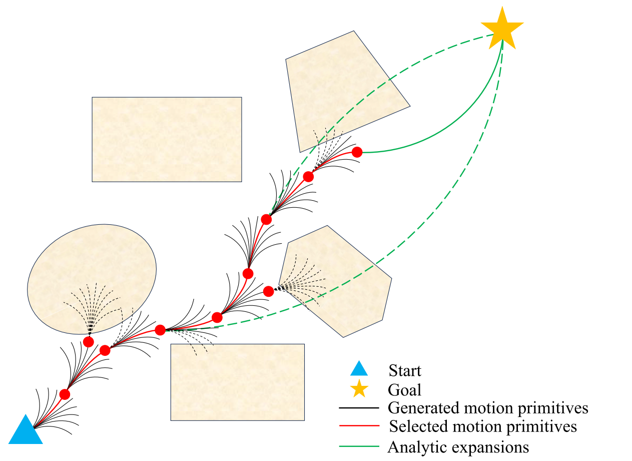


FIGURE 4. Illustration of the *Improved Kinodynamic A** search process. The algorithm expands nodes using motion primitives (red curves) and employs periodic analytic expansions (green curves) to enhance the search efficiency. The dashed trajectories are abandoned due to collision with obstacles.

Mapper LiDAR[2] for environmental perception, which integrates a high-performance SLAM algorithm outputting 2D grid maps up to 0.05m precision whilst performing high-accuracy robot localization. The Aliengo's on-board computing resources are provide by an NVIDIA Jetson TX2 and a miniPC equipped with an Intel Core i5-6300U processor (2.4 GHz, 2 cores, 4 threads) and 4GB RAM.

B. SYSTEM ARCHITECTURE

The overall architecture of our proposed system is illustrated in Figure 3. The perception module updates the map and robot localization at a frequency of 10Hz. The planning module begins by searching for a global initial trajectory using the *Improved Kinodynamic A** (see Section IV), then

[1][Online]. Available: https://www.unitree.com/aliengo/

[2][Online]. Available: https://www.slamtec.com/cn/Lidar/Mapper

parameterizes this trajectory into a B-spline and optimizes it based on the gradient of control points (see Section V-A). Finally, it iteratively refines the trajectory to eliminate the infeasibilities (see Section V-B). The control module evaluates and publishes the position and velocity commands in real-time based on the robot's current state and the ideal trajectories, facilitating autonomous navigation for the quadruped robot.

IV. IMPROVED KINODYNAMIC A* PATH SEARCHING

We extend the Fast-Planner [22] algorithm by introducing the smoothness cost to design our path searching method. We expand nodes by generating a set of motion primitives through discretized control inputs, design the cost function by combining the energy consumption and the path smoothness, and enhance search efficiency through analytic expansions. Figure 4 illustrates the search process of our method.

A. MOTION PRIMITIVES GENERATION

For the generation of motion primitives, we employ a second-order integrator model adapted for the quadruped robot. We decompose the robot's trajectory on the plane into two independent one-dimensional polynomial curves with respect to time under the world coordinate system, and consider the robot's position and velocity on the plane as the system states, with acceleration as the system input. The state-space model of the system can be defined as:

$$\dot{\boldsymbol{s}} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{B}\boldsymbol{u}, \tag{1}$$

where

$$\boldsymbol{A} = \begin{bmatrix} 0 & \boldsymbol{I}_2 \\ 0 & 0 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ \boldsymbol{I}_2 \end{bmatrix}.$$

Subsequently, with discretized accelerations$\{-u_{max}, -\frac{1}{2}u_{max}, 0, \frac{1}{2}u_{max}, u_{max}\}$ in each axis as the system input, a set of motion primitives can be generated through the following state transition equation:

$$\boldsymbol{s}(t) = e^{\boldsymbol{A}t}\boldsymbol{s}(0) + \int_0^t e^{\boldsymbol{A}(t-\tau)}\boldsymbol{B}\boldsymbol{u}(\tau)\, d\tau. \tag{2}$$

It is noteworthy that quadruped robots exhibit distinct mobility capabilities longitudinally and latitudinally. However, we do not differentiate such differences in the path searching. We will further optimize the trajectory with the quadruped robot kinematics in the back-end optimization, and the directional mobility differences will be inherently embedded in it.

B. EDGE COST

The edge cost consists of two parts. The first part represents the control energy consumption, calculated through the control input with a time trade-off. Specifically, we define the control energy consumption cost of one motion primitive with a duration of τ and control input $\boldsymbol{u}(t)$ as:

$$e_{egy}(\tau) = \int_0^\tau \|\boldsymbol{u}(t)\|^2\, dt + \rho\tau, \tag{3}$$

where $\rho\tau$ represents the time weight of the motion primitive, with ρ being the time trade-off coefficient.

The second part represents the smoothness of the trajectory. We denote $\boldsymbol{q}_{cur}$ as the current node, $\boldsymbol{q}_{par}$ as its parent node, and $\boldsymbol{q}_{pro}$ as the node being expanded. Take the x-axis as an example, we introduce $\Delta x = \boldsymbol{q}_{cur}(x) - \boldsymbol{q}_{pro}(x)$ and $\Delta x' = \boldsymbol{q}_{par}(x) - \boldsymbol{q}_{cur}(x)$ for clarity. Δy and $\Delta y'$ are defined in a similar manner for the y-axis.

