A Closed-Loop Shared Control Framework for Legged Robots

Peng Xu[1*], Zhikai Wang[1*], Liang Ding[1], *Senior Member, IEEE*, Zhengyang Li[1], Junyi Shi[2], Haibo Gao[1], Guangjun Liu[3], *Senior Member, IEEE*, Yanlong Huang[4]

***Abstract*—Shared control, as a combination of human and robot intelligence, has been deemed as a promising direction towards complementing the perception and learning capabilities of legged robots. However, previous works on human-robot control for legged robots are often limited to simple tasks, such as controlling movement direction, posture, or single-leg motion, yet extensive training of the operator is required. To facilitate the transfer of human intelligence to legged robots in unstructured environments, this paper presents a user-friendly closed-loop shared control framework. The main novelty is that the operator only needs to make decisions based on the recommendations of the autonomous algorithm, without having to worry about operations or consider contact planning issues. Specifically, a rough navigation path from the operator is smoothed and optimized to generate a path with reduced traversing cost. The traversability of the generated path is assessed using fast Monte Carlo tree search (FastMCTS), which is subsequently fed back through an intuitive image interface and force feedback to help the operator make decisions quickly, forming a closed-loop shared control. The simulation and hardware experiments on a hexapod robot show that the proposed framework gives full play to the advantages of human-machine collaboration, and improves the performance in terms of learning time from the operator, mission completion time, and success rate than comparison methods.**

I. Introduction

LEGGED robots are often expected to be working in complex environments, such as disaster rescue and planet exploration [1]. However, challenging environments place great demands on the autonomous perception, planning, and decision-making capabilities of robots. Specifically, autonomous systems are highly susceptible to unexpected changes in the environment like atrocious weather. They heavily rely on the system's perception capabilities, making it difficult to ensure the reliability of auto-control behavior in the presence of inaccurate environmental information. On the other hand, humans have advanced biological intelligence like the ability of logical thinking, figurative thinking, high-dimensional decision-making, comprehensive understanding, etc. Therefore, it is more reliable to perform tasks combining human and robot intelligence, such as using human intelligence for high-level perceptual understanding, task decomposition, and decision-making and using robots for data collection, calculation, robot stabilization and motion control.

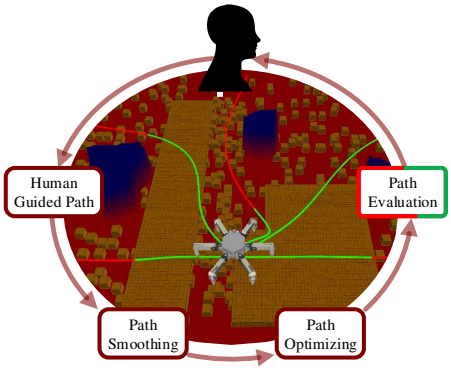

Fig. 1: The shared control framework.

Although shared control has been extensively studied in recent years, especially in the field of autonomous driving [2], [3], few works have been done for legged robots. Compared to vehicles and unmanned aerial vehicles (UAV), legged robots make contact with terrains using discrete footholds and have high-dimensionality of the contact search-space [4], which brings more challenges for the shared control of legged robots. For example, a collision-free path guided by operators is still no guarantee that the legged robot can track it; or the motion generated by the robot may be incomprehensible to humans. The planning and decision-making tasks for legged robots are complicated, which need to consider not only the walking path of robots but also contact planning issues, namely, gait transition, foothold selection, and posture optimization. These issues limit the shared-control scheme for legged robots to direct manual control, exerting much pressure and workload on operators. Therefore, designing an intuitive, easy-to-operate, comfortable shared control method for legged robots is highly desirable.

A. Related Works

The complexity of contact planning, the high dimensionality of the search space, the discontinuities, instabilities, and non-convexity of the robot dynamics, as well as the non-convexity of the terrain environment, among other factors, make locomotion synthesis of legged robots a challenging problem [4]. To simplify matters, many efforts separate contact planning from trajectory planning, while gait, foothold, and posture of the robot are also considered separately in contact planning [5]–[7]. Although such treatment reduces planning

This work was supported by the National Key Research and Development Program of China (Grant No.SQ2019YFB130016), the National Natural Science Foundation of China (Grant No. 91948202, 51822502), the Fundamental Research Funds for the Central Universities (Grant No. HIT.BRETIV.201903), and the "111 Project" (Grant No. B07018). (*Corresponding author:* Liang Ding.)

*These authors contributed equally to this work and should be considered as co-first authors.

[1]State Key Laboratory of Robotics and Systems, Harbin Institute of Technology, Harbin 150001, China. Email: pengxu_hit@163.com

[2]Department of Electrical Engineering and Automation, Aalto University, Finland.

[3]Department of Aerospace Engineering, Toronto Metropolitan University, Toronto, ON M5B 2K3, Canada.

[4]School of Computing, University of Leeds, Leeds LS29JT, UK.

Video materials: https://sites.google.com/view/shared-control-leggedrobot/home

time considerably, the results are unlikely to achieve optimality. In contrast, some work has attempted to formulate contact planning as an optimization problem to be solved by integer optimization algorithms [8], [9], but it is not able to meet real-time requirements. Monte Carlo Tree Search (MCTS) [10] is a method for solving sequential decision problems. Peng et al. proposed a sliding-MCTS method for solving state-sequence planning problems for legged robots and demonstrated that the robots have better traversability in sparse-foothold environments [11], but it still takes a long time to search for the result. Although progress has been made in terms of autonomous motion, the algorithms above cannot satisfy the full autonomy of legged robots due to imperfect environment sensing and localization algorithms.

The flexibility of a legged robot comes from high degrees of freedom and discrete contact ability, while the drawback is that it increases the complexity of human-robot cooperative control. Existing works for legged robots concentrate on direct control by humans. For example, researchers use different input tools such as a joystick [12], [13], 3D input device [14], [15], and user interface [16], [17] to directly control the robot's posture, legs' position [18], and gait modes. However, the tedious operations significantly increase the burden on operators and are highly error-prone. In recent years, some shared control work combined with auxiliary algorithms has also emerged, such as giving the robot's destination through wearable devices [19] or sensing the gesture of humans [20], then robots navigate there autonomously. However, the existing research on human-robot coordination mechanisms for legged robots is too straightforward, considering only limited environmental factors to perform simple movements. In addition, humans and robots are relatively independent in the collaborative process. Operators mostly make decisions based on subjective consciousness, lacking feedback information from the robot.

Shared control develops rapidly on robots with low degrees of freedom, such as wheeled and aerial vehicles [21]–[23]. Researchers focus on planning navigation paths for robots consistent with human intention. For example, Lee K. H. et al. provide continuous forward guidance for wheeled robots by sketching paths on images, complementing the local intelligence of the robot to perform obstacle avoidance path planning for multiple scenarios [24]. Fennel M. et al. designed a force feedback device via immersive human-robot interaction to generate paths that both meet the human intention and satisfy the constraints of the kinematic model of a wheeled robot [25]. In the field of UAVs (unmanned aerial vehicles), due to their high-speed requirements, researchers hope to plan paths that conform to human expectations while being smooth, search fast, and kinetic constraints satisfied [26]. In contrast, shared control for legged robots aims to make the robot's path as secure, affordable, and, most importantly, trackable as feasible. Nevertheless, few works have considered these demands. A similar shared control scheme for path-level interaction is proposed for high-dimensional robots by Islam et al. [27]. The operator only intervenes when the autonomous algorithm is unable to continue and operates completely manually, without any assistance from machine intelligence.

