Efficient Motion Planning Based on Kinodynamic Model for Quadruped Robots Following Persons in Confined Spaces

Zhen Zhang, Jiaqing Yan, Xin Kong, Guangyao Zhai, and Yong Liu, *Member, IEEE*

***Abstract*—Quadruped robots have superior terrain adaptability and flexible movement capabilities than traditional robots. In this article, we innovatively apply it in person-following tasks, and propose an efficient motion planning scheme for quadruped robots to generate a flexible and effective trajectory in confined spaces. The method builds a real-time local costmap via onboard sensors, which involves both static and dynamic obstacles. And we exploit a simplified kinodynamic model and formulate the friction pyramids formed by ground reaction forces' inequality constraints to ensure the executable of the optimized trajectory. In addition, we obtain the optimal following trajectory in the costmap completely based on the robot's rectangular footprint description, which ensures that it can walk through the narrow spaces avoiding collision. Finally, a receding horizon control strategy is employed to improve the robustness of motion in complex environments. The proposed motion planning framework is integrated on the quadruped robot JueYing and tested in simulation as well as real scenarios. It shows that the execution success rates in various scenes are all over 90%.**

***Index Terms*—Indoor navigation, Mobile robots, Motion planning, Robot kinematics.**

I. Introduction

FOLLOWING person arises when a human and a robot cooperate on a task [1]. It is widely used in agriculture, industry, military, and other diverse domains. There have been some research works, and most of them concentrate on wheeled or crawler robots [2]–[5]. Sometimes robots are required to assist humans in complicated environments, including narrow indoor and rugged unstructured outdoor scenes. Under these circumstances, space is always confined and narrow, where the limited mobility of the wheeled robot restricts its performance. Quadruped robots possess high mobility and dynamic motion capability compared with traditional robots. Furthermore, their remarkable adaptability to terrain enables them superior performance. Hence, it is promising and significant for quadruped robots to perform person-following tasks in confined terrains.

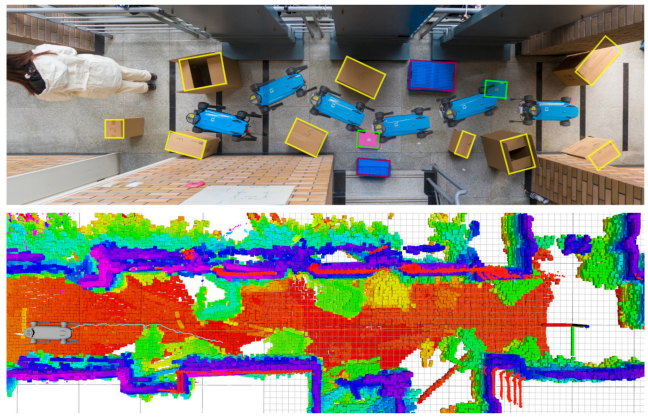

Fig. 1. Glimpse of field tests. The quadruped robot follows the walking person in the confined space.

Like an omnidirectional wheeled mobile robot, quadruped robots have approximate omnidirectional mobility and can translate and rotate simultaneously at a location. There are some research works on the motion planning of omnidirectional robots in confined space [6]–[8]. But when considering actuator limits, the quadruped robot's kinodynamics model is more complicated [9] and the motion planning is therefore more challenging. Some works [10] for quadruped body planning neglect this constraint, thus, the generated trajectories may exceed the limit of the actuators and cause performance degradation. Besides, due to the lack of a prior map, the robot needs to update the surrounding information online by onboard sensors. Both the nonstationary goal and the dynamic environment require real-time state feedback and trajectory generation. It brings greater challenges for person-following task.

We propose an efficient online motion planning approach for quadruped robots to perform person-following tasks in confined environments as shown in Fig. 1. The quadruped robot runs in trot gait to meet speed and flexibility requirements. In order to reduce the complexity, we apply simplifications for the map

Manuscript received December 31, 2020; revised March 26, 2021; accepted April 2, 2021. Date of publication May 25, 2021; date of current version August 13, 2021. Recommended by Technical Editor D. Wollherr and Senior Editor X. Chen. This work was supported in part by the National Natural Science Foundation of China under Grant 61836015 and in part by the National Key R&D Program of China under Grant 2017YFB1302003. *(Corresponding author: Yong Liu.)*

The authors are with the Advanced Perception on Robotics and Intelligent Learning Laboratory, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: zhenz@zju.edu.cn; qingjy@zju.edu.cn; xinkong@zju.edu.cn; zgyddzyx@zju.edu.cn; yongliu@iipc.zju.edu.cn).

This article has supplementary downloadable material available at http://10.1109/TMECH.2021.3083594, provided by the authors.

Digital Object Identifier 10.1109/TMECH.2021.3083594

and the model. Besides, we consider the dynamic constraints of the actuators and propose a simplified kinodynamic model. The model imposes constraints on the *ground reaction forces (GRFs)* to ensure the feasibility.

Furthermore, we decompose the motion planning problem into two steps: a search-based path finder and an optimization-based trajectory generation. The front-end planner takes the traversability cost into account and plans out a safe path. The back end uses this path as the initial condition, which is constrained by GRFs and solved by a nonlinear optimizer. For safety, we approximate the robot's footprint as a rectangle and regard the robot as a rigid body, which guarantees no collision between the trunk and environment. To improve robustness, we employ a receding horizon strategy, which means only the optimal control inputs within the prediction range are executed. We validate our method efficiency in both simulation and physical environments, including the narrow indoor corridor and rough outdoor terrain.

The main contributions of this article are as follows:

1) Propose a novel two-step, optimization-based, receding horizon method for quadruped robots to follow the person in confined environments.
2) Present an effective kinodynamics model for quadruped robots, whose constraints ensure the safety and feasibility.
3) Perform experiments both in simulation and the real environments to verify our method's efficiency and accuracy.

II. Related Work

With the development of robotics, the cooperation between robot and human gains more attention. Among them, the task of the robot following person is particularly critical due to its wide applications [1]. There have been some research works, and most of them focus on wheeled robots in agriculture and industry [2]–[5]. Sampling-based methods [11] and heuristic methods [12] are widely used and perform well in the relatively empty scenes. And there are two critical issues and limitations with wheeled robots. On the one hand, few researchers consider unstructured and complex environments, which pose greater challenges. On the other hand, wheeled robots have limited movement capabilities. Hence, they hardly handle confined spaces. Recently, omnidirectional robots gain more attention due to the holonomic characteristic. They can freely adjust the steering in all directions independently [6], [7]. However, its limited terrain adaptability determines its bounded application. Quadruped robots possess approximate omnidirectional mobility and adaptability to the terrain. Hence, it is necessary and promising for quadruped robots to solve the trajectory generation for the person-following problem.