The smoothness cost of one motion primitive is defined as:

$$e_{smt} = \left| tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) - tan^{-1}\left(\frac{\Delta y'}{\Delta x'}\right) \right|. \tag{4}$$

The total cost of a motion primitive is then expressed as the weighted sum of the energy consumption and the smoothness cost:

$$e_{total} = \lambda_e e_{egy}(\tau) + \lambda_s e_{smt}, \tag{5}$$

where λ_e, λ_s are the respective weights. For a trajectory with N motion primitives, the cumulative edge cost is:

$$g_c = \sum_{n=1}^{N} e_{total_n}. \tag{6}$$

C. HEURISTIC COST AND ANALYTIC EXPANSION

For the heuristic cost, we follow the design in Fast-Planner, utilizing Pontryagin's Minimum Principle [25] to compute the energy-optimized closed-form trajectory from the current state to the goal state:

$$\begin{aligned} \begin{bmatrix} \alpha_n \\ \beta_n \end{bmatrix} &= \frac{1}{T^3}\begin{bmatrix} -12 & 6\mathrm{T} \\ 6T & -2T^2 \end{bmatrix}\begin{bmatrix} p_{ng} - p_{nc} - v_{nc}T \\ v_{ng} - v_{nc} \end{bmatrix}, \\ \mathcal{H}^*(T) &= \min_T\left(\int_0^T \|\boldsymbol{u}(t)\|^2\, dt + \rho T\right) \\ &= \min_T\left(\sum_{n\in\{x,y\}} \left(\frac{1}{3}\alpha_n^2 T^3 + \alpha_n\beta_n T^2 + \beta_n^2 T\right) + \rho T\right), \end{aligned} \tag{7}$$

where p_{nc}, v_{nc} are the position and velocity of the current node, p_{ng}, v_{ng} are the position and velocity of the goal.

Similar to Fast-Planner, we utilize periodic analytic expansions to enhance the search efficiency (illustrated by the green curves in Figure 4). If the trajectory is collision-free (solid green curve), we consider this trajectory as the remaining path segment, thus allowing the search to terminate early.

V. GRADIENT-BASED TRAJECTORY OPTIMIZATION

A. PROBLEM FORMULATION

We first sample the initial trajectory at a given sampling frequency and fit the sampled points into a uniform B-spline. A p-degree B-spline is defined by $N+1$ control points $\{\boldsymbol{Q}_0, \boldsymbol{Q}_1, \boldsymbol{Q}_2, \ldots, \boldsymbol{Q}_N\}$ and a monotonically non-decreasing knot vector $[t_0, t_1, t_2, \ldots, t_M]$, where $\boldsymbol{Q}_i \in \mathbb{R}^2$, $t_m \in \mathbb{R}$ and $M = N + p + 1$. A uniform B-spline has identical knot span $\Delta t_i = t_{i+1} - t_i$, and the control points of velocity and acceleration in the world frame is calculated as:

$$\boldsymbol{V}_{w,i} = \frac{1}{\Delta t}\left(\boldsymbol{Q}_{i+1} - \boldsymbol{Q}_i\right), \boldsymbol{A}_{w,i} = \frac{1}{\Delta t}\left(\boldsymbol{V}_{w,i+1} - \boldsymbol{V}_{w,i}\right). \tag{8}$$

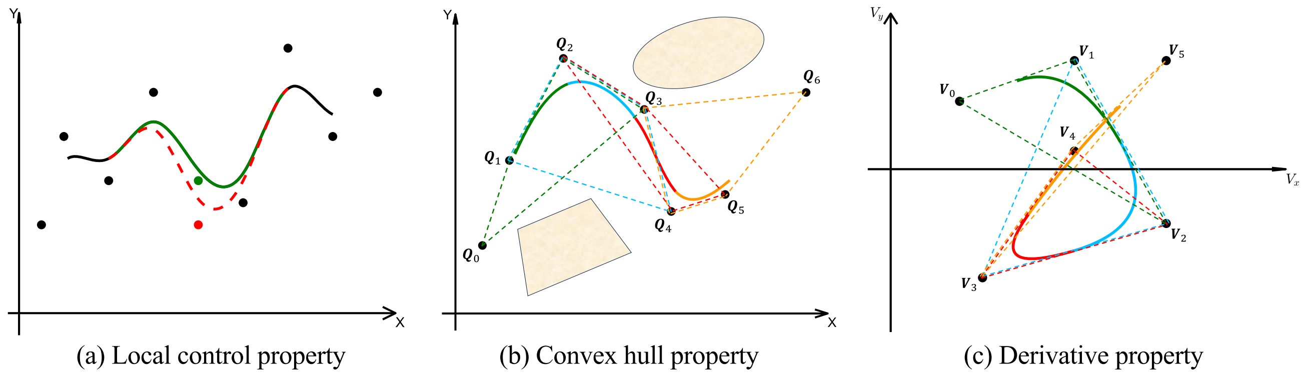


FIGURE 5. Illustration of the B-spline properties, exemplified by third-order B-splines. (a) demonstrates the local control property, where different colors indicate the relationship between control points and their corresponding local curves. (b) illustrates the convex hull property, with different colors showing the relationship between the convex hull and the corresponding segmented curves. By ensuring the safety of the convex hull, the safety of the entire trajectory is ensured. (c) shows the velocity curve derived from the B-spline trajectory in (b). The feasibility is ensured by the convex hull formed by the velocity control points.

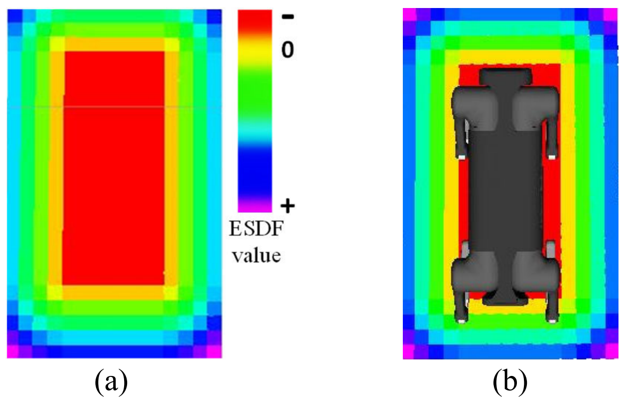


FIGURE 6. (a) visualizes the constructed RC-ESDF, with different colors corresponding to different ESDF values. (b) explains the relationship between RC-ESDF and the robot's body size, with the dimensions slightly expanded to 1.2m × 0.75m beyond the robot's body.