The development of technology makes shared control schemes adopt various human-robot interaction means. In the Darpa challenge, legged robots are operated mainly through software interfaces [28]. Besides, Kurisu M. et al. designed a manipulation platform with force, and haptic feedback for large hexapod robots [15]. However, these input tools are all cumbersome to operate, requiring quick reaction and professional knowledge. In order to reduce its difficulty, voice [29], gestures recognition [30], human posture detection [31], and other means that do not require specific equipment are emerging. Recently, contact input methods such as electromechanical signals and brain-computer interfaces have also been used [32]. It can be seen that humans tend to interact with robots using a more natural and straightforward way. The work presented in this paper can provide a basis for applying user-friendly devices such as brain-machine interfaces for the shared control of legged robots.

B. Contribution

In this paper, we design a shared control framework for legged robots, which allows an operator to control legged robots to perform navigation tasks by making decisions on the evaluated results rather than controlling the motion of the robot without a break. The input of the framework is a rough human-guided path, which will be smoothed by B-spline and optimized by a stochastic trajectory optimization approach considering traversing costs, including collision, foothold density, physical characteristics, distance, and smoothing cost. Then, the optimized path is evaluated by fast Monte Carlo tree search (FastMCTS) in terms of the traversability of discrete contact state sequences. Finally, the quality of the optimized path is fed back to the operator via images and touch signals so as to help the operator make decisions promptly. Our main contribution includes four aspects:

1) An easy-to-operate shared control framework is proposed for legged robots. The novelty is that operator does not need to be involved throughout the entire process and only makes decisions based on the recommendations of the autonomous algorithm, reducing operational difficulties while ensuring the traversability of robots in complex environments.
2) A closed-loop human-robot collaborative path planning method is proposed, which is able to generate safe and smooth paths while considering different environmental costs with compatible human intention functions.
3) A fast Monte Carlo tree method is proposed to evaluate the trackable degree of the planned paths by searching for a feasible contact state sequence of legged robots with a shorter search time compared with related methods.
4) A new form of an interactive system is designed. In addition to joystick input, haptic, and image feedback, motion prediction has been added to the system to make it safer and more reliable for operators.

The rest of this article is organized as follows. Section II-A introduces the method overview. Human-guided path method

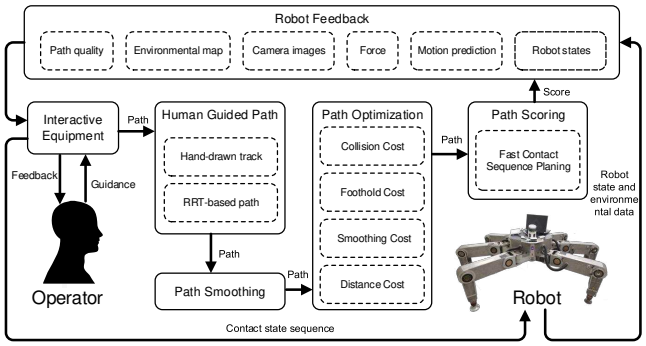

Fig. 2: The workflow of the proposed method. There are five main parts, including human-guided path, path smoothing, path optimization, path scoring, and robot feedback, corresponding to the five parts of the method. The operator provides the initial path and final decisions via an interactive interface to generate a sequence of contact states for the robot. The robot-machine aspect optimizes the operator's input path and evaluates its quality; finally, feedback information such as evaluation results, environmental information, and robot status is sent back to the operator, forming a closed loop of cooperative control.

is described in Section II-B. Section II-C presents the path-optimized method considering traversing costs. The methods for evaluating path and feedback are proposed in Section II-D and Section II-E respectively. Simulations and hardware experiments are presented in Section III. Finally, Section IV concludes this article, and Section V discusses the advantages and disadvantages of the proposed methods.

II. Methodology

A. Method Overview

As shown in Fig. 2, the entire algorithm framework starts with human guidance until the processed information, such as the evaluated path or force signal, feeds back to the operator to form a closed loop. The operator first gives a reference path through the remote control handle. Then the algorithm adjusts its initial path by smoothing and optimizing steps, considering the environmental characteristics, to generate a safer and smoother path. After that, the proposed FastMCTS method is adopted to evaluate the optimized path from the view of contact passability. Finally, the evaluated result will be fed back to the operator using an intuitive way. During the whole process, the operator only needs to provide some initial information about the reference path (such as the direction of the path or a sketch of the path) and then decide on the robot navigation path based on the path score evaluated by the evaluation algorithm.

B. Human Guided Path

When it comes to the shared control of mobile robots, a more intuitive and comfortable way for humans is to simplify the robot as a particle and guide its navigation path.

1) Guided Path Input Method: Two human-guided path input methods are introduced in this essay. The first one is the semi-manual method. Humans only need to give the target destination, and then autonomous path planning algorithms like rapidly exploring random tree (RRT) [33] are adopted to generate the whole path to avoid obstacles. The operator simply gives the forward target point to generate the initial path.

Specifically, the operator sets the robot's forward direction α and forward distance r_φ through the haptic device, and then gets the target location $\boldsymbol{P}$ which can be calculated in robot coordinates as

$$\boldsymbol{P} = r_\varphi \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix}. \tag{1}$$

Then, with $\boldsymbol{P}$ as the destination and the position of the robot as the starting point, an obstacle avoidance path can be planned by RRT as the reference trajectory for subsequent optimization. In this way, the time consumed by subsequent trajectory optimization can be significantly reduced. The STOMP (Stochastic Trajectory Optimization) algorithm [34] can also be prevented from falling into local optimum and failing to find a viable path.

The second input method is manual path drawing. When the path planned by autonomous planning algorithms cannot meet the needs of human beings, such as controlling the robot to approach the obstacle areas or forcing the robot to take a specific path, completely manual input methods can be a better way. This paper adopts a haptic device to teach the path because of its advantages of intuitive input and tactile feedback. The position of the handle end is mapped to the speed of the virtual control object via

$$\boldsymbol{v} = k_m \boldsymbol{d}, \tag{2}$$

where $\boldsymbol{d}$ denotes the displacement vector of the handle end with reference to an initial point; $\boldsymbol{v}$ represents the speed vector of the virtual control object; k_m is the displacement amplification factor. In this way, the operator can draw the running trajectory of the virtual robot by controlling the position of the handle.

2) Path Smoothing Method: Considering that the path generated by RRT or human teaching is tortuous and unnatural, the cubic uniform B-spline is used to smooth the initial path. The equation for a standard B-spline is

$$P(u) = \sum_{i=0}^{n} d_i N_{i,k}(u), \tag{3}$$

where $d_i (i = 0, 1, \ldots, n)$ is the control point, also known as the DeBoor point; $N_{i,k}(i = 0, 1, \ldots\ldots, n)$ is the basis function of k-degree canonical B-spline. A smooth B-spline curve can be generated by sampling the value of the guided path as data points, which can provide a better initial solution for subsequent optimization.

Although the path guided by humans can be a reasonable reference for robots, the rough sketched path often ignores many factors, such as the traversing cost or whether the trajectory is trackable. Therefore, it makes sense to improve the security and executability of guided paths combined with robot intelligence.