Several key characteristics of quadruped robots make trajectory generation methods generally different from wheeled robots, including the complex model, specific map representations, and hierarchical planners. First of all, different from omnidirectional robots, quadruped robots possess a rather than complete omnidirectional mobility. For the omnidirectional robots, Sprunk *et al.* [6] present an optimization-based method in $\mathbb{R}^2$ and validate their method in the flat and spacious indoor environment [7]. Differently, the configuration of the quadruped robot is $SE(2) \equiv \mathbb{R}^2 \times \mathbb{S}^1$. Kennedy *et al.* [8] consider navigation in $SE(2)$ in confined space, which is similar to our problem. Nevertheless, when considering execution constraints, the quadruped robot model is more complicated [9]. Di Carlo *et al.* [13] introduce a simplified dynamics model and indicate that GRF is a crucial factor for quadruped robots in maintaining motion stability. Furthermore, the sensitivity of terrain determines the need for a more accurate representation of the environments and safer collision checking [14]. Chilian and Hirschmüller *et al.* [15] use the stereo camera and Wooden *et al.* [16] use LiDAR to estimate terrain's traversability. They both approximate the robot as a point and apply searching-based methods considering the traversability for planning. Different from them, we think it is critical to consider differences in mobility in different directions.

Moreover, due to the problem's complexity, some works [17]–[20] divide the planning for quadruped robots into a high-level Center of Mass (CoM) planner in the state space and a low-level footstep planner, which generates appropriate footholds. However, they rely on external systems for localization and traversability evaluation. Fankhauser *et al.* [21], [22] build a 2.5-D elevation map online, the cell of which stores a height value. Wermelinger *et al.* [10] and Winkler *et al.* [23] extend the aforementioned architectures by combining onboard perception to replan actions online. To reduce complexity, they employ a hierarchical structure that simplifies the footprint's representation. But this simplification may prevent the robot through locally narrow passages. Zhao *et al.* [24] use the two circles to simplify the hexagonal body, resulting in not safe or efficient enough paths. Based on the issues mentioned earlier, we accurately represent the footprint as a rectangle and consider its traversability based on the elevation map. We also incorporate the dynamics model to the planning problem to achieve motion stability. To the best of our knowledge, we first present a methodology for trajectory planning of quadruped robots in the following person.

III. System Overview

As shown in Fig. 2, our person-following scheme includes a perception module (see Section IV) and a motion planning module (see Section V). All the data required by the perception module are collected by sensors mounted on the robot body. The pose of the followed person is obtained by point cloud segmentation algorithm with the LiDAR data. Also, the LiDAR, depth camera, imu, and leg odometry data are fused to estimate the robot's pose and construct an elevation map of the surrounding environment.

The motion planning module receives the pose of the followed person as a local goal then employs a geometric path planner to find a coarse path. After initialization, an optimization procedure iteratively refines the CoM trajectory of the robot. Afterward, a receding horizon controller is used to produce control commands for quadruped robots. The commands are passed to the executor of quadruped robots to achieve the required movement. The whole system forms a closed loop of *perception–mapping–planning–controlling*, and updates at 10 Hz.

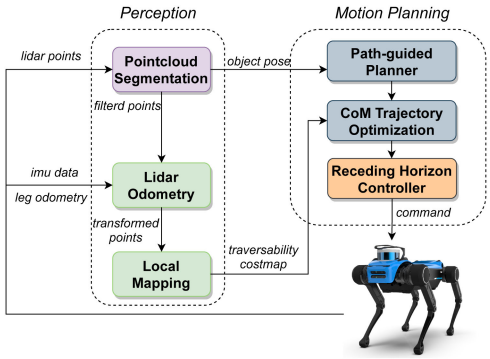

Fig. 2. Overview of our scheme for efficient navigation of quadruped robots. The dashed line mark the separation between perception and planning. The streamlines and arrows indicate the data flow. The feedback loop enhances the robustness of trajectory generation.

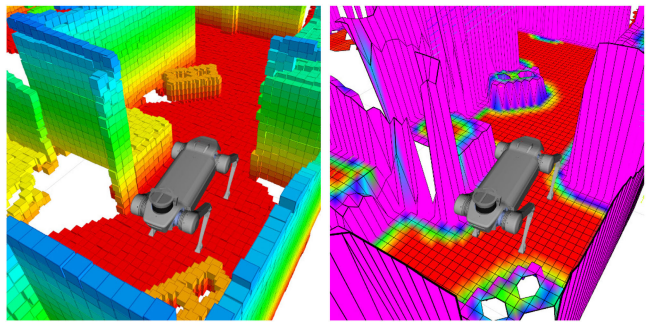

Fig. 3. Visualization of a generated traversability costmap. The left is the local point cloud map after fusion with multiple historical poses, and the right is the created elevation map with traversability cost.

IV. Perception Module

Since we have no prior knowledge of the environment, it is significant to generate maps for navigation and estimate poses online. First of all, for a frame of point cloud captured from LiDAR, we apply the min-cut segmentation algorithm [25] to extract the points of the target object. The points will be filtered out from the raw point cloud to avoid troubles for subsequent mapping functions. Meanwhile, the object's pose is served as the local goal for planning module and updates as the person moves. Then, the filtered point cloud is fed to the laser odometer of FLOAM[1] to estimate the pose of the current frame. As an advanced version of LOAM [26], FLOAM reduces the computational cost by three times and is more suitable for real-time applications.

We merge point clouds from LiDAR and depth cameras to deal with the unexpected obstacles in the environment. Specifically, the point clouds from sensors are transformed to a fixed coordinate system and summed. The single-frame point cloud only contains the current environmental information but does not store the past obstacles. Therefore, we use several frames to keep a memory of obstacles around. To improve efficiency, the mapping module is set up as a rolling window that always keeps a queue of detected obstacles around the robot. Different from Wooden *et al.* [16] using the time-based rolling window, we present a method based on the robot's moving distance. Given a collection $\mathcal{O}$ of the detected obstacles in the current and historical frames, we compute a parameterized subset of $\mathcal{O}$

$$Q(d) := \{q \in \mathcal{O} \quad | \quad \text{distance}(r_{\text{past}}, r_{\text{now}}) < d\} \tag{1}$$

where $\text{distance}(r_{\text{past}}, r_{\text{now}})$ is the distance between the past position of the robot and the current position. Only the obstacles included in $Q(d)$ are retained, which is, as the robot moves away from a certain distance d, the farthest obstacles pop out, and the newly detected obstacles update. The distance parameter is flexibly adjusted according to the size and complexity of the environmental space. The local point cloud map not only deals with the area in the sensor's field of view, but also stores the past obstacles or those behind the robot. Therefore, the robot can avoid moving in the wrong way during the path planning process.