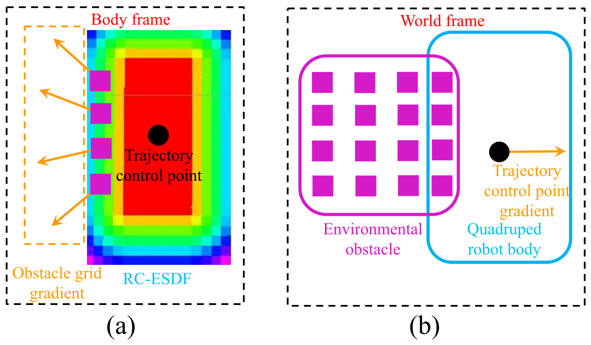


FIGURE 7. The purple squares in (a) denote the obstacle grids, and the orange arrows indicate the gradients of each obstacle grid in the body frame. (b) illustrates the gradient of the trajectory control point in the world frame, derived from the obstacle grid gradients shown in (a).

We use B-splines in the trajectory optimization because they have some unique properties. Firstly, modifying a few control points only affects the local region of the curve, which can be helpful for localized optimization. Secondly, the B-spline curve lies within the convex hull formed by its control points, ensuring the safety of the trajectory. Finally, the derivatives of a B-spline are also B-splines and adhere to the convex hull property, which ensures the feasibility of controller as long as the control points of the velocity and acceleration trajectories are within the actuator range. Figure 5 summarizes these properties.

We construct the optimization problem in a soft constrained form as:

$$\min_{Q} J = \lambda_s J_s + \lambda_c J_c + \lambda_f J_f + \lambda_t J_t, \tag{9}$$

where J_s, J_c, J_f, and J_t represent the costs of smoothness, collision, feasibility, and stability, respectively. λ_s, λ_c, λ_f, and λ_t are the corresponding weights. For a p-degree B-spline, we purposefully do not optimize the initial and final p control points, as they determine the trajectory's initial and terminal states and thus should remain unchanged.

1) SMOOTHNESS COST

We minimize the square of trajectory acceleration and jerk to optimize the smoothness of the trajectory. The smoothness cost is formulated as:

$$J_s = \sum_{i=0}^{N-2} \|\boldsymbol{A}_i\|^2 + \sum_{i=0}^{N-3} \|\boldsymbol{J}_i\|^2 , \tag{10}$$

where

$$\begin{aligned}\boldsymbol{A}_i &= (\boldsymbol{Q}_{i+2} - \boldsymbol{Q}_{i+1}) - (\boldsymbol{Q}_{i+1} - \boldsymbol{Q}_i)\\ &= \boldsymbol{Q}_{i+2} - 2\boldsymbol{Q}_{i+1} + \boldsymbol{Q}_i\\ \boldsymbol{J}_i &= (\boldsymbol{Q}_{i+3} - 2\boldsymbol{Q}_{i+2} + \boldsymbol{Q}_{i+1}) - (\boldsymbol{Q}_{i+2} - 2\boldsymbol{Q}_{i+1} + \boldsymbol{Q}_i)\\ &= \boldsymbol{Q}_{i+3} - 3\boldsymbol{Q}_{i+2} + 3\boldsymbol{Q}_{i+1} - \boldsymbol{Q}_i\end{aligned}$$

2) COLLISION COST

[22] adopts a traditional ESDF-based method for safety cost. However, this approach demands continuous updates of a local ESDF, consuming considerable computational resources and memory. Moreover, it treats the robot as a point mass, potentially leading to conservative or unsafe planning results for quadruped robots.

Therefore, we design our collision cost based on Robo-centric ESDF (RC-ESDF) [26], which requires only a one-time, offline construction of ESDF for the robot's body shape. This approach not only eliminates the resource consumption of maintaining a local environmental ESDF map but also takes the robot's shape into account. The RC-ESDF of the quadruped robot is shown in Figure 6, and Figure 7 illustrates its obstacle avoidance process. For a trajectory control point, we first calculate the gradient for each obstacle grid in the body frame within RC-ESDF through bilinear interpolation of the ESDF values. These gradients are then summed and converted into the world frame to obtain the gradient of the control point, enhancing trajectory safety. For a detailed introduction to RC-ESDF, readers may refer to [26], and the RC-ESDF values can be efficiently computed by [27].

Using the convex hull property of B-splines, we formulate the safety cost function as:

$$J_c = \sum_{i=0}^{N-1} F_c\left(\boldsymbol{Q}_i\right), \tag{11}$$

where $F_c\left(\boldsymbol{Q}_i\right)$ is the cumulative cost calculated from M obstacle grids in the RC- ESDF at control point $\boldsymbol{Q}_i$.

$$F_c\left(\boldsymbol{Q}_i\right) = \sum_{k=1}^{M} F_k, \tag{12}$$

$$F_k = \begin{cases} \left(d_k - d_{th}\right)^2, & d_k < d_{th} \\ 0, & d_k \geq d_{th} \end{cases}, \tag{13}$$

where d_k is the ESDF value for the k-th obstacle grid, and d_{th} is the predetermined distance threshold.

