Biomimetic Intelligence and Robotics 5 (2025) 100219

Contents lists available at ScienceDirect

Biomimetic Intelligence and Robotics

journal homepage: www.elsevier.com/locate/birob

Research Article

An improved path planning and tracking control method for planetary exploration rovers with traversable tolerance

Haojie Zhang, Feng Jiang, Qing Li *

School of Automation and Electrical Engineering, University of Science and Technology Beijing, Beijing 100083, China

ARTICLE INFO

Article history:
Received 12 November 2024
Revised 21 January 2025
Accepted 23 January 2025
Available online 15 February 2025

Keywords:
Planetary exploration rovers
Path planning
Motion primitives
Optional arc paths

ABSTRACT

In order to ensure the safety and efficiency of planetary exploration rovers, path planning and tracking control of a planetary rover are expected to consider factors such as complex 3D terrain features, the motion constraints of the rover, traversability, etc. An improved path planning and tracking control method is proposed for planetary exploration rovers on rough terrain in this paper. Firstly, the kinematic model of the planetary rover is established. A 3D motion primitives library adapted to various terrains and the rover's orientations is generated. The state expansion process and heuristic function of the A* algorithm are improved using the motion primitives and terrain features. Global path is generated by improved A*-based algorithm that satisfies the planetary rover's kinematic constraints and the 3D terrain restrictions. Subsequently, an optional arc path set is designed based on the traversable capabilities of the planetary rover. Each arc path corresponds to a specific motion that determines the linear and angular velocities of the planetary rover. The optimal path is selected through the multi-objective evaluation function. The planetary rover is driven to accurately track the global path by sending optimal commands that corresponds to the optimal path for real-time obstacle avoidance. Finally, the path planning and tracking control method is effectively validated during a given mission through two simulation tests. The experiment results show that the improved A*-based algorithm reduces planning time by 30.05% and generates smoother paths than the classic A* algorithm. The multi-objective arc-based method improves the rover's motion efficiency, ensuring safer and quicker mission completion along the global path.

1. Introduction

Path planning is crucial for planetary exploration rovers to move safely and efficiently to the mission goal, which is composed of global path planning and local path planning. A traversable map with low resolution is usually generated for global path planning using orbital maps and environmental data. Global path planning typically occurs at the ground control center and produces a rough long-distance path. The generated global path is sent to the rover in the form of target and path points to guide it to the mission goal [1]. The perception module of the rover aims to generate a high-resolution traversable map required for local path planning by detecting the environment. The rover is controlled by local path planning to travel from its current position to the target without collisions while still following the global path. The perception module of the rover generates a high-resolution traversability map required for local path planning by detecting the environment. This planning process is based on information about the rover's position, attitude, goals, and obstacles. Local path planning is performed onboard in real time. Due to the distinctive conditions of extraterrestrial planets, the safety of planetary rovers and the effect of terrain features are expected to be taken into account more in path planning than that in terrestrial vehicles. Considerable research has been conducted in recent years to investigate techniques for path planning and control of planetary rovers.

Graph-search and sampling-based methods have become two popular techniques for solving path planning problems in mobile robotics [2]. However, limited by the challenging terrains, the locomotion control methods of different configurable robots vary greatly [3]. Global path planning methods for planetary rovers can generally be categorized into two groups: cost-based methods and learning-based methods. Raw perception data is converted into grid maps, Voronoi diagrams, and other forms of graph representation. A variety of cost-based algorithms are then used to find the optimal path from the start point to the goal point, such as graph search algorithms like A* [4] and D* [5,6], bio-inspired algorithms [7], sampling-based algorithms [8], etc. These methods have been enhanced to better suit the requirements of path planning on extraterrestrial planets. For instance, in order to improve the efficiency of planetary rovers in performing tasks in

* Corresponding author.
E-mail address: liqing@ies.ustb.edu.cn (Q. Li).

https://doi.org/10.1016/j.birob.2025.100219

dynamic and rough terrain environments, Raja et al. integrate the potential field method with the A* algorithm to develop a global path planning method for planetary rovers [9]. Terrain roughness, path length and curvature are used as components of the cost function. This method enhances the flexibility and robustness of the planning in dynamic rough terrain. The terrain features and energy constraints are introduced to the cost function of the A* algorithm, which improve path safety while reducing energy consumption of planetary rovers [10]. Sutoh et al. analyze the impact of weight changes on the distance, terrain, and lighting costs in planetary rover path planning, aiming to improve the planning results [11]. In order to reduce the slippage of planetary rovers, the slippage is predicted based on terrain slope and the genetic algorithm is improved to generate an optimal path without slippage [12]. However, the motion constraints of the planetary rover in relation to the smoothness of the path are typically not taken into account in these methods. This shortcoming can make it difficult for the planetary rover to accurately follow the global path.