C. Path Optimization Method

Since it is difficult for humans to consider the constraints of multiple environmental characteristics simultaneously, the STOMP method is adopted to optimize the smoothed trajectory. STOMP is a trajectory optimization framework, that relies on generating noisy trajectories to estimate a better trajectory at a lower cost [34]. Not only does STOMP support adjustments to the reference trajectory, but the method uses derivative-free stochastic optimization, which allows the method to add multiple constraints without affecting its convergence. Therefore, STOMP is well suited for our problem and guarantees that the generated paths are cost-optimized given multiple environmental constraints. In order to generate a path more suitable for the legged robot to track, the optimization problem to be solved can be expressed as the following equation,

$$\min_{\theta} \sum Q_{(.)}(\boldsymbol{\theta}), \tag{4}$$

where $Q_{(.)}(\boldsymbol{\theta}) = \sum_{i=1}^{N} q_{(.)}(\boldsymbol{\theta}_i)$ denotes different cost functions; $q_{(.)}$ denotes the cost in point $\boldsymbol{\theta}_i$; $\boldsymbol{\theta} \in \mathbb{R}^{2\times N}$ represents normalized discretized path points of $\boldsymbol{x}$; $\boldsymbol{x} \in \mathbb{R}^{2\times N}$ represents discretized path points that the robot should actually follow in the Cartesian space, and N is the number of the discretized points. It is worth mentioning that cost normalization can help match the dimension of noisy increment to ensure that the generated noisy exploratory paths can explore Cartesian space as much as possible.

In order to obtain a reliable trajectory, several costs for legged robots are considered.

1) Collision Cost: Completely forbidden areas, such as areas that cannot be crossed, untouchable terrains such as silt, and terrain with steep slopes, are treated as obstacles. Then the corresponding signed distance field (SDF) is established. SDF reflects the closest distance of a point to the boundary of the obstacle [35], and the value of SDF outside the obstacle is positive and inside the obstacle is negative. and the obstacle cost is

$$q_{obs}(\boldsymbol{\theta}_i) = \|\boldsymbol{x}_{i+1} - \boldsymbol{x}_i\| \max\left(r + \varepsilon - f(\boldsymbol{x}_i), 0\right), \tag{5}$$

where $f(\boldsymbol{x}_i)$ represents the value of SDF at $\boldsymbol{x}_i$; r denotes the size of a robot; ε is the safety margin set aside.

2) Feasible Foothold Density Cost: Walking in areas with a higher density of feasible footholds can ensure the passability of legged robots. Once the environmental map is ready, the binary map of footable and non-footable can be segmented by establishing a cost map. Therefore, the foothold-density cost of the robot can be expressed as

$$q_{fea}(\boldsymbol{\theta}_i) = k_{fea}\left(N_{s,\max} - D_s(\boldsymbol{\theta}_i)\right), \tag{6}$$

where $D_s(\boldsymbol{\theta}_i)$ represents the number of footholds in the area of s around the path point $\boldsymbol{x}_i \in \mathbb{R}^2$ corresponding to $\boldsymbol{\theta}_i$ in Cartesian space coordinates; $N_{s,\max}$ represents the theoretical upper limit of the number of footholds in this region; k_{fea} is the weight coefficient. The larger the value of $q_{fea}(\boldsymbol{\theta}_i)$ is, the higher the crossing cost the robot will pay.

3) Physical Characteristic Cost: In the wild environment, robots need not only to deal with geometric obstacles but also to meet terrains with various physical properties. However, these terrains may have different traversing costs. For example, the traversing cost of a dry slate is far lower than that of walking on slippery ice. Therefore, the physical characteristic cost is considered in this essay, and its calculation formula is

$$q_{phy}(\boldsymbol{\theta}_i) = k_{phy}\left(\sum_{j=1}^{D_s(\boldsymbol{\theta}_i)} k_{j,\text{soft}} + k_{j,\text{friction}}\right), \tag{7}$$

where $k_{i,\text{soft}}$ and $k_{i,\text{friction}}$ respectively denote the softness and friction degree of terrains, and their calculation method can be found in our previous work [36].

4) Path Smoothing Cost: To ensure that the smoothness of the initial path is not damaged during the optimization process, a smoothness cost [34] is introduced, i.e.,

$$Q_{sm}(\boldsymbol{\theta}) = k_{sm}\boldsymbol{\theta}^T\boldsymbol{R}\boldsymbol{\theta}, \tag{8}$$

where $\boldsymbol{R}$ is a positive semi-definite matrix; $Q_{sm}(\boldsymbol{\theta})$ denotes the sum of squares of accelerations along the trajectory. This term is directly added in Eq. (4).

5) Path Distance Cost: The application experience has shown that ignoring the distance cost will cause the characteristics of the generated path to deteriorate, and the path will be excessively twisted and bent when facing complex environments. Therefore, the cost of the shortest possible path is introduced here, and its calculation method is

$$q_{sp}(\boldsymbol{\theta}_i) = k_{sp}\left[\left(\sum_{j=1}^{N-1} \|\boldsymbol{\theta}_{j+1} - \boldsymbol{\theta}_j\|\right) - \|\boldsymbol{\theta}_N - \boldsymbol{\theta}_1\|\right]. \tag{9}$$

After optimization, a path suitable for legged robots is obtained, which comprehensively considers the characteristics of smoothness, obstacle avoidance, and physical and geometrical characteristics of terrains.

D. Path Scoring Mechanism via Contact Planning

In the path generation process mentioned above, although many environmental constraints are considered, the path generated still cannot guarantee that it is trackable for legged robots. This is because legged robots are different from wheeled and unmanned aerial vehicles. They are expected to work in an environment with discontinuous grounds and need to consider whether they can follow a given path by planning a contact state sequence. From the perspective of human-machine collaboration, humans are not good at solving combinatorial optimization problems for legged robots. For example, finding the optimal combination of foothold, gait, and pose is extremely brain-taxing. Therefore, the complex discrete contact state planning is left apart to the machine intelligence here, and the operator only needs to choose the path with better quality. This section will introduce a path evaluation method named FastMCTS to solve the problem.

1) FastMCTS Method: Monte Carlo Tree Search (MCTS) [10] is an algorithm that uses random (Monte Carlo) samples to find the best decision. MCTS is expert at solving problems such as discrete decision-making, such as alphaGo [37], making it suitable for solving the complex motion planning problem of legged robots. In the Monte Carlo tree, each node represents a state of the robot $\Phi :=< {}^{W}_{B}R,\ {}^{W}_{B}r\ , c_F,\ t_F,\ {}^{W}_{F}r >$, including the robot's posture, position, foothold position, support status during the transfer process, leg error status. More details of the calculation of each parameter can be found in our preprint [38].

Although the traditional MCTS method can be used directly for contact state planning, the search speed is still plodding because there are plenty of available contact states to choose for a given state, which leads to much calculation amount [38]. Therefore, the FastMCTS is introduced there, which can find feasible results in seconds (or even faster, it depends on the complexity of the terrain) and have a high passability. Nevertheless, because there is less time for exploration and optimization, the sacrifice made is that the result sequence is redundant and not fully optimized, making the robot's walking efficiency extremely low. However, this method is very suitable for judging the pros and cons of a path. Because being able to quickly judge whether a path is trackable or not is what the evaluation method needs, rather than the quality of the search sequence results.

In the simulation step of the standard MCTS method, many simulations have been performed. Nevertheless, only the simulation results are utilized, and the state sequence obtained during the simulation is discarded [38]. The FastMCTS uses simulation sequences to quickly build a master branch of the search tree and iteratively updates the master branch by the branch with the highest potential to the destination. The primary purpose of this algorithm is to construct a feasible state sequence quickly, but its optimality cannot be guaranteed. The fast Monte Carlo tree search algorithm is different from the standard MCTS framework. It consists of four main steps: Extension, Simulation, Updating Master Branch, and Backtracking.