Then, we create a discrete 2.5-D elevation map upon fused point cloud map, and it is based on the universal grid map library [27]. According to the elevation map, we estimate each grid's terrain traversability by three characteristics: the roughness r, the slope s, and the step height h, similar to Wermelinger *et al.* [10]. The traversability cost C_t of each cell can be formulated as follows:

$$C_t = w_1 \frac{r}{r_{\text{thre}}} + w_2 \frac{s}{s_{\text{thre}}} + w_3 \frac{h}{h_{\text{thre}}} \tag{2}$$

where w_1, w_2, and w_3 are the weights for them, respectively, whose sum is 1. The thresholds r_{thre}, s_{thre}, and h_{thre} are the allowed maximum values. When one of the features exceeds its threshold, its corresponding cost value is set to 1. An example of the visualization of a generated traversability costmap is shown in Fig. 3. If the cell is closer to the wall or the slope is larger, it is assigned a higher cost. The traversability cost ranges from 0 to 1, and the corresponding color is transited from red to purple.

V. Motion Planning Module

We present a simplification scheme for quadruped robots that allows fast computation of optimal movement commands and entails reacting timely in complex environments. This scheme is decoupled into three steps: First, we approximate the robot as a point and plan a rough path via a path-guided planner with the local traversability costmap. Then, we consider the robot as a rigid body and project it onto the 2-D plane as a rectangular footprint. The precise trajectory is generated by the optimization-based motion planner. Finally, we apply the receding horizon control strategy, which means only the first velocity control command in the trajectory is send to the robots in each control cycle. The planning mentioned earlier and the optimization process are repeated once the costmap updates, which significantly improves the robustness of robot movement .

[1][Online]. Available: https://github.com/wh200720041/floam

A. Path-Guided Planner

Similar to general wheeled robots, we adopt a variant A* algorithm [28] to search for a rough path guiding to the local goal in the 2-D costmap because of its optimality and search efficiency. Due to rapid environmental changes and the target's randomness movement, the map producer generates a new costmap at each planning iteration.

The traversability information of the terrain is stored as the cost value of cells in the costmap. We consider the traversability cost in the path search phase, and the cost value is integrated into the cost function. Then, the planner is inclined to find a path toward low traversability cost areas. This guided path provides initial waypoints for the back-end trajectory optimization function.

If the new path deviates too much from the previous one, the robot will change its directions back and forth. This means that the path is required to be as stable as possible over each planning iteration. Therefore, the path-guided planner keeps a small history of planned paths. Based on the previous path, cells' costs where the robot planned to go decrease in the new iteration.

B. Simplified Dynamics Model

It is critical for quadruped robots to maintain stability and balance in challenging spaces. Due to its underactuation during the trot gait, the control of a highly dynamic body is more complicated. Generally, in order to avoid slipping at the stance feet, the GRFs must remain in the *friction pyramids* [29].

Similar to Carlo *et al.* [13] and Focchi *et al.* [30], we assume that the GRFs are the only external forces acting on the robot and ignore Coriolis forces. Different from Focchi *et al.* [30], we do not think centrifugal forces are negligible because turning at a relatively high speed is a typical slippery scene. Based on the aforementioned conditions, we formulate the linear velocity and acceleration of the CoM $\dot{x}_{\text{com}}$ $\ddot{x}_{\text{com}}$, and the angular ones of the base w_b $\dot{w}_b$ as functions of GRF f_c, where c is the number of stance feet

$$\begin{bmatrix} I & \dots & I \\ [p_{\text{com},i}\times] & \dots & [p_{\text{com},i}\times] \end{bmatrix} \begin{bmatrix} f_1 \\ \dots \\ f_c \end{bmatrix} = \begin{bmatrix} m\hat{a} \\ I_g\dot{\hat{w}}_b \end{bmatrix} \tag{3}$$

$$\hat{a} = \ddot{x}_{\text{com}} + \dot{x}_{\text{com}} \times w_b + g \tag{4}$$

where $m \in \mathbb{R}$ is the robot's mass, $g \in \mathbb{R}^3$ is the gravity acceleration vector, $I_g \in \mathbb{R}^{3\times3}$ is the centroidal rotational inertia, and $p_{\text{com},i} \in \mathbb{R}^{3\times3}$ is the vector from the CoM to the position of the ith foot (see Fig. 4).

C. Footprint Representation

Different representations are applied to approximate the quadruped robot's footprints, including a point, a circle, and a rectangle. The roughest and straightforward representation is a point whose pose is (x, y) without the heading angle. This method does not take the robot's kinodynamic characteristics into account. Thus, the generated trajectory may not be feasible. It is efficient to use either the inscribed circle or the circumcircle,

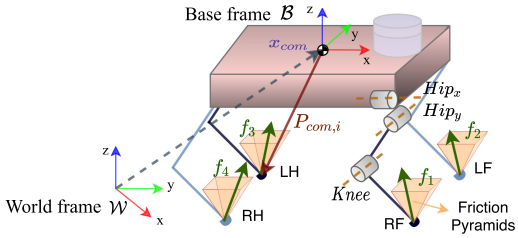

Fig. 4. World frame $\mathcal{W}$ and base frame $\mathcal{B}$ denote the world and body coordinates system, respectively. The left subscript shows the reference frame. The nomenclature about the quadruped robot is: right front (RF), left front (LF), left hind (LH), and right hind (RH). x_{com} represents the location of the CoM, and the vector $p_{\text{com},i}$ points the displacement from the CoM to the position of the ith foot in the world frame $\mathcal{W}$.

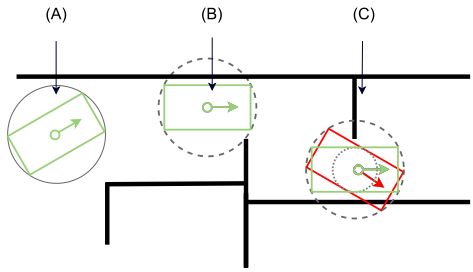

Fig. 5. Quadruped robot represented by the rectangle can pass through narrow spaces. (a) Rectangle footprint considers the orientation and the robot can rotate in open spaces. (b) It can avoid conservative solutions by circumcircle and select the exact angle to move forward. (c) Collision check guarantees the safety and therefore the robot is able to deal with several consecutive turns.

and the heading angle is tangential to the generated path. The former guarantees safety and feasibility, but it is too conservative to pass through narrow passages [see Fig. 5(b)]. The latter can handle these terrains but requires further safety verification [see Fig. 5(c)]. If the robot does not adjust its heading angle, its actual body will be stuck. Only planning with the rectangle footprint yields reliable and valid paths in more complicated situations. This method fully considers the quadruped robot's dynamic motion capability so that the robot can flexibly adjust its orientation. Therefore, we employ an accurate rectangle for trajectory optimization.