Raw orbital data or generated maps are used as input for learning-based methods to generate paths end-to-end. To avoid environment mapping and reduce the dependence of path planning on human's prior knowledge, Zhang et al. propose a deep learning-based global path planning method for planetary rovers, which is able to plan paths directly using orbital images [13]. Tanaka et al. use a reinforcement learning-based algorithm to plan more flexible global paths, where the environmental terrain and the electricity of the rover are used as constraints [14]. Learning-based methods are more flexible and simple, but they require a large amount of training data and lack interpretability.

Local path planning algorithms have been integrated into the planetary rovers launched by the United States and China. Spirit and Opportunity are the earliest planetary rovers which achieve local path planning capabilities. The local path planning algorithm used is integrated into the GESTALT (Grid-Based Estimation of Surface Traversability) system [15]. A digital elevation model(DEM) is used to compute the terrain features. The traversability and target distances are used as metrics to rank a set of optional arc paths, where the optimal path is chosen. The improvements based on this method have been widely applied to subsequent planetary rovers [16–19]. Unfortunately, larger or denser obstacles around the planetary rover may cause the local path planning failure since the lack of global path as a guide. As a result, the planetary rover will get stuck.

In addition, some of the algorithms that have been used for terrestrial vehicles are improved as local path planning methods for planetary rovers, such as the A*, the RRT* [20], the FMM [21], and machine learning [22], etc. Wu et al. introduce terrain slope and roughness into the heuristic function to improve the A* algorithm, which makes it more suitable for navigating terrain features on extraterrestrial planetary surfaces [23]. In order to avoid information loss during environmental mapping, the RRT* has been enhanced to accommodate rough terrain and enable direct utilization of point cloud data for path planning. Garrido et al. improve the FMM by constructing an external vector field based on slippage. This enhancement allows for considering planetary rover slippage in path planning [24]. Yu et al. use a deep reinforcement learning approach to decrease slippage of planetary rovers and improve path safety [25]. Environmental information and the rover's position relative to the target are used as inputs, and the optimal action is selected from a series of predefined rover actions as the output. Machine learning has also been used to classify terrain to determine terrain traversability and reduce slippage of planetary rovers [26,27]. These methods also risk causing planetary rovers to get stuck due to the absence of global path as a guide. Only the optimal path within the current field of view of the planetary rover is planned, but environmental constraints are not considered.

In order to achieve a comprehensive planetary exploration mission, several planetary planning frameworks have been proposed. In these frameworks, global path planning is first used to generate a roughly safe path. Then, local path planning is used to guide the planetary vehicle to follow the global path safely. JR et al. propose a planning framework for planetary rovers based on the FMM algorithm [28]. Conventional FMM is first used to generate global paths based on orbital images. When the planetary rover encounters an obstacle that blocks the global path, a heuristic version of FMM is used to repair the global path based on the rover's perception information. Wang et al. use the A* algorithm to create a simple global auxiliary path [29]. Subsequently, DWA (Dynamic Windows Approach) is used to plan local paths under the guidance of the global path. With this framework, a planetary rover can be controlled to safely travel to the global goal. However, the global path planning methods employed in existing planning frameworks are typically simple and only generate a rough global path. The planning process ignores the smoothness of paths and the kinematic constraints of planetary rovers. This limitation leads to more difficulty in tracking the global path and may make the planetary rover encounter difficulties navigating complex terrain. Complex local path planning methods increase resource requirements. This makes it difficult to actually use them for resource-poor planetary rovers.

To address above mentioned problems, an improved path planning and tracking control method is proposed for planetary exploration rovers in this paper. An improved A*-based global path planning method is given, which incorporates the motion constraints of planetary rovers as motion primitives and takes the 3D terrain features and path smoothness into account. Thus, the method can be used to strategically generate optimal global paths. Furthermore, a multi-objective arc-based local motion planning method is presented. This method is characterized by its computational simplicity, real-time processing capabilities, and security features. It ensures real-time obstacle avoidance while accurately adhering to the global path. As a result, it is guaranteed that the planetary rover will travel to the mission goal more safely and efficiently after using the proposed path planning and control method.