To calculate d_k in (13), we transform the obstacle grids around control point $\boldsymbol{Q}_i$ into the body frame at $\boldsymbol{Q}_i$. For the k-th obstacle grid whose world coordinates are $\boldsymbol{q}_w^k$, its coordinates $\boldsymbol{q}_b^k$ in the body frame at control point $\boldsymbol{Q}_i$ can be calculated by:

$$\boldsymbol{q}_b^k = \boldsymbol{R}_{wb,i}^{-1}\left(\boldsymbol{q}_w^k - \boldsymbol{Q}_i\right), \tag{14}$$

where $\boldsymbol{R}_{wb,i}$ is the rotation matrix from body frame to world frame at control point $\boldsymbol{Q}_i$:

$$\boldsymbol{R}_{wb,i} = \begin{bmatrix} cos\theta_i & -sin\theta_i \\ sin\theta_i & cos\theta_i \end{bmatrix},$$

where θ_i is the orientation of the robot at control point $\boldsymbol{Q}_i$. Considering the optimal longitudinal motion capability of the quadruped robot, we set the robot's orientation to be aligned with the trajectory's tangent direction. The orientation of the robot at control point $\boldsymbol{Q}_i$ can be calculated by:

$$\theta_i = \tan^{-1}\left(\frac{\boldsymbol{Q}_{i+1}(y) - \boldsymbol{Q}_i(y)}{\boldsymbol{Q}_{i+1}(x) - \boldsymbol{Q}_i(x)}\right). \tag{15}$$

The gradient of the cost function $F_c\left(\boldsymbol{Q}_i\right)$ with respect to the control point $\boldsymbol{Q}_i$ is obtained by:

$$\frac{\partial F_c\left(\boldsymbol{Q}_i\right)}{\partial \boldsymbol{Q}_i} = -2\boldsymbol{R}_{wb,i}\sum_{k=1}^{M}\left(d_k - d_{th}\right)\frac{\partial d_k}{\partial \boldsymbol{q}_b^k}, \tag{16}$$

where $\partial d_k / \partial \boldsymbol{q}_b^k$ can be efficiently computed through bilinear interpolation of the RC-ESDF values.

3) FEASIBILITY COST

The feasibility cost primarily aims to constrain the trajectory's velocity and acceleration to remain within the actuator range. The feasibility cost is defined as the penalization of velocities and accelerations that exceed the actuator limits.

Due to the distinct longitudinal and latitudinal movement capabilities of quadruped robots, we design our feasibility cost as follows:

$$J_f = \sum_{i=0}^{N-1} F_d\left(V_{b,i}\right) + \sum_{i=0}^{N-2} F_d\left(A_{b,i}\right), \tag{17}$$

where

$$\begin{aligned} &F_d\left(\boldsymbol{V}_{b,i}\right) \\ &= f_x\left(\boldsymbol{V}_{b,i}^{(x)}\right) + f_y\left(\boldsymbol{V}_{b,i}^{(y)}\right), \\ &f_x\left(\boldsymbol{V}_{b,i}^{(x)}\right) \\ &= \begin{cases} e^{\boldsymbol{V}_{b,i}^{(x)} - V_{max,f}} - 1, & \boldsymbol{V}_{b,i}^{(x)} \geq V_{max,f} \\ 0, & V_{max,b} \leq \boldsymbol{V}_{b,i}^{(x)} < V_{max,f} \\ e^{V_{max,b} - \boldsymbol{V}_{b,i}^{(x)}} - 1, & \boldsymbol{V}_{b,i}^{(x)} < V_{max,b} \end{cases}, \\ &f_y\left(\boldsymbol{V}_{b,i}^{(y)}\right) \\ &= \begin{cases} e^{\boldsymbol{V}_{b,i}^{(y)} - V_{max,l}} - 1, & \boldsymbol{V}_{b,i}^{(y)} \geq V_{max,l} \\ 0, & -V_{max,l} \leq \boldsymbol{V}_{b,i}^{(y)} < V_{max,l} \\ e^{-V_{max,l} - \boldsymbol{V}_{b,i}^{(y)}} - 1, & \boldsymbol{V}_{b,i}^{(y)} < -V_{max,l} \end{cases}. \end{aligned} \tag{18}$$

The velocity cost $F_d\left(\boldsymbol{V}_{b,i}\right)$ is designed as (18), where $\boldsymbol{V}_{b,i}$ is the trajectory's velocity control point in the body frame. $V_{max,f}, V_{max,b}, V_{max,l}$are the maximum allowable velocities of the robot in the forward, backward, and lateral directions, respectively. The acceleration cost $F_d\left(\boldsymbol{A}_{b,i}\right)$ has the identical form. The control points $\boldsymbol{V}_{b,i}$ and $\boldsymbol{A}_{b,i}$ for the velocity and acceleration in the body frame are calculated with $\boldsymbol{R}_{wb,i}$:

$$\boldsymbol{V}_{b,i} = \boldsymbol{R}_{wb,i}^{-1}\boldsymbol{V}_{w,i},\ \boldsymbol{A}_{b,i} = \boldsymbol{R}_{wb,i}^{-1}\boldsymbol{A}_{w,i}. \tag{19}$$

4) STABILITY COST

Since quadruped robots often work on uneven terrain, excessive velocity changes may introduce significant instability. To mitigate this risk, we design the stability cost by imposing penalties on the excessive changes in the robot's velocity. This cost aims to ensure smoother transitions in the robot's speed:

$$J_t = \sum_{\mu\in\{x,y\}}\sum_{i=0}^{N-2} F_t\left(\Delta V_{\mu,i}\right), \tag{20}$$

where

$$\begin{aligned} &F_t\left(\Delta V_{\mu,i}\right) \\ &= \begin{cases} \left(\Delta V_{\mu,i}^2 - \Delta V_{\mu,max}^2\right)^2, & \Delta V_{\mu,i}^2 \geq \Delta V_{\mu,max}^2 \\ 0, & \Delta V_{\mu,i}^2 < \Delta V_{\mu,max}^2 \end{cases}, \end{aligned}$$

$\Delta \boldsymbol{V}_i = \boldsymbol{V}_{b,i+1} - \boldsymbol{V}_{b,i}$ is the difference between adjacent velocity control points in the body frame, $\Delta \boldsymbol{V}_{max}$ is the maximum allowable velocity change of the robot's body.