D. Trajectory Generation

We define the trajectory generation as an optimization problem taking the kinodynamic constraints into account and solve it using nonlinear programming. We describe the discretized trajectory consisting of a sequence of n robot poses $\mathbf{x}_i = (x_i, y_i, \theta_i)^T \in \mathbb{R}^2 \times S^1$ and $(n-1)$ time intervals $\Delta T_i \in \mathbb{R}^+$, where $(x_i, y_i) \in \mathbb{R}^2$ denotes the robot's position and $\theta_i \in S^1$ denotes its orientation. The time interval ΔT_i represents the time for the robot to move from pose $\mathbf{x}_i$ to subsequent pose $\mathbf{x}_{i+1}$. The set of parameters describing the trajectory is defined by

$$\mathcal{B} := \{\mathbf{x}_1, \Delta T_1, \mathbf{x}_2, \Delta T_2, \dots, \mathbf{x}_{n-1}, \Delta T_{n-1}, \mathbf{x}_n\}. \tag{5}$$

1) Objective Function: The robot is expected to follow the target quickly and flexibly, which requires the time intervals of

trajectory as short as possible. Thus, we formulate this issue as an optimization problem that minimizes the sum of the square of each time interval

$$\underset{\mathcal{B}}{\text{minimize}} \sum_{i=1}^{n-1} (\Delta T_i)^2. \tag{6}$$

As the time intervals of the trajectory is used as the optimization object, the robot tends to act aggressively. It directly causes the robot's velocity to change sharply and steps in accelerations, which not only consumes a lot of energy but also produces discontinuous torques causing damage to the hardware and affecting stability. In order for the robot to walk as stable as possible, it needs smooth acceleration during the execution period. We consider the sum of accelerations on the trajectory as the optimization object, formulated as follows:

$$\underset{\mathcal{B}}{\text{minimize}} \sum_{i=1}^{n} \|a_i\|^2 . \tag{7}$$

Moreover, the generated trajectory should be executed directly by robots. Thus, some kinodynamic constraints, including translational velocity and acceleration, should be considered. Besides, the distance between the boundary of the robot footprint and the obstacle points should be limited to ensure safety.

2) Kinematic Constraints: The quadruped robot has excellent mobility and dynamic motion capability, including walking forward, laterally, and rotating simultaneously, which means that the robot can walk flexibly in any direction. The translational velocity v_i and rotational velocity ω_i in pose $\mathbf{x}_i$ of discrete trajectory are calculated according to the Euclidean distance or angular difference with subsequent pose $\mathbf{x}_{i+1}$ and the time interval ΔT_i between both poses

$$\begin{bmatrix} v_{i,x} \\ v_{i,y} \end{bmatrix} = \frac{1}{\Delta T_i} \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix} \times \begin{bmatrix} x_{i+1} - x_i \\ y_{i+1} - y_i \end{bmatrix} \tag{8}$$

$$\omega_i = \frac{\theta_{i+1} - \theta_i}{\Delta T_i} \tag{9}$$

where translational velocity v is decomposed into the forward velocity v_x along the heading direction of robots, and the lateral velocity v_y perpendicular to the heading. Then, the velocity constraints are expressed as follows:

$$v_{\min} \le v_x \le v_{\max} \tag{10}$$

$$|v_y| \le |v_{\text{bound}}|, \quad |\omega| \le |\omega_{\text{bound}}| \tag{11}$$

where $v_{\min}$ and $v_{\max}$ are the lower and upper boundaries of the longitudinal translational velocity v_x, due to the difference in the forward and backward mobility of quadruped robots.

3) GRFs Constraints: Equations (3) and (4) establish a relationship between the robot's centroid kinematics and GRFs of the foot. Additionally, the linear acceleration a_i and angular acceleration α_i of the robot CoM in pose $\mathbf{x}_i$ can be approximately calculated separately by the following equations: the linear acceleration a_i and angular acceleration α_i of the robot CoM in pose $\mathbf{x}_i$ can be approximately calculated separately by the following equations:

$$a_i = \frac{2(v_{i+1} - v_i)}{\Delta T_i + \Delta T_{i+1}}, \quad \alpha_i = \frac{2(\omega_{i+1} - \omega_i)}{\Delta T_i + \Delta T_{i+1}}. \tag{12}$$

Then, the aforementioned kinematic parameters v_i, ω_i, a_i, and α_i at the trajectory are brought into (3) and (4) to obtain GRFs of the quadruped robot. We decompose the GRF f into three components along the dimensions x, y, and z. To keep stability, the component forces need to satisfy the friction formula as follows:

$$|f_z| \le f_{\max}, \quad |f_x| \le \mu |f_z|, \quad |f_y| \le \mu |f_z| \tag{13}$$

where μ is the friction factor. These constraints used as the optimization's constraints ensure the motion stability and avoid slippery.

4) Obstacles Constraints: In the costmap, the cells that obstacles occupy are assigned a very high lethal cost. The Euclidean distance between obstacle points and the polygonal boundary of the robot footprint can be calculated and denoted as d_k ($k = 1, 2, \ldots, P$ and P is the number of obstacle points). The obstacle points around the robot are considered as a constraint of the optimization function

$$d_k \ge d_{\min}. \tag{14}$$

To this end, we have set up a threshold parameter $d_{\min}$, adjusted according to the actual environment.

5) Nonlinear Programming: As we know, solving nonlinear programs with hard constraints is computationally expensive. In order to improve the efficiency of fast online solvers, the constraints are generally added to the objective function as additional quadratic penalty terms. Then, the nonlinear programs are converted to an unconstrained nonlinear least-squares optimization problem. The aforementioned constraints can be divided into single boundary constraints and dual boundary ones. The penalty functions used to approximate them are defined as a single boundary function (for obstacle constraints)

$$F_1(u) = \begin{cases} 0 & \text{if } u \ge u_{\min} + \epsilon \\ \gamma(\exp(|u - (u_{\min} + \epsilon)|) - 1) & \text{if } u < u_{\min} + \epsilon \end{cases} \tag{15}$$

and a dual boundary function (for kinodynamic constraints)

$$F_2(u) = \begin{cases} 0 & \text{if } u_{\min} + \epsilon < u < u_{\max} - \epsilon \\ \gamma(\exp(|u - (u_{\text{lower}}^{\text{upper}} \mp \epsilon)|) - 1) & \text{if others} \end{cases} \tag{16}$$

where $u_{\max}$ and $u_{\min}$ are upper and lower bounds of variable u, and γ and ϵ are factors that affect the accuracy of the approximation.