First, take the starting state Φ_{start} of the hexapod robot as the specified starting node Φ_k.

• Extension: Expand all candidate states of the specified node Φ_k. Each candidate node can only be expanded once. Note the nodes expanded as set AS_{Φ_k}.

• Simulation: To each node $\Phi_0 \in AS_{\Phi_k}$, using the default strategy simulation (Random selection of next step gait, foothold, and posture is taken in this easy way) until reaching the termination condition. Noting the distance of the simulation as $d(\Phi_0)$. Taking the nodes of the simulation generated as set T_{Φ_0}. The simulation termination condition of this method is that the robot is continuously stuck or reaches its destination.

• Updating Master Branch: Select the extended maximum simulation distance node $\Phi_{k,f} \in AS_{\Phi_k}$.

$$\Phi_{k,f} = \underset{\Phi \in AS_{\Phi_k}}{\arg\max}(d(\Phi)). \tag{10}$$

Then add the simulation node sequence T_{Φ_0} to the search tree and update it as the new master branch.

• Backtracking: If the master branch does not reach the destination, then select the node Φ_I from leaf nodes of the search tree, which is closest to the target, toward the root node successively, and start to expand, simulate, and update the master branch.

Next, introduce the flow of the entire algorithm, according to Fig. 3. In Fig. 3(a), all the candidate state nodes are expanded according to the selected node. Then the simulation is performed with them as the starting point, and the simulation distance and state sequence are recorded. In Fig. 3(b), the node with the largest simulation distance is selected for expansion, and each node in the state sequence recorded in the simulation is added one by one. The thick solid line indicates the master branch in the figure. Figures 3(c)(d)(e) indicate that if the master branch does not reach the destination, the algorithm will gradually expand backward from the furthest child node and update the master branch. The end condition of the entire algorithm is: the tree node reaches the destination, or the program traces back to the root node.

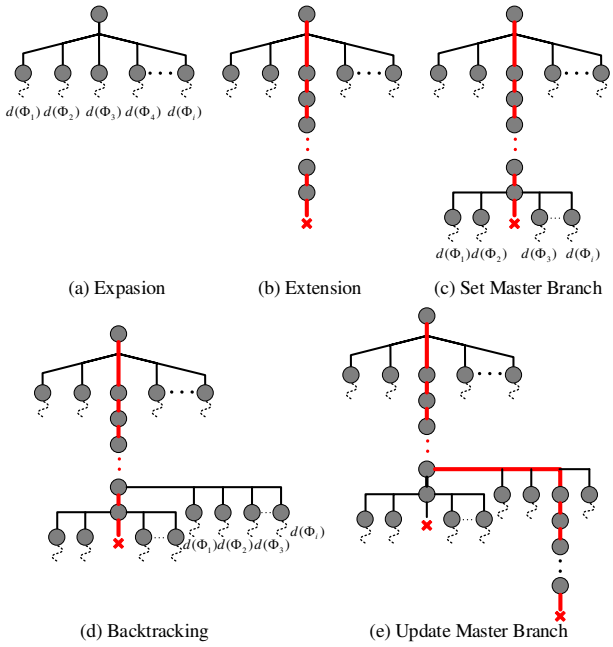

Fig. 3: Workflow of FastMCTS. (a)–(c) establish the initial search tree, and the red line represents the master branch. (d) if the furthest node does not reach the destination, the node closest to the target and not extended is selected as the new extended node. (e) if the furthest node of the newly extended branch is closer to the destination than that of the main branch, the main branch will be updated by the new one.

The algorithm uses the simulation results to establish the master branch quickly and updates the master branch by backtracking until it reaches the destination or back to the root node. The main idea is to find the position where the robot is easily trapped through multiple simulations and then keep back and try again until it finds an available solution.

2) Path Scoring Method: Once an optimized path is generated, FastMCTS can be used to evaluate its quality. Specifi-

cally, the $\boldsymbol{x}_i$ represents the ith discrete point of a tracked path $\boldsymbol{x}$ that the robot should actually follow in the Cartesian space. n is the discrete point number of the tracked path. And the location where the robot cannot move forward at j iteration judged by FastMCTS denotes as $\boldsymbol{x}_{\mathrm{fMCTS,j}}$. N_{iter} denotes the iteration times of the evaluation process. Then the evaluation path score is defined as

$$\text{score} \ = \max \frac{\sum_{i=2}^{I_j} \left\| \boldsymbol{x}_i - \boldsymbol{x}_{i-1} \right\|}{\sum_{i=2}^{n} \left\| \boldsymbol{x}_i - \boldsymbol{x}_{i-1} \right\|}, j = 1, \cdots, N_{iter} \quad (11)$$

with

$$I_j = \underset{1 \le i \le n}{\arg\min} \left(\boldsymbol{x}_i - \boldsymbol{x}_{\mathrm{fMCTS,j}} \right). \quad (12)$$

E. Robot Feedback Method

1) Image Feedback: The entire path planning process is conducted in a three-dimensional coordinate system, but a more comfortable and informative way for humans is to observe images or videos directly. Humans can obtain more environmental information through raw visual images. Therefore, the final path is projected to the image coordinate system, marking the trackable part in green and the untrackable part in red, enabling the operator to make decisions conveniently. The formula for delivering 3D points to the image coordinate system for camera models is

$$\boldsymbol{P}_{uv} = z^{-1} \boldsymbol{K} \boldsymbol{T} \begin{bmatrix} \boldsymbol{P}_w \\ 1 \end{bmatrix}, \quad (13)$$

where $\boldsymbol{P}_{uv}$ represents the coordinates in the image coordinate system, $\boldsymbol{P}_w$ is the position in the world frame, z is the depth value of the point cloud, $\boldsymbol{K}$ is the internal parameter of the camera, and $\boldsymbol{T}$ is the external parameter of the camera.

2) Touch Feedback: In addition to visual feedback, with the help of a 3D force teleoperated handle, operators can get feedback force from the robot and make decisions through feeling the touch force. For the human intervention method of target position selection, the mapping relationship between force feedback and path score is

$$\boldsymbol{F} = k_{\text{haptic}} \times (1 - \ \text{score} \ ) \times \begin{pmatrix} \cos\alpha \\ \sin\alpha \\ 0 \end{pmatrix}, \quad (14)$$

where α denotes the yaw angle of the haptic device, same as equation (1), and k_{haptic} denotes the weight coefficient. During operation, the greater the resistance felt by operators, the lower the quality of the path is considered to be. In this way, operators can clearly feel the path quality, reminding them of the dangers through tactile feedback.

3) Motion Prediction Visual Feedback: When path evaluation results are obtained, we visualize the next movement of the robot and feed it back to the operator through the simulation interface. By simulating the next motion, the operator can judge the current state of the robot and predict the next movement of the robot effectively. Specifically, the simulation system collects all state information of the robot, including position, posture, joint data, and contact forces. Based on them, simulations predict the next movement of the robot and visualize it in the interactive interface. Once the robot's state is different from what operators expected, they can find possible issues timely. Meanwhile, the visualization of the next step can also help operators to predict the future behavior of the robot clearly, facilitating their decision process.