B. TRAJECTORY ITERATIVE REFINEMENT

To ensure the feasibility of the planned trajectory, we enlarge the infeasible knot spans and represent the trajectory as a non-uniform B-spline. For a p-degree non-uniform B-spline trajectory, the velocity and acceleration control points in the world frame are:

$$\begin{aligned}\boldsymbol{V}_{w,i} &= \frac{p\left(\boldsymbol{Q}_{i+1} - \boldsymbol{Q}_i\right)}{t_{i+p+1} - t_{i+1}},\\ \boldsymbol{A}_{w,i} &= \frac{(p-1)\left(\boldsymbol{V}_{w,i+1} - \boldsymbol{V}_{w,i}\right)}{t_{i+p+1} - t_{i+2}}.\end{aligned} \tag{21}$$

The corresponding control points in the body frame are obtained by (19). Denote $\boldsymbol{V}_{b,i}$ the infeasible velocity and $\boldsymbol{A}_{b,i}$ the acceleration in the body frame, the enlarged knot spans are calculated as follows:

$$\begin{aligned}&\Delta t_i' = min\left\{\gamma, max\left\{F_{re,v}\left(\boldsymbol{V}_{b,i}\right), F_{re,a}\left(\boldsymbol{A}_{b,i}\right)\right\}\right\} \cdot \Delta t_i,\\ &F_{re,v}\left(\boldsymbol{V}_{b,i}\right)\\ &= \begin{cases} max\left\{\dfrac{\boldsymbol{V}^{(x)}_{b,i}}{V_{max,f}}, \dfrac{\boldsymbol{V}^{(y)}_{b,i}}{V_{max,l}}\right\}, & \boldsymbol{V}^{(x)}_{b,i} \geq 0\\ max\left\{\dfrac{\boldsymbol{V}^{(x)}_{b,i}}{V_{max,b}}, \dfrac{\boldsymbol{V}^{(y)}_{b,i}}{V_{max,l}}\right\}, & \boldsymbol{V}^{(x)}_{b,i} < 0\end{cases},\\ &F_{re,a}\left(\boldsymbol{A}_{b,i}\right)\\ &= \begin{cases} max\left\{\left(\dfrac{\boldsymbol{A}^{(x)}_{b,i}}{A_{max,f}}\right)^{\frac{1}{2}}, \left(\dfrac{\boldsymbol{A}^{(y)}_{b,i}}{A_{max,l}}\right)^{\frac{1}{2}}\right\}, & \boldsymbol{A}^{(x)}_{b,i} \geq 0\\ max\left\{\left(\dfrac{\boldsymbol{A}^{(x)}_{b,i}}{A_{max,b}}\right)^{\frac{1}{2}}, \left(\dfrac{\boldsymbol{A}^{(y)}_{b,i}}{A_{max,l}}\right)^{\frac{1}{2}}\right\}, & \boldsymbol{A}^{(x)}_{b,i} < 0\end{cases},\end{aligned} \tag{22}$$

where γ is a constant slightly larger than 1 to prevent over-enlarging the knot spans. $V_{max,f}$, $V_{max,b}$, $V_{max,l}$, $A_{max,f}$, $A_{max,b}$ and $A_{max,l}$ are the robot's maximum allowable forward, backward, and lateral velocities and accelerations, respectively. As quadruped robots have different maximum allowable speeds and accelerations in the forward and backward directions, the enlarged knot spans need to be calculated based on the robot's direction of movement.

VI. EXPERIMENT RESULTS

A. IMPLEMENTATION DETAILS

In this section, utilizing the system architecture described in Section III-B (see Figure 3), a finite state machine is adopted to coordinate the various modules of the system. In each planning cycle, it executes the path searching described in Section IV and the trajectory optimization described in Section V. The optimized trajectory is then evaluated by the control module for execution by the quadruped robot.

TABLE 1. Kinematic constraints of quadruped robot.

Path searching	v_{max}	1.0	m/s
	a_{max}	0.6	
Trajectory optimization	$v_{max,forward}$	1.5	m/s
	$v_{max,backward}$	-0.8	
	$v_{max,lateral}$	0.4	
	$a_{max,forward}$	0.7	m/s^2
	$a_{max,backward}$	-0.4	
	$a_{max,lateral}$	0.25	

TABLE 2. Simulation trajectory planning comparison.

Res.(m)	Alg.	t_{plan}(ms)		Length(m)		Energy(m^2/s^3)	
		Mean	Max	Mean	Max	Mean	Max
0.05	Proposed	127.57	223.04	**40.52**	**43.11**	**33.48**	**67.22**
	Fast-Planner	**15.47**	**28.33**	40.88	43.71	40.33	73.35
0.075	Proposed	68.38	138.41	**40.88**	**42.57**	**38.28**	**68.07**
	Fast-Planner	**14.73**	**23.40**	41.33	44.90	42.95	69.01
0.1	Proposed	42.36	84.48	**40.81**	**43.62**	**34.83**	**54.48**
	Fast-Planner	**13.62**	**20.38**	40.96	43.87	39.69	62.47

We use the nonlinear optimization solver NLopt[3] to solve the constructed unconstrained optimization problem in section V-A. The Aliengo robot runs in trot gait and its kinematic constraints are shown in Table 1. For the perception and state estimation, we utilize the map and localization provided by the LiDAR's internal high-precision SLAM algorithm. To mitigate the impact of environmental perceptual errors on autonomous navigation, we pre-constructed the map with the LiDAR mounted on the robot, which is then used to correct only minor errors during navigation.