The whole trajectory optimization can be formulated as a nonlinear programming problem, in which there are multiple objective functions, and each item is given a weight

$$\mathcal{B}^* = \min_{\mathcal{B}} \left\{ \sum_{i=0}^{n-1} (\mathbf{a}^\mathsf{T} \mathbf{W_1} \mathbf{a} + \mathbf{b}^\mathsf{T} \mathbf{W_2} \mathbf{b} + \mathbf{c}^\mathsf{T} \mathbf{W_3} \mathbf{c}) \right\} \tag{17}$$

where

$$\mathbf{a} = [\Delta T_i, \|a_i\|]^\mathsf{T}$$

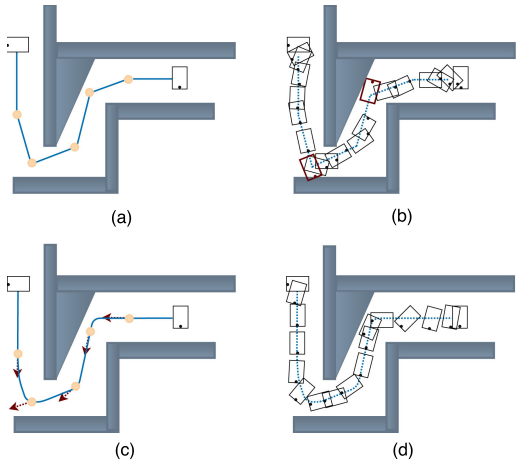

Fig. 6. (a) Variant A* algorithm finds a rough path from the start state to the target state. (b) Heading angle is always tangential to the path without optimization, and therefore, the generated trajectory will collide with the wall when the space is too narrow. (c) Our method will optimize the orientation and smooth the trajectory. (d) Generated trajectory using our method is further away from the obstacles.

$$\mathbf{b} = \left[\sum_{k=1}^{P} F_1(d_k)\right]^{\mathsf{T}}$$
$$\mathbf{c} = [F_2(v_i), F_2(\omega_i), F_2(f_i)]^{\mathsf{T}} \tag{18}$$

where $\mathcal{B}^*$ denotes the optimal solution vector, $\mathbf{a}$ is the associated term for optimizing the time and acceleration of the trajectory, $\mathbf{b}$ and $\mathbf{c}$ represent the penalty functions of obstacle constraints and kinodynamic constraints. $\mathbf{W_1}$, $\mathbf{W_2}$, and $\mathbf{W_3}$ are symmetric and positive definite matrix to weight and balance these factors.

For this problem, the efficiency and extensibility of the solver are the key considerations. The G^{2}o [31] is an open-source graph optimization framework widely used in SLAM problems. The overall optimization problem is described as a hypergraph in the G^{2}o framework, where the nodes represent the parameter variables to be optimized, and the hyperedges make connections between multiple nodes to represent objective functions. With this framework, it is easy to set numerous objective functions and constraints as we needed. Setting reasonable objective weights and optimizer parameters allows the functions to converge efficiently. The weight factors in the objective function are crucial, and we have to adjust the ratio of each weight properly in our experiments. For example, we make a tradeoff between movement speed and safety (the distance to obstacles in the environment).

A good initial guess is significant for solving nonlinear programs, and a poor one can cause the solver to get "stuck" on a wrong solution or fail to converge entirely. Our initial solution is sampled from a set of collision-free path points generated by the *path-guided planner*. The initial trajectory is composed of robot poses $\mathbf{x}_i$ located on the coarse path equidistant in space and time. The initial sampling distance between two points and time interval is configured manually. Fig. 6 shows the superiority of our proposed method.

E. Receding Horizon Control

Considering the external disturbances, the uncertainty of the local map, and dynamics model, we take a receding horizon strategy for commands control. The costmap describing the environment is updated, as a new frame of sensors is received. The optimization procedure is executed, and we get an optimal trajectory $\mathcal{B}^*$. Again, (8), (9), and (12) are applied to obtain the desired velocities and accelerations of the robot body, including forward ones v_x, a_x, lateral ones v_y, a_y, and rotational ones ω, α.

Only the first group of desired values on the trajectory is delivered to the gait controller as a control command, indicating the real-time control of the robot at the current position. However, there is an unavoidable time delay from sensor data acquisition to control command execution in realistic applications. Therefore, we set a time parameter t_d for correcting the control inaccuracy caused by the delay. The desired values of the group K in the trajectory are chosen as the eventual control command, where

$$\sum_{i=1}^{K} \Delta T_i \geq t_d. \tag{19}$$

VI. Experiments and Results

The person-following system described in this article is evaluated and verified in both simulated and physical environments with a quadruped robot JueYing Mini. It has an approximate rectangular shape of 0.7 m $\times$ 0.4 m and a mass of 22 kg. The sensors equipped on the robot are a Robosense RS-LiDAR-16 and a forward-facing (with a 20° downward pitch) Intel RealSense D435 depth camera. Both the perceptive and planning methods are performed online on an onboard PC (i7-8665UE processor at 1.7-GHz and 16-GB DRAM). To execute the planned trajectory, the robot JueYing uses a reactive trotting gait, in which the translation velocity is up to 1.5 m/s.

A. Simulation Experiments in Gazebo

1) Environments Setting: We construct three scenarios of different congestion levels in the Gazebo simulator to evaluate our planning scheme. A walking actor is the target to be followed and walks at a constant speed along the defined route. Boxes, bookcases, and bricks of different sizes are placed randomly and compactly within this space. The first and easiest scene has a 4 m $\times$ 9 m task area, which starts with a relatively open space and follows by a corridor. The corridor's minimum gap distance is 1 m, whereas the width of the robot is 0.4 m. The robot is required to follow the actor at a safe distance from the upper left corner to the lower right corner. The next scene contains lots of messy obstacles, where the area is 4.5 $\times$ 13 m. Narrow passages occupy 80% of the route, the minimum gap distance of which is only 0.7 m. The robot follows the actor through seven turns, from the lower right corner to the upper left corner. The last scene is full of multiple narrow and continuous passages, which has the same size as the second scene.