III. Experiments

A. Experimental Setup

The proposed method is verified on ElSpider [11] (Fig. 4), an electric-driven heavy-duty hexapod robot (1.9 × 2.1 × 0.5 m, 330 kg) developed by the Harbin Institute of Technology. For environment sensing tasks, a depth camera (Intel D435i) is installed on the robot. A visual capture system is used for state estimation. The algorithm is run on a notebook computer with an i7 2.60 GHz processor. Besides, a Huawei wireless router is installed to build a local network for remote communication, and a PHANToM teleoperated handle, supporting six-DOF (Degrees of Freedom) input and three-DOF output, is adopted in our experiments. The whole system runs based on the Robot Operating System (ROS), and the operation command from operators can be remotely transmitted to the robot node through the ROS. The robot node transmits data, including the environmental map, robot state, raw images, prediction motion, and force, to the operator. For simulation, the robot and environmental maps are visualized in the RVIZ software of ROS. The structure and purpose of the experimental section are summarized in Table I.

B. Simulation Results

1) The Path Generation Results: The shared control framework is tested first in simulations. The simulated environments are randomly deployed with various types of obstacles and random footholds for walking. Three different paths are shown in Fig. 5(a)-(b): the blue one is the human-guided path, B-spline smooths the white, and the green is the final path optimized by STOMP. It can be seen that the proposed method can obtain a more secure path from a rough input path while ensuring that the path is smooth and intuitive. In terms of semi-manual input path, as shown in Fig. 5(b), the operator gives a destination on the map and a zigzag path (yellow line in the figure) generated by RRT is set as the initial path for the proposed method. Finally, the proposed method still optimizes the path and retains the shape of the initial trajectory to

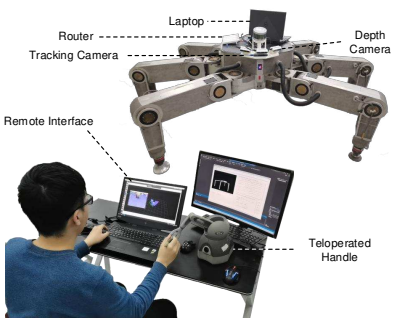

Fig. 4: Hardware equipment.

TABLE I: Experimental Setup

Experiment category		Purpose
Simulation	Path Generation Test	It is carried out to demonstrate that the proposed method can improve the safety and reliability of human-guided paths.
	Path Cost Comparison	It is carried out to demonstrate the advantage of the proposed method in reducing traversing costs.
	Path Passability Evaluation	It is carried out to verify the advantages of FastMCTS over other methods in terms of computational speed.
Hardware Experiments	Navigation Experiments	It is carried out to demonstrate the effect of the proposed method applied to robot navigation.
	Robot Feedback	It is carried out to demonstrate the role of robot feedback in our approach.
	Comparison Experiments	It is carried out to compare the proposed method with the latest shared control method and the traditional method.

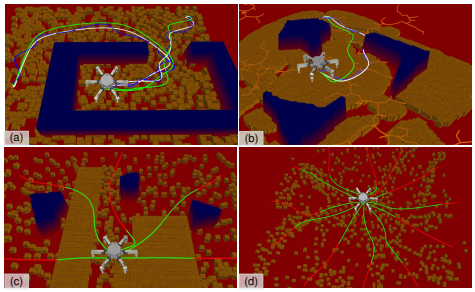

Fig. 5: Simulation results of path planning. Blue lines denote human guided path, white lines represent the smoothed result, and red and green lines are the final path evaluated by FastMCTS.

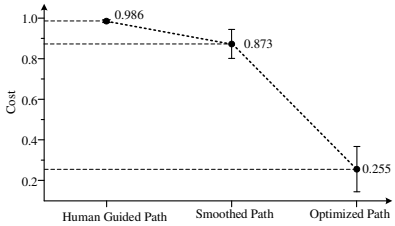

Fig. 6: The cost of different paths.

some extent. Besides, by minimizing the foothold density cost, paths tend to converge to areas with dense footholds while avoiding obvious obstacles and keeping smooth and intuitive (see Fig. 5(d)). The final paths with different human-guided paths are shown in Fig. 5(c)(d). The green part denotes the trackable area while the red represents the untrackable area, which is evaluated by FastMCTS. It can be seen that these paths tend to converge to areas with dense footholds while avoiding obvious obstacles and keeping smooth and intuitive. Through visualization, operators can evaluate the quality of paths easily, helping them to find a better way for the robot.

2) Path Cost Comparison: In order to further quantify the quality of path optimized, we collected the cost of 100 sets of paths (including human-guided path, smoothed path, and optimized path) in different simulation scenarios. The evaluated function includes three parts: collision cost, foothold density cost, and smoothing cost. Each cost is normalized by dividing the maximum cost among the three cost parts. As shown in Fig. 6, it can be seen that the cost of the initial human-guided path is much larger than the optimized path cost result, while the smoothed path only reduces its cost by about 11% compared with the optimized one. Therefore, by smoothly optimizing the path, paths with lower cost and higher security can be obtained.

3) Path Passability Evaluation: The planned continuous path does not guarantee that legged robots can track it in a sparse foothold environment. It is necessary to consider whether a corresponding sequence of contact states can be planned along the path. Therefore, we conducted several experiments to validate the proposed FastMCTS on terrain maps with three different densities of available footholds to test its passability and search speed. As shown in Fig. 7, the footholds on these maps are randomly distributed in a specific rectangular area (10m $\times$ 2m), including 100 footholds, 150 footholds, and 200 footholds. For statistical results, these two planning scenarios are performed on 20 different maps for each foothold density. The starting point of the robot is set as the coordinate origin, and the target point is (8m, 0m). When the robot moves more than 8 meters in the X-axis direction, it is considered that the robot reaches its destination. The first comparison method is another variant of the MCTS method (slidingMCTS) designed for contact planning for legged robots [11], which is proven to have a higher passability advantage and higher quality of contact sequences than RRT-based methods. The other comparison method is the standard MCTS method, and when the search time is over 20 minutes or the robot reaches the destination, the algorithm stops.

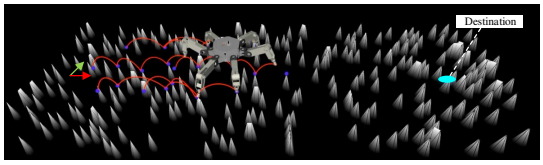

Fig. 7: Random foothold distribution map.

We test the proposed FastMCTS method and the comparison methods on 60 processed terrain maps with different distributions of available footholds, and the results of the passability are shown in Table II. It can be seen that the proposed method has a similar performance to slidingMCTS in terms of passability, but its search time is significantly reduced. Specifically, the average planning time is within 1s, while the search time of the slidingMCTS method is close to 100s and that of the standard MCTS is much more. However, the compromise is the redundancy of contact state sequences, which means more steps are required to walk for the same distance. As shown in Tab. II, the average step length of the proposed method is lower than that of the comparison methods. This is because FastMCTS focuses on finding feasible solutions rapidly without being fully optimized. Such characteristics make FastMCTS suitable for rapid assessment of the feasibility of pathways.

Based on FastMCTS, the planned navigation path is evaluated. As shown in Fig. 5(c)–(d), it can be seen intuitively that the evaluation method labeled the path using red and green colors clearly, representing the untrackable parts and trackable parts respectively. Although obvious obstacles have been avoided at the path level, conventional path planning

TABLE II: Comparison results of different contact sequence planning methods. FD denotes forward distance, representing passability of robots; ASL is average step length, representing the speed of robots; AST means average search time; Sl is slidingMCTS method; St is the standard MCTS method; Pr is the proposed method.