In the path searching stage, considering the quadruped robot model is not precisely integrated, we select moderate values within the kinematic limits. For the trajectory optimization, we set the B-spline order to $p = 3$ and set the weights λ_s, λ_c to be much larger than λ_f, λ_t. This is because we prefer to first generate a trajectory that is geometrically safe and smooth, and then eliminate the infeasibilities through iterative refinement later.

We compare our method with Fast-Planner [22] in both simulated and real-world environments. Because Fast-Planner is unable to handle the distinct motion capabilities of quadruped robots in longitudinal and latitudinal directions, we adopt the constraints of path searching in Table 1 for it and use its default parameters for the rest, and

[3][Online]. Available:https://nlopt.readthedocs.io/en/latest/

TABLE 3. **Real-world autonomous navigation comparison.**

Alg.	t_{comp}(s)	t_{plan}(ms)	t_{ESDF}(ms)	Length(m)	Energy (m^2/s^3)	Succ. Rate
Proposed	12.06	469.15	**0.58**	**7.33**	**11.25**	**100%**
Fast-Planner	**11.38**	**47.23**	797.92	7.82	23.14	60%

TABLE 4. **Autonomous navigation stability comparison.**

Alg.	Vel.(m/s)		Acc.(m/s^2)	
	Mean	Std	Mean	Std
Proposed	0.563	**0.197**	0.284	**0.122**
Fast-Planner	**0.628**	0.261	**0.408**	0.213

TABLE 5. **Real-world autonomous cruising comparison.**

Group	Alg.	t_{comp}(s)	t_{plan}(ms)	t_{ESDF}(ms)	Length(m)	Energy (m^2/s^3)	Succ. Rate
Exp. (a)	Proposed	**12.06**	431.60	**0.62**	**17.87**	**44.56**	**100%**
	Fast-Planner	26.20	**194.26**	1704.56	18.43	69.69	75%
Exp. (b)	Proposed	**21.67**	450.15	**0.64**	**17.60**	**38.77**	**100%**
	Fast-Planner	31.71	**247.72**	2189.37	18.92	62.14	75%

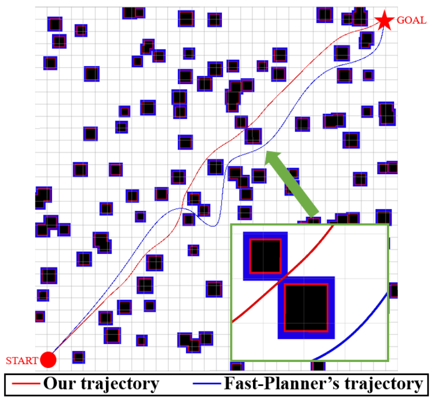


FIGURE 8. **Visualization of one simulation result. The black areas represent environmental obstacles. The red and blue areas illustrate the obstacle expansion regions for our method and Fast-Planner, respectively.**

we modify Fast-Planner's planning space to better align with the movement plane of the quadruped robot.

In the computation of the collision cost, the extensive interpolation calculations lead to the long plan time. To increase the optimization efficiency, we pre-calculate ESDF values and corresponding gradients for RC-ESDF at $0.01m$ resolution, which enables the quick retrieval of the ESDF values and gradients using a look-up table method. In addition, we adopt a receding-horizon re-planning strategy to enhance system robustness.

B. SIMULATION BENCHMARK COMPARISONS

In the simulations, we focus on the planning performance without the execution of the robot. The simulated environment is a $30m \times 30m$ field with 120 randomly deployed obstacles. Fixed start and goal positions are set approximately 40 meters apart. We perform simulations on three different map resolutions ($0.05m$, $0.75m$, $0.1m$). For each resolution, both algorithms are run 50 times, with each test featuring a random obstacle distribution. The simulation results are listed in Table 2, and Figure 8 shows one simulation result.

Simulation results indicate that our method plans trajectories with shorter distances and lower energy consumption (Integral of squared acceleration) across different resolutions. This can be attributed to our non-point mass model in the computation of the collision cost, which requires only slight expansion of the obstacles to compensate for the LiDAR scanning gaps, yielding larger free spaces and shorter trajectories through narrower spaces, as shown in Figure 8. Furthermore, our method integrates the movement characteristics of the quadruped robot into the optimization and mitigates aggressive velocity changes in the trajectory through stability optimization, which helps improve the energy efficiency. The planning time of our method is longer, primarily due to the separate cost calculations required for each obstacle grid within the RC-ESDF at each control point for collision cost. However, our approach efficiently avoids the resource consumption to maintain a local environmental ESDF map (see Section VI-C) and incorporates the robot's shape into the optimization. Moreover, due to our hierarchical deployment of planning and control modules, the real-time performance of the system is ensured to some extent, which mitigates the impact of the longer planning time on the system.