2) Simulation Results: The key of our proposed method is to consider the kinodynamic model of the quadruped robot in

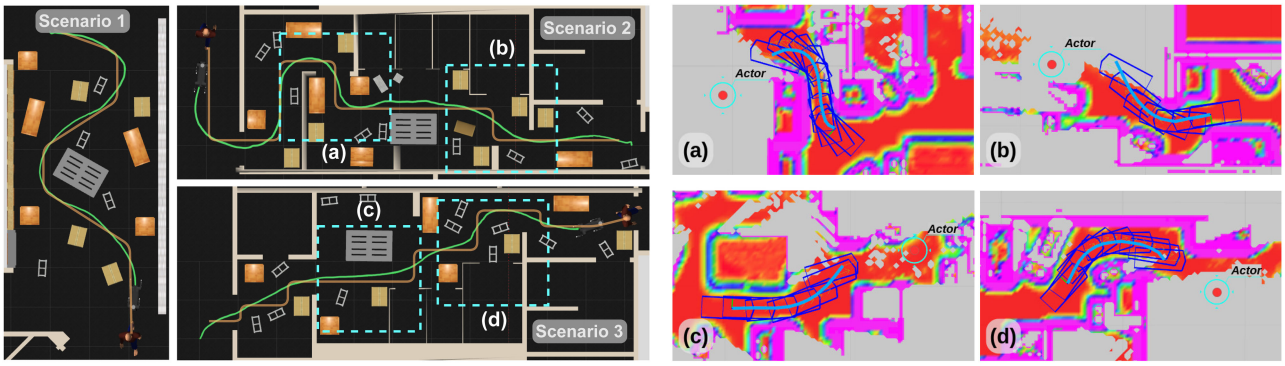

Fig. 7. Left is the route of the person (orange) and the trajectory generation by our method (green). This figure shows the experimental results in three different scenarios. The right is the close-up of details about several narrow spaces and the corresponding traversability costmap. The cyan circles represent the actor's position. The blue rectangle represents a set of footprints of the robot on the predicted trajectory, and the triangle head indicates the orientation.

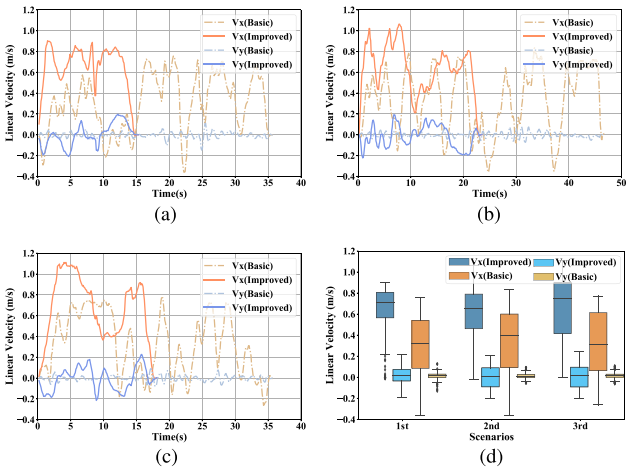

Fig. 8. Comparison about the linear velocities includes forward speed v_x and lateral v_y between the basic method and our improved method. (a)–(c) Plotted curves for moving in different scenes, respectively. (d) Statistical information, where the colored box indicates the high quartiles of the velocity values. (a) Scenario 1. (b) Scenario 2. (c) Scenario 3. (d) Linear velocity.

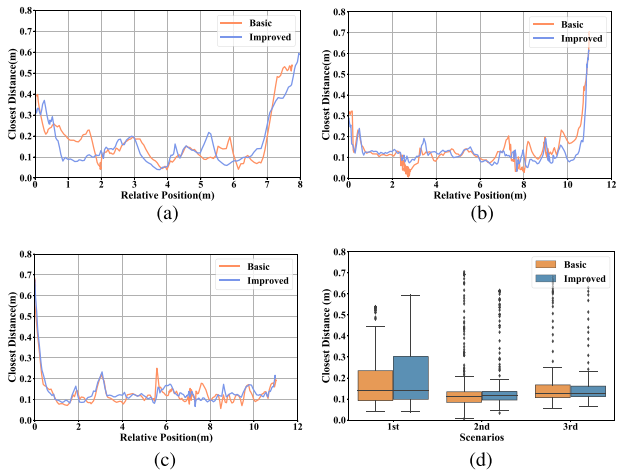

Fig. 9. Comparison about the closest distance from obstacles between the basic method and our improved method. (a)–(c) Plotted curves for moving in different scenes, respectively, where the horizontal axis represents the relative distance that the robot moves forward along the route from the starting position. (d) Interpreted in the same way as Fig. 8(d). (a) Scenario 1. (b) Scenario 2. (c) Scenario 3. (d) Closest distance.

the optimization to generate a flexible and robust motion. The application of GRFs constraints is critical, which guarantees the effectiveness and stability of the trajectory. We set up a *basic planner* that neglects kinodynamic model in trajectory optimization to compare with our improved method. It has the same objective functions (6), (7) and obstacle constraints as the proposed method. Then, we count three indicators of two methods: the success rate, the linear velocity, and the closest distance to the surrounding obstacles.

In the simulator, the actor move along the path recorded in advance, as shown in the left panel of Fig. 7. Initially, the actor walks at a constant velocity of 1 m/s. It demonstrates that the improved method makes the robot follow the actor successfully. But it fails to keep up with the actor with the basic planner, because of the poor flexibility of the trajectory, and the robot is easy to get stuck in obstacles. We have to reduce the walking speed until 0.5 m/s so that the robot can catch up at a slower speed. Fig. 8(d) indicates that the average linear velocity of basic planner is almost half of improved, and the lateral velocity is almost zero. It can be seen from Fig. 8(a)–(c) that the forward speed of the robot changes between positive and negative value, and it is because that the robot usually gets stuck and has to back up. The relatively smoother speed satisfies the dynamic characteristics of the robot. It clearly shows that our proposed method achieves far flexible results than the basic. The detailed results are available at.[2] The detailed traversability costmap and generated trajectories are presented in Fig. 7 (right).

Fig. 9(a)–(c) draws the curves of the distance of the robot to the surrounding obstacles, where the distance value is the closest Euclidean distance between the polygonal boundary of the footprint and obstacles. Fig. 9(d) indicates that the average and minimum distance values of the improved method are higher, where the distance value is the closest Euclidean distance between the polygonal boundary of the footprint and obstacles. Especially in the much crowded scenarios 2 and 3, the improved

[2][Online]. Available: https://youtu.be/0qz1RORqMj8

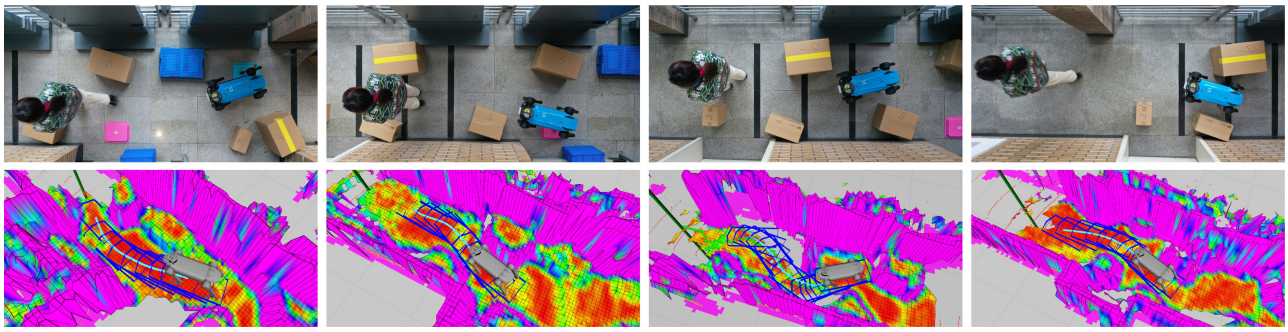

Fig. 10. Image flows of following a walking person in the confined corridor and the corresponding traversability costmaps. The planned local trajectory is shown as a sequence of blue polygon boxes on the bottom.