	FD(m)			ASL(m/step)			AST(s)		
FN	Sl	St	**Pr**	Sl	St	**Pr**	Sl	St	**Pr**
100	5.63	5.88	**5.84**	0.195	0.202	**0.059**	89.6	1120	**0.9**
150	7.42	6.71	**7.36**	0.213	0.218	**0.068**	105.3	>1200	**0.6**
200	7.81	6.12	**7.82**	0.280	0.292	**0.071**	82.7	>1200	**0.3**

Fig. 8: The hardware experimental scenes.

methods cannot judge whether a path is trackable. However, FastMCTS can make a good evaluation of the given path, and the results can give feedback to humans, helping them know more about their decision reasonability.

C. Hardware Experiments

1) Navigation Experiments: Three different challenging terrain scenarios are constructed to test the effectiveness of the proposed method, as shown in Fig. 8(a)–(c). Fig. 8(a) represents a scene where the robot needs to avoid collisions with the obstacles (towering boxes); Fig. 8(b) shows terrain with uneven distribution of footholds where stones represent forbidden areas (they cannot be chosen as footholds); Fig. 8(c) is another kind of terrain with random forbidden stones and a high obstacle. Figures 9(a)–(f) show the process of the robot traversing these scenes. Each figure has three subgraphs. The left subgraph shows the camera view of the legged robot, and much information including elevation maps and paths is projected on it, where the white represents the forbidden area, the red is the high obstacles, and the blue area means safe areas. For the paths, the blue line is the human-guided path, and the green line means the optimized path that the robot can track, while the red path cannot be tracked by the robot. The middle subgraph is the simulation review which shows the real-time visualization of the hardware robot and the environmental elevation maps. The right subgraph is the screenshot of the legged robot.

In these experiments, the operator was required to control the robot to traverse the challenging terrain by observing the human-computer interaction interface only. For the first scene, the experimental process is shown in Fig. 9(a)–(c). In the beginning, the operator intends to let the robot cross the tunnel between the obstacle and the wall. However, the path available in the operator's view is judged as a low-quality one by the robot. This is because the distance between the two obstacles is smaller than the robot's size, so the area is considered as untrackable part by FastMCTS. In Fig. 9(b), due to the drift of the pose tracking module, the constructed semantic map of grounds goes wrong, and the floor is regarded as the forbidden area. Nevertheless, although

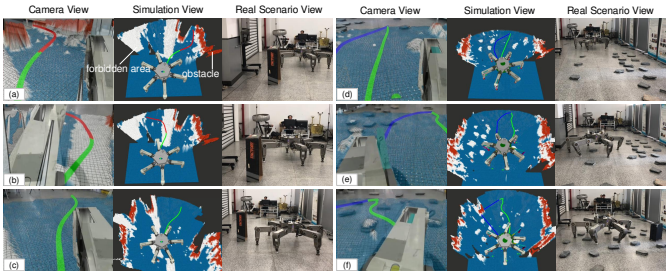

Fig. 9: Hardware Experiments results. The left subfigure is the processed feedback image from the robot, and the evaluated path and elevation map is projected in it. The white represents the forbidden area, the red is the high obstacles, and the blue area means the safe area. The blue line is the human-guided path, and the green line means the optimized path that the robot can track, while the red path cannot be tracked by the robot. The middle subgraph is the simulation view which shows the real-time visualization of the hardware robot and the environmental elevation maps. The right subgraph is the screenshot of the walking robot.

the robot algorithm evaluates the path selected by the operator as partly untrackable, the operator still chooses it because he thinks the area is safe through his analysis. Finally, the robot traversed this area by tracking the path selected by the operator (see Fig. 9(c)). It can be seen that human decision-making ability and robot intelligence cooperate with each other in the whole process. When meeting unreliable perception issues, the role of human beings is highlighted. However, humans are not good at doing elaborate planning, so the evaluation results of a massive search by the algorithm can help them make better decisions. For the second scene (Fig. 8(b)), we can see that the trajectory drawn by the operator is close to the area with sparse footholds. The path optimized by the proposed algorithm allows the robot to choose the safer area with fewer forbidden areas, and the obtained path is evaluated as an available one. Similarly, for the last scene (Fig. 8(c)), the algorithm helps humans to choose a path away from the towering obstacles, and the robot can traverse the terrain by switching gait and footholds reasonably (see Fig. 9(f)). It can be seen that the proposed method can help humans make better decisions, and the human-machine collaboration scheme is able to tackle some tricky occasions like occurring sensor failures, which cannot handle by a fully autonomous scheme. More details can be found in our demonstration videos [1].

2) Robot Feedback Results: In the hardware experiments, operators judge the quality of generated path by not only colors but also by the force exerted by the handle. In order to show the tactile feedback results clearly, a test on a specific scene is performed to collect the force exerted on operators. As shown in Fig. 10(a), a towering obstacle is placed in front of the robot. The operator selects different forward directions of the robot by controlling the handle. Figure 10(b) shows the relationship between handle force and moving angles. It can be seen that users can feel forces of different magnitudes along the changes in paths, and when the quality of the path is low, the operator can get a larger force to remind him of

[1] https://sites.google.com/view/shared-control-leggedrobot/home

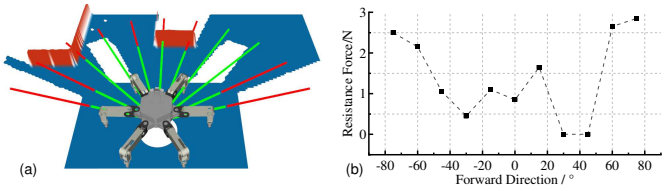

Fig. 10: (a) force feedback experiments scene. (b) the relationship between handle force and the moving direction of the robot.

the danger. On the other hand, through real-time visualization of the hexapod and motion simulation of the next step (see Fig. 9(d)–(f) in the simulation review), it is very convenient for the operator to evaluate the robot's status and predict its next behavior of the robot, which greatly facilitates the remote control of the operator. More feedback details can be found in our demonstration videos.

3) Comparison Experiments: Comparison experiments are also carried out to verify the convenience of the proposed method in terms of human-machine operation. Two comparison methods are adopted in the experiment. The first one is closest to the approach in this paper, introducing another shared-control planning framework for high-dimensional robots [27]. This approach recommends asking for user advice only when the planner determines that it is no longer significantly advancing toward the goal. In the rest of the cases, the robot plans its motion entirely by autonomous algorithms. We call it the restricted intervention method here. For the second comparison control scheme, operators need to control the hexapod robot using the operation interface shown in Fig. 11(d)(e). This interface includes various operation commands including front and back, left and right, rotation, step height setting, step length setting, gait selection, system emergency stop operation, and so on. In the experiment, five volunteers are invited to control the robot traverse different terrains (see Fig. 11(a)-(c)) using two control systems. Each volunteer learns to operate the control system first, and then they can control the robot. 30 tests are conducted in all, for each one, (1) any collision between the robot with obstacles is not allowed, (2) the robot cannot also touch the forbidden area (stones), and (3) the operators can only observe the state and surroundings of the robot by the operation interface. During the experiment, we counted the time the operator learned to use the system, the time to complete the operation task, and the success rate of the task. Finally, NASA-TLX [39] is adopted to estimate operator workload based on the questionnaire in terms of mental demand, physical demand, temporal demand, performance, effort, and frustration. The higher score of NASA-TLX means a higher workload.