C. REAL-WORLD BENCHMARK COMPARISON

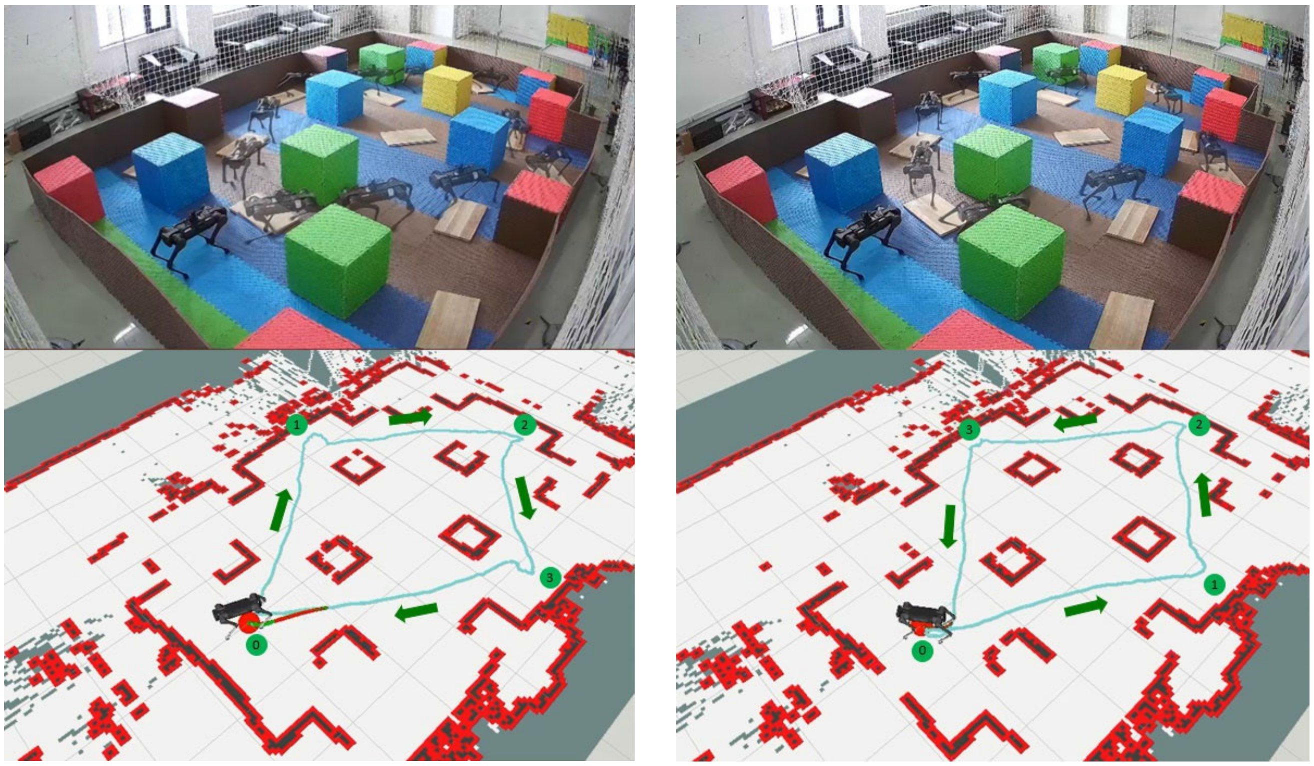
Our test field is a $7.8m \times 5.4m$ area, containing 15 obstacles (each measuring $0.6m \times 0.6m \times 0.6m$), and multiple wooden boards randomly placed on the ground to emulate the uneven terrain, introducing additional instability for the quadruped robot (Figure 1).

1) AUTONOMOUS NAVIGATION EXPERIMENT

We first conduct the autonomous navigation experiment with fixed start and goal positions about 6 meters apart, and repeat the experiment 10 times for each method. Figure 1 shows a composite image of one such trajectory using our method. The results of the 10 experiments are averaged, and the comparison results are shown in Table 3.

The experiment results are consistent with those from the simulation experiments. In particular, in the autonomous navigation experiments, the trajectories planned by our algorithm demonstrated advantages in terms of lower energy consumption and shorter trajectory length. Notably, our method benefits from a one-time RC-ESDF construction at system initialization. In contrast, Fast-Planner requires continuous updates of the local environmental ESDF during navigation (set to update within a $5m$ radius of the robot), resulting in a significantly longer ESDF construction time.

Additionally, our algorithm fully integrates the anisotropic motion characteristics of the quadruped robot in trajectory optimization, effectively avoiding overly aggressive



(a) (b)

FIGURE 9. Composite images and visualization of two sets of autonomous cruising experiments with our method. The green dots and arrows represent the waypoints and the trajectories directions, respectively. The blue curves show the trajectories of the quadruped robot.

movements. Although this results in a slightly longer completion time compared to Fast-Planner, it achieves a higher success rate. In contrast, the success rate of Fast-Planner is only 60%, with some aggressive movements during navigation, causing collisions with the obstacles.

To demonstrate the stability during navigation, we analyze the velocity and acceleration, with the comparative results shown in Table 4. The results indicate that although our method exhibits a lower average speed and acceleration, leading to a marginally longer completion time compared to Fast-Planner, it demonstrates smaller standard deviations in both velocity and acceleration, which indicates higher stability of our proposed method.

2) AUTONOMOUS CRUISING EXPERIMENT

To comprehensively verify the performance of our method, we also conduct two sets of autonomous cruising experiments involving multiple continuous navigation tasks. Within the test field, we preset multiple fixed waypoints for the quadruped robot to sequentially navigate through and finally return to the start. Table 5 shows the comparative results for these experiments, and Figure 9 visualizes the cruise trajectories.

As shown in Table 5, the results of the autonomous cruising experiments are consistent with those of the autonomous navigation experiment. In the autonomous cruising experiment, our proposed method takes less time. This is primarily due to our shorter obstacle expansion distance in narrow passages, which decreases the collision cost and leads to higher speeds, thereby reducing the completion time.

VII. CONCLUSION

In this paper, a safe and robust motion planning system is proposed for the autonomous navigation of quadruped robots in cluttered environments.