TABLE I
SUCCESS RATE OF THE EXPERIMENT IN DIFFERENT SCENARIOS

	Scenario1	Scenario2	Scenario3
Basic Planner	18 / 20	11 / 20	14 / 20
Improved Motion Planner	20 / 20	18 / 20	19 / 10

method has obvious superiority. And the robot is able to maintain a fast moving speed while still ensures safety. But the robot with base planner tends to lose control and crash into the obstacle.

Moreover, we apply our improved method and the basic method to repeat experiments 20 times in each scenario. The success rate of the improved planner is above 90%, significantly higher than the basic, as shown in Table I. There are two main reasons: First, the basic planner cannot match the movement characteristics of the robot and has poor ability to regulate the body, then the robot sometimes gets stuck completely. In addition, without the consideration of GRFs constraints, the generated trajectory is terrible to execute.

B. Field Experiment

The scheme has also been experimentally validated in two different real environments: a narrow corridor with cluttered obstacles and outdoor wooded areas undulating terrain. We also place messy cardboard boxes that have to be bypassed and planks that the robot can walk over. A person walks randomly as a moving object, and the robot JueYing is commanded to follow through some confined spaces. Pose estimation, mapping, and motion planning algorithms are all performed online without prior knowledge about the environments.

1) Indoor Corridor Scenes: Fig. 10 (top) shows the whole process of the robot JueYing in the confined corridor, and Fig. 10 (bottom) represents the corresponding traversability costmap. The image flow clearly shows that it successfully follows a moving person and adjusts its body to avoid collisions. The total length is 7.65 m with the closest distance to the obstacle $d_{\min}$ is 0.12 m. For obstacles with a smaller height (such as planks), the traversability cost is not very high. Therefore, the robot can walk over them if necessary (see first and second snapshots in Fig. 10). While for larger obstacles, it adjusts its steering and bypasses them (see third and fourth snapshots in Fig. 10).

In order to obtain the distance from obstacles in real environments effectively, we mount sonar sensors around the body of the robot, as shown in Fig. 11. It is a reliable distance measuring equipment with an accuracy of 1 cm. The beam angle of each probe is 120° , and we calculate the closest distance between the robot body and the surrounding objects by receiving data from multiple probes. The quantitative results are shown in Table II. There is only 0.1-m clearance on both sides of the robot at the narrowest part of the passage, and the robot aligns to maximize the distance from the obstacles. This reflects the superiority of moving flexibility.

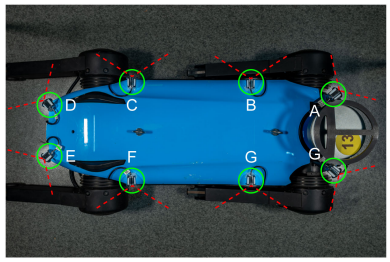

Fig. 11. Layout of sonar sensors. Eight probes (A–G) are mounted around the body of the JueYing.

TABLE II
STATISTICAL DATA OF THE EXPERIMENT AT INDOOR SCENES

	Minimum	Maximum	Average
Linear Velocity (m/s)	/	0.85	0.48
Closest Distance (m)	0.12	/	0.23

In addition, the elapsed time of all perception and planning algorithms is shown in Table III. Each of these submodules is assigned a separate thread and satisfied to execute at 10 Hz on the onboard computer. The time delay of executing control commands is at worst no more than 150 ms, which ensures the effectiveness of the motion planning algorithm.

2) Outdoor Grove Scenes: To verify our method is effective and robust in different terrains, we conduct several tests in an outdoor wooded area. The area is with boulders, trees, and undulating ground (lawn ground with slope up to 16). The detailed image flows are presented in Fig. 12. A person walks through the grove at a constant speed and goes up and downhill. The cardboard boxes are also placed in the scene as uncrossable obstacles. The robot tends to generate an easy-to-pass trajectory

Fig. 12. Image flows of experimental trials on a lawn with undulating terrain and messy obstacles. From top to bottom: through multiple narrow passages with the closest distance gap 0.58 m and the maximum terrain slope 16.

TABLE III
ELAPSED TIME OF ALGORITHMS AS RUN ON ONBOARD PC IN FIELD EXPERIMENTS

	Best	Average	Worst
PointCloud Segmentation[1]	10.003	21.085	60.924
Lidar Odometry[1]	10.713	25.720	76.026
Local Mapping[1]	30.228	68.150	141.094
Guided-Path Planner[1]	3.575	7.356	26.559
Trajectory Optimization[1]	0.148	44.474	94.375
Execution Delay[12]	61.278	92.157	134.964

[1]All times are given in milliseconds, the duration of the recorded data is 69.498 s.
[2] Execution delay is the duration from receipt of sensor data to the execution of control commands by the robot.

considering the traversability cost of ground and interval between obstacles, and follows closely behind the moving person with an adaptive gait. It always keeps a distance of 1.5 m from the object. It indicates that our method is also effective and robust for quadruped robots in complex terrain.

VII. CONCLUSION

In this article, we propose a novel optimization-based motion planning scheme for quadruped robots to follow moving objects in confined spaces. It integrates environment perception (including object segmentation and local terrain description) and local trajectory generation methods. It only relies on the robot's onboard sensors and embedded computers to execute all algorithms. The motion planning efficiency ensures the robustness of following a moving person in a crowded environment, without any prior environmental information, which is suitable for field applications.

For efficiency, we present a simplified kinodynamic model to optimize the trajectory without losing the flexibility of quadruped motion while ensuring the executable and effective. We planned the optimal trajectory based on the robot's rectangular footprint in the traversability costmap. This guarantees the trajectory's safety to a great extent and prevents the robot body from colliding with the environment. We conduct simulations and real experiments to demonstrate the proposed method. It shows that the success rates in diverse scenes are all over 90%.

For further research, we will focus on extending the planning of quadruped robots to three-dimensional confined spaces. This will greatly expand the application scenarios of quadruped robots.