The results can be found in Table III, and we can see that volunteers spent a total of 31 minutes familiarizing themselves with the traditional UI operating system, 37 minutes getting used to the restricted intervention control framework, and 14 minutes learning the proposed shared-control system. Volunteers need to adapt to the traditional UI operating system but also the autonomous system, which results in the longest learning time for the restricted intervention method. Regarding the robot navigation experiments, it took each volunteer an

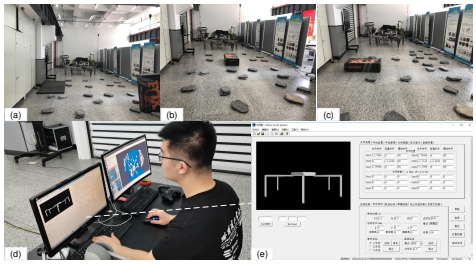

Fig. 11: (a)-(c) the comparison experimental scenes. (d)(e) the UI interface of the comparison operation scheme.

TABLE III: Comparison Results. The "Manual" means the UI operating method; "Restricted" is the restricted intervention method.

	Learning Time	Operating Time	Success Rate	NASA-TLX
Manual	31 min	21 min	60.0%	84.0
Restricted [27]	37 min	13 min	80.0%	45.33
Proposed	14 min	9 min	93.3%	18.33

average of 21 minutes to complete the task, with a success rate of only 60.0% when using the UI control method. In contrast, the volunteers took an average of 13 minutes to navigate the robot to the target, with a success rate of 80%. The robot navigation task was completed in an average of 9 minutes with a success rate of almost 93.3% using the proposed method. The only failure is caused by the robot touching a forbidden area, as the vision algorithm did not recognize the stone. It is worth noting that the operators using the traditional UI operating method sometimes control the robot by observing the hardware robot rather than using the remote interface during the process. Although this was against our rules (3), we lowered the difficulty and continued their tests. It is just an indication of the remote control difficulty of the traditional UI operating system. On the other hand, most of the operators tend to feel tired after finishing the experiment using the UI operating system, and the index of NASA-TLX is much higher than other methods. Regarding the restricted intervention method, although it reduces the operating time, it still cannot avoid using the fully manual operation, decreasing the success rate and increasing the NASA-TLX. Another disadvantage is that the human only intervenes when the autonomous algorithm gets stuck, whereas often many dangers can be predicted in advance by operators observing live images of surroundings, which is the main reason why it is more time-consuming than our method. Therefore, the results show that the proposed shared-control method can significantly reduce the training time of operators, and improve the completion degree of navigation tasks. More details can be found in our demonstration videos.

IV. Conclusions

In this paper, we have presented a closed-loop shared control framework, facilitating the operation of humans for legged robots. Rough paths provided by humans will be smoothed and optimized considering different traversing costs. The FastM-CTS is proposed to evaluate the optimized path, which can judge the trackable degree of a given path by planning contact

state sequences of legged robots. The evaluated result will be fed back through images and touch force, forming a closed loop to help the operator control the robot concisely. Simulations have been performed to investigate the advantages of our method for generating navigation paths and the advantages of FastMCTS over other methods for contact state sequence planning. The result shows that the paths generated can optimize the rough guided path as smoother and safer ones, and the FastMCTS have much higher search speeds than that of the comparison method without reducing the passability of legged robots. Finally, hardware experiments and comparison experiments have been carried out to verify the performance of our method in real scenarios. The results show that using the proposed method, operators can control the hexapod robot to execute navigation tasks more efficiently, with a higher success rate than that of the comparison control scheme. Through the feedback information (the visualized path results, motion prediction, and feedback forces), intelligent algorithms can help operators evaluate the quality of their decision behavior, while operators can also find the problem that the algorithm cannot handle. This cooperative way makes the navigation system more robust to tricky issues like perception errors.

V. Discussion and Future Work

The proposed method focuses on human-robot interaction at the upper level of the path, placing more emphasis on the robot's traversability in complex environments with sparse available footholds. At the same time, emphasizing the contribution of machine intelligence to human decision-making also contributes to the success of the proposed method. The algorithm assesses the passability of the path and passes it on to the operator, who makes a decision based on his own judgment and feedback assessment, making the human-machine interaction a closed-loop process. On the other hand, in terms of technical integrity, we have designed a human-robot interaction system that provides real-time feedback on the robot's status, the environmental map, and raw images. Before the robot performs each step, the next behavior of the robot is displayed in the interactive interface, giving the operator an expectation of the robot's movement, making the operation safer and more reliable. Compared to other methods, these strategies, therefore, reduce the operator's workload and improve the robot's ability to traverse challenging environments.

However, there are still some shortcomings in our approach. For example, the evaluation algorithm does not take into account constraints such as robot dynamics, joint moments, and terrain-bearing capacity, which makes the evaluation results potentially incorrect. However, adding more constraints must lead to more computation and longer evaluation times, which are not sufficient for real-time human-robot interaction. In addition, the robot experiments are still being validated in the laboratory. Outdoor experiments inevitably place higher demands on the human-robot interaction system, such as collecting more information about the environment, dealing with perceptual uncertainty, and resolving communication delays. In the future, we tend to conduct research on these aspects to promote more intelligent human-computer interaction solutions.

References

[1] P. Xu, L. Ding, Z. Li, H. Yang, Z. Wang, H. Gao, R. Zhou, Y. Su, Z. Deng, and Y. Huang, "Learning physical characteristics like animals for legged robots," *National Science Review*, p. nwad045, 2023.

[2] C. Wang, T. H. Weisswange, M. Krueger, and C. B. Wiebel-Herboth, "Human-vehicle cooperation on prediction-level: Enhancing automated driving with human foresight," in *2021 IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*. IEEE, 2021, pp. 25–30.

[3] C. Huang, P. Hang, Z. Hu, and C. Lv, "Collision-probability-aware human-machine cooperative planning for safe automated driving," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 9752–9763, 2021.

[4] R. Budhiraja, J. Carpentier, C. Mastalli, and N. Mansard, "Differential dynamic programming for multi-phase rigid contact dynamics," in *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2018, pp. 1–9.

[5] S. Wang, Z. Chen, J. Li, J. Wang, J. Li, and J. Zhao, "Flexible motion framework of the six wheel-legged robot: Experimental results," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 4, pp. 2246–2257, 2021.

[6] Z. Chen, J. Li, S. Wang, J. Wang, and L. Ma, "Flexible gait transition for six wheel-legged robot with unstructured terrains," *Robotics and Autonomous Systems*, vol. 150, p. 103989, 2022.

[7] S. Tonneau, A. Del Prete, J. Pettré, C. Park, D. Manocha, and N. Mansard, "An efficient acyclic contact planner for multiped robots," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 586–601, 2018.

[8] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, 2017.

[9] B. Aceituno-Cabezas, H. Dai, J. Cappelletto, J. C. Grieco, and G. Fernández-López, "A mixed-integer convex optimization framework for robust multilegged robot locomotion planning over challenging terrain," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 4467–4472.

[10] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[11] P. Xu, L. Ding, Z. Wang, H. Gao, R. Zhou, Z. Gong, and G. Liu, "Contact sequence planning for hexapod robots in sparse foothold environment based on monte-carlo tree," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 826–833, 2021.

[12] Y. Rabhi, M. Mrabet, and F. Fnaiech, "Intelligent control wheelchair using a new visual joystick," *Journal of Healthcare Engineering*, vol. 2018, 2018.

[13] J. H. Choi, Y. Chung, and S. Oh, "Motion control of joystick interfaced electric wheelchair for improvement of safety and riding comfort," *Mechatronics*, vol. 59, pp. 104–114, 2019.