We decompose the planning process into front-end and back-end stages. In the front-end stage, we quickly search for a smooth and energy-efficient initial trajectory. In the back-end stage, we enhance the trajectory's smoothness, safety, and motion stability by utilizing the convex hull and local control properties of B-splines. Finally, we eliminate the infeasibilities through iterative refinement to ensure the feasibility of the trajectory. The system is validated in both simulated and real-world environments. Benchmark comparisons demonstrate that the trajectories generated by our method have low energy consumption and high stability, while achieving high success rates in autonomous navigation. The proposed method achieves 100% success rates in real-world experiments, compared to 60% and 75% for Fast-Planner, and reduces energy consumption by approximately 35%–50% on average. Additionally, the proposed method demonstrates more stable velocity and acceleration profiles, with lower mean and standard deviation values compared to Fast-Planner.

REFERENCES

[1] P. Biswal and P. K. Mohanty, "Development of quadruped walking robots: A review," *Ain Shams Eng. J.*, vol. 12, no. 2, pp. 2017–2031, Jun. 2021.

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

[3] B. Katz, J. Di Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2019, pp. 6295–6301.

[4] S. Katayama and T. Ohtsuka, "Whole-body model predictive control with rigid contacts via online switching time optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2022, pp. 8858–8865.

[5] S. Qi, W. Lin, Z. Hong, H. Chen, and W. Zhang, "Perceptive autonomous stair climbing for quadrupedal robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 2313–2320.

[6] M. Sombolestan, Y. Chen, and Q. Nguyen, "Adaptive force-based control for legged robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 7440–7447.

[7] D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A. A. Rizzi, and M. Raibert, "Autonomous navigation for BigDog," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 4736–4741.

[8] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 3664–3671.

[9] M. Brandão, O. B. Aladag, and I. Havoutis, "GaitMesh: Controller-aware navigation meshes for long-range legged locomotion planning in multi-layered environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 3596–3603, Apr. 2020.

[10] 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 Trans. Mechatronics*, vol. 26, no. 4, pp. 1997–2006, Aug. 2021.

[11] S. Feng, Z. Zhou, J. S. Smith, M. Asselmeier, Y. Zhao, and P. A. Vela, "GPF-BG: A hierarchical vision-based planning framework for safe quadrupedal navigation," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2023, pp. 1968–1975.

[12] P. Wang, X. Zhou, Q. Zhao, J. Wu, and Q. Zhu, "Search-based kinodynamic motion planning for omnidirectional quadruped robots," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Jul. 2021, pp. 823–829.

[13] A. Chilian and H. Hirschmüller, "Stereo camera based navigation of mobile robots on rough terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 4571–4576.

[14] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2016, pp. 1184–1189.

[15] S. Gilroy, D. Lau, L. Yang, E. Izaguirre, K. Biermayer, A. Xiao, M. Sun, A. Agrawal, J. Zeng, Z. Li, and K. Sreenath, "Autonomous navigation for quadrupedal robots with optimized jumping through constrained obstacles," in *Proc. IEEE 17th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2021, pp. 2132–2139.

[16] J. Norby and A. M. Johnson, "Fast global motion planning for dynamic legged robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 3829–3836.

[17] M. Chignoli, S. Morozov, and S. Kim, "Rapid and reliable quadruped motion planning with omnidirectional jumping," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 6621–6627.

[18] I. D. Miller, F. Cladera, A. Cowley, S. S. Shivakumar, E. S. Lee, L. Jarin-Lipschitz, A. Bhat, N. Rodrigues, A. Zhou, A. Cohen, A. Kulkarni, J. Laney, C. J. Taylor, and V. Kumar, "Mine tunnel exploration using multiple quadrupedal robots," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2840–2847, Apr. 2020.

[19] M. Kulkarni, M. Dharmadhikari, M. Tranzatto, S. Zimmermann, V. Reijgwart, P. De Petris, H. Nguyen, N. Khedekar, C. Papachristos, L. Ott, R. Siegwart, M. Hutter, and K. Alexis, "Autonomous teamed exploration of subterranean environments using legged and aerial robots," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2022, pp. 3306–3313.

[20] W. Ding, W. Gao, K. Wang, and S. Shen, "An efficient B-spline-based kinodynamic replanning framework for quadrotors," *IEEE Trans. Robot.*, vol. 35, no. 6, pp. 1287–1306, Dec. 2019.

[21] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *Int. J. Robot. Res.*, vol. 29, no. 5, pp. 485–501, Apr. 2010.

[22] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3529–3536, Oct. 2019.

[23] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao, "EGO-planner: An ESDF-free gradient-based local planner for quadrotors," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 478–485, Apr. 2021.

[24] R. Zhang, Y. Wu, L. Zhang, C. Xu, and F. Gao, "Autonomous and adaptive navigation for terrestrial-aerial bimodal vehicles," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3008–3015, Apr. 2022.

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

[26] S. Geng, Q. Wang, L. Xie, C. Xu, Y. Cao, and F. Gao, "Robo-centric ESDF: A fast and accurate whole-body collision evaluation tool for any-shape robotic planning," 2023, *arXiv:2306.16046*.

[27] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance transforms of sampled functions," *Theory Comput.*, vol. 8, no. 1, pp. 415–428, Sep. 2012.

HONGYI LIU received the B.S. degree in electronic science and technology from Soochow University, Suzhou, China. He is currently pursuing the M.S. degree in circuits and systems with Fudan University, Shanghai, China. His research interests include legged robots, autonomous navigation, and motion planning of mobile robots.

QUAN YUAN (Member, IEEE) received the B.S. degree in electronic information science and technology and the Ph.D. degree in circuits and systems from Fudan University, Shanghai, China, in 2021. His research interests include robotic systems, distributed control of multi-agent systems, and the coordination of unmanned autonomous vehicles.