REFERENCES

[1] A. Goldhoorn, A. Garrell, R. Alquézar, and A. Sanfeliu, "Continuous real time POMCP to find-and-follow people by a humanoid service robot," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 741–747.
[2] K. Bohlmann, A. Beck-Greinwald, S. Buck, H. Marks, and A. Zell, "Autonomous person following with 3D LIDAR in outdoor environment," *J. Autom. Mobile Robot. Intell. Syst.*, vol. 7, no. 2, pp. 24–29, 2013.
[3] J. Laneurit, R. Chapuis, and C. Debain, "TRACKBOD, an accurate, robust and low cost system for mobile robot person following," in *Proc. Int. Conf. Mach. Control Guid.*, 2016.
[4] M. J. Islam, J. Hong, and J. Sattar, "Person-following by autonomous robots: A categorical overview," *Int. J. Robot. Res.*, vol. 38, no. 14, pp. 1581–1618, 2019.
[5] H. Dong, C. Y. Weng, C. Guo, H. Yu, and I. M. Chen, "Real-time avoidance strategy of dynamic obstacles via half model-free detection and tracking with 2D lidar for mobile robots," *IEEE/ASME Trans. Mechatronics*, early access, Oct. 30, 2020, doi: 10.1109/TMECH.2020.3034982.
[6] C. Sprunk, B. Lau, P. Pfaffz, and W. Burgard, "Online generation of kinodynamic trajectories for non-circular omnidirectional robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 72–77.
[7] C. Sprunk, B. Lau, P. Pfaff, and W. Burgard, "An accurate and efficient navigation system for omnidirectional robots in industrial environments," *Auton. Robots*, vol. 41, no. 2, pp. 473–493, 2017.
[8] M. Kennedy, D. Thakur, M. A. Hsieh, S. Bhattacharya, and V. Kumar, "Optimal paths for polygonal robots in SE (2)," *J. Mechanisms Robot.*, vol. 10, no. 2, 2018, Art. no. 021005.
[9] Y.-D. Hong and B. Lee, "Real-time feasible footstep planning for bipedal robots in three-dimensional environments using particle swarm optimization," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 1, pp. 429–437, Feb. 2020.
[10] 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.*, 2016, pp. 1184–1189.
[11] R. Triebel *et al.*, "SPENCER: A socially aware service robot for passenger guidance and help in busy airports," in *Field and Service Robot.*, Berlin, Germany: Springer, 2016, pp. 607–622.
[12] H. Gong, J. Sim, M. Likhachev, and J. Shi, "Multi-hypothesis motion planning for visual object tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 619–626.
[13] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–9.

[14] M. M. Venancio, R. S. Goncalves, and R. A. Bianchi, "Terrain identification for humanoid robots applying convolutional neural networks," *IEEE/ASME Trans. Mechatronics*, early access, Sep. 1, 2020, doi: 10.1109/TMECH.2020.3020781.
[15] A. Chilian and H. Hirschmüller, "Stereo camera based navigation of mobile robots on rough terrain," in *Proc.IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 4571–4576.
[16] 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.*, 2010, pp. 4736–4741.
[17] M. Zucker *et al.*, "Optimization and learning for rough terrain legged locomotion," *Int. J. Robot. Res.*, vol. 30, no. 2, pp. 175–191, 2011.
[18] M. A. Arain, I. Havoutis, C. Semini, J. Buchli, and D. G. Caldwell, "A comparison of search-based planners for a legged robot," in *Proc. 9th Int. Workshop Robot Motion Control*, 2013, pp. 104–109.
[19] C. Mastalli *et al.*, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 1096–1103.
[20] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *Int. J. Robot. Res.*, vol. 30, no. 2, pp. 236–258, 2011.
[21] P. Fankhauser, M. Bloesch, C. Gehring, M. Hutter, and R. Siegwart, "Robot-centric elevation mapping with uncertainty estimates," in *Mobile Service Robot. Singapore: World Scientific*, 2014, pp. 433–440.
[22] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robot. Autom. Lett.*, vol. 3, no. 4, pp. 3019–3026, Oct. 2018.
[23] A. W. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5148–5154.
[24] Y. Zhao, X. Chai, F. Gao, and C. Qi, "Obstacle avoidance and motion planning scheme for a hexapod robot Octopus-III," *Robot. Auton. Syst.*, vol. 103, pp. 199–212, 2018.
[25] A. Golovinskiy and T. Funkhouser, "Min-cut based segmentation of point clouds," in *Proc. IEEE 12th Int. Conf. Comput. Vis. Workshops*, 2009, pp. 39–46.
[26] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst. Conf.*, 2014, vol. 2, no. 9.
[27] P. Fankhauser and M. Hutter, "A universal grid map library: Implementation and use case for rough terrain navigation," in *Robot Operating System*, Berlin, Germany: Springer, 2016, pp. 99–120.
[28] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, Jul. 1968.
[29] C. D. Bellicoso, C. Gehring, J. Hwangbo, P. Fankhauser, and M. Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *Proc. IEEE-RAS 16th Int. Conf. Humanoid Robots*, 2016, pp. 558–564.
[30] M. Focchi *et al.*, "High-slope terrain locomotion for torque-controlled quadruped robots," *Auton. Robots*, vol. 41, no. 1, pp. 259–272, 2017.
[31] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G^2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 3607–3613.

Zhen Zhang received the B.S. degree in communication engineering from the Zhejiang University of Technology, Hangzhou, China, in 2018. He is currently working the Ph.D. degree in control science and engineering with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

His latest research interests include motion planning of quadruped robots and mobile robots.

Jiaqing Yan received the B.S. degree in marine engineering and technology from Zhejiang University, Hangzhou, China, in 2019. She is currently working toward the M.S. degree in control science and engineering from the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

Her latest research interests include mobile robots and motion planning.

Xin Kong received the B.S. degree in automation from the Harbin Institute of Technology, Harbin, China, in 2018. He is currently working toward the M.S. degree in control science and engineering from the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

His latest research interests include robotic perception, SLAM and 3D computer vision.

Guangyao Zhai received the B.S. degree in automation from Northwestern Polytechnical University, Xi'an, China, in 2018. He is currently working toward the M.S. degree in control science and engineering from the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China.

His latest research interests include geometric learning and multi-object tracking.

Yong Liu (Member, IEEE) received the B.S. degree in computer science and engineering and the Ph.D. degree in computer science from Zhejiang University, Hangzhou, China, in 2001 and 2007, respectively.

He is currently a Professor with the Institute of Cyber-Systems and Control, Zhejiang University. He has authored or coauthored more than 30 research papers on machine learning, computer vision, information fusion, and robotics. His main research interests include intelligent robot systems, robot perception and vision, deep learning, Big Data analysis, and multisensor fusion.