2. The overview of the improved path planning and tracking control method

The overall diagram of the improved path planning and tracking control method is shown in Fig. 1. A DEM of the planetary surface and RGB-D images captured by the depth camera mounted on the planetary exploration rover are served as inputs to this method. The optimal motion commands are generated to guide the rover to the mission goal.

Initially, 3D motion primitives of a planetary rover are generated offline based on the planetary rover's kinematic model. Motion Primitives are predefined basic trajectory units that represent feasible paths from one state to the next state [30]. In this paper, motion primitives are represented as third-order polynomial spirals, which serve as the basic units for state expansion in the global path planning process. Each point on the primitive represents the rover's position and heading. Subsequently, DEMs are analyzed to calculate terrain features. The motion primitives and terrain features are used to improve the A* algorithm. The cost between states during the A* search is refined by elevating the cost of motion primitives and terrain features, such as slope, roughness, etc. The state expansion of the A* algorithm is enhanced by incorporating motion primitives for the path search. The generated global path will be more smoothness and safety

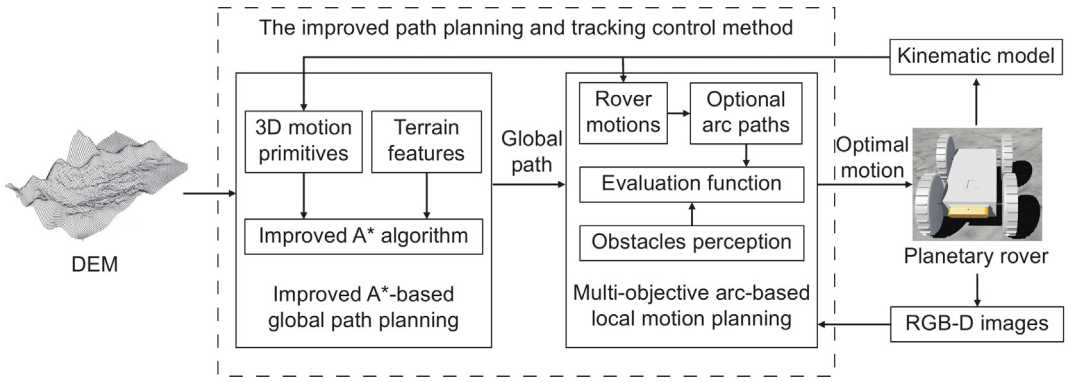

Fig. 1. The diagram of the improved path planning and tracking control method.

which satisfies the kinematic constraints of the planetary rover and the 3D terrain restrictions.

After the global path is obtained, a series of planetary rover motions will be sampled to generate the optional arc paths which are derived from the planetary rover's kinematic model. These motions are represented as a pair of linear and angular velocities. Additionally, these optional arc paths are generated offline. This sampling process simulates the trajectory of the planetary rover at a predefined linear and angular velocity for a specified duration. The optimal arc path is selected based on the multi-objective evaluation function. These optimized objectives include matching the arc path to the global path, reducing the distance between the arc path and the local goal, and causing the arc path to avoid obstacles. Local goals are selected through sampling points on the global path. An enhanced color segmentation algorithm is employed to localize obstacles through the RGB-D image. Finally, the optimal motion commands for the planetary rover that corresponds to the optimal arc path is generated as output commands. The planetary rover will be driven by the optimal motion to follow the selected arc path which results in collision-free, accurate, and efficient tracking of the global path to the mission goal.

3. Improved A*-based global path planning with motion primitives

In this paper, an improved A*-based global path planning method with motion primitives is proposed for generating a global path on the DEMs. The A* algorithm is a heuristic search algorithm and can be effectively enhanced to accommodate various applications by designing and adjusting the heuristic value function. The smallest feasible trajectory units for planetary rovers are represented by motion primitives, which can be utilized by A* algorithms for state expansion. The cost between states during the A* search is determined by the cost of the motion primitives and 3D terrain features, producing a global path that adheres to the rover's motion constraints and terrain limitations.

The algorithm is presented as Algorithm 1. Given the start state s_0, goal state s_g, DEM, and motion primitives library, the global path P is computed. Initially, the current node s is set to s_0, and $Parent(s)$ is set to null (Algorithm 1, Line 1). As long as s has not reached s_g, the algorithm iteratively performs the following operations: s is expanded using the motion primitives library