[14] J. Li, B. You, L. Ding, X. Yu, W. Li, T. Zhang, and H. Gao, "Dual-master/single-slave haptic teleoperation system for semiautonomous bilateral control of hexapod robot subject to deformable rough terrain," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 4, pp. 2435–2449, 2021.

[15] M. Kurisu, "A study on teleoperation system for a hexapod robot—development of a prototype platform," in *2011 IEEE International Conference on Mechatronics and Automation*. IEEE, 2011, pp. 135–141.

[16] B. Stoddard, A. Fallatah, and H. Knight, "A web-based user interface for hri studies on multi-robot furniture arrangement," in *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, 2021, pp. 680–681.

[17] D. Monakhov, J. Latokartano, M. Lanz, R. Pieters, and J.-K. Kämäräinen, "Mobile and adaptive user interface for human robot collaboration in assembly tasks," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 812–817.

[18] R. Chipalkatty, H. Daepp, M. Egerstedt, and W. Book, "Human-in-the-loop: Mpc for shared control of a quadruped rescue robot," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 4556–4561.

[19] B. Gromov, L. M. Gambardella, and G. A. Di Caro, "Wearable multi-modal interface for human multi-robot interaction," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 240–245.

[20] X. Pan, T. Zhang, B. Ichter, A. Faust, J. Tan, and S. Ha, "Zero-shot imitation learning from demonstrations for legged robot visual navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 679–685.

[21] J. Luo, Z. Lin, Y. Li, and C. Yang, "A teleoperation framework for mobile robots based on shared control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 377–384, 2019.

[22] S. Gholami, V. R. Garate, E. De Momi, and A. Ajoudani, "A probabilistic shared-control framework for mobile robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 11 473–11 480.

[23] M. Marcano, S. Díaz, J. Pérez, and E. Irigoyen, "A review of shared control for automated vehicles: Theory and applications," *IEEE Transactions on Human-Machine Systems*, vol. 50, no. 6, pp. 475–491, 2020.

[24] K.-H. Lee, U. Mehmood, and J.-H. Ryu, "Development of the human interactive autonomy for the shared teleoperation of mobile robots," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 1524–1529.

[25] M. Fennel, A. Zea, and U. D. Hanebeck, "Haptic-guided path generation for remote car-like vehicles," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4087–4094, 2021.

[26] F. Gao, L. Wang, B. Zhou, X. Zhou, J. Pan, and S. Shen, "Teach-repeat-replan: A complete and robust system for aggressive flight in complex environments," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1526–1545, 2020.

[27] F. Islam, O. Salzman, and M. Likhachev, "Online, interactive user guidance for high-dimensional, constrained motion planning," in *IJCAI'18: Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 4921–4928.

[28] M. DeDonato, V. Dimitrov, R. Du, R. Giovacchini, K. Knoedler, X. Long, F. Polido, M. A. Gennert, T. Padır, S. Feng, *et al.*, "Human-in-the-loop control of a humanoid robot for disaster response: a report from the darpa robotics challenge trials," *Journal of Field Robotics*, vol. 32, no. 2, pp. 275–292, 2015.

[29] M. Chandarana, E. L. Meszaros, A. Trujillo, and B. Danette Allen, "Natural language based multimodal interface for uav mission planning," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 61, no. 1. SAGE Publications Sage CA: Los Angeles, CA, 2017, pp. 68–72.

[30] G. Cicirelli, C. Attolico, C. Guaragnella, and T. D'Orazio, "A kinect-based gesture recognition approach for a natural human robot interface," *International Journal of Advanced Robotic Systems*, vol. 12, no. 3, p. 22, 2015.

[31] G. Du and P. Zhang, "Markerless human–robot interface for dual robot manipulators using kinect sensor," *Robotics and Computer-Integrated Manufacturing*, vol. 30, no. 2, pp. 150–159, 2014.

[32] S. N. Flesher, J. E. Downey, J. M. Weiss, C. L. Hughes, A. J. Herrera, E. C. Tyler-Kabara, M. L. Boninger, J. L. Collinger, and R. A. Gaunt, "A brain-computer interface that evokes tactile sensations improves robotic arm control," *Science*, vol. 372, no. 6544, pp. 831–836, 2021.

[33] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[34] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[35] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed distance fields: A natural representation for both mapping and planning," in *RSS 2016 Workshop: Geometry and Beyond-Representations, Physics, and Scene Understanding for Robotics*. University of Michigan, 2016.

[36] P. Xu, L. Ding, H. Gao, R. Zhou, L. Nan, and D. Zongquan, "Environmental characterization and path planning for legged robots considering foot-terrain interaction," *JOURNAL OF MECHANICAL ENGINEERING*, vol. 56, no. 23, pp. 21–33, 2021.

[37] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[38] L. Ding, P. Xu, H. Gao, Z. Wang, R. Zhou, Z. Gong, and G. Liu, "Fault tolerant free gait and footstep planning for hexapod robot based on monte-carlo tree," *arXiv preprint arXiv:2006.07550*, 2020.

[39] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load ... s of empirical and theoretical research," in *Advances in ... Elsevier, 1988, vol. 52, pp. 139–183.*

Peng Xu received the Master degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2020.

He is currently a Ph.D. student at the State Key Laboratory of Robotics and Systems, Harbin Institute of Technology. His current research interests include motion planning, unsupervised perception, and optimal control of legged robots.

Zhikai Wang received the Master degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2022.

He is currently an engineer in Huawei.

Liang Ding (Senior Member, IEEE) was born in 1980. He received the Ph.D. degree in mechanical engineering from the Harbin Institute of Technology, Harbin, China, in 2009. He is currently a Professor with the State Key Laboratory of Robotics and System, Harbin Institute of Technology. His current research interests include mechanics (in particular terramechanics), intelligent control, design and simulation of robotic systems, in particular for planetary exploration rovers and multi-legged robots.

Zhengyang Li received the bachelor degree in mechanical engineering from Harbin Institute of Technology, China, in 2021.

He is currently a master student at the State Key Laboratory of Robotics and Systems, Harbin Institute of Technology. His current research interests include motion planning, mapping and legged robots.

Junyi Shi received the Master degree in mechanical engineering from Harbin Institute of Technology, China, in 2020. He is currently a doctoral student within the Department of Electrical Engineering and Automation, Aalto University. His research topics include perception, mapping and scene understanding for mobile robots.

Haibo Gao was born in 1970. He received the Ph.D. degree in mechanical design and theory from the Harbin Institute of Technology, Harbin, China, in 2004. He is currently a Professor with the State Key Laboratory of Robotics and System, Harbin Institute of Technology. His current research interests include specialized and aerospace robotics and mechanism.

Guangjun Liu (Senior Member, IEEE) received the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 1996. He is currently a Professor and a Former Canada Research Chair in control systems and robotics with the Department of Aerospace Engineering, Ryerson University, Toronto. His 5 current research interests include control systems and robotics, particularly in modular and reconfigurable robots, mobile manipulators, and aircraft systems.

Yanlong Huang received the Ph.D. degree in robotics from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2013. After that, he carried out his research as a Postdoctoral Researcher with the Max-Planck Institute for Intelligent Systems and Italian Institute of Technology.

He is currently a University Academic Fellow with the School of Computing, University of Leeds, Leeds, U.K. His interests include imitation learning, optimal control, reinforcement learning, motion planning and their applications to robotic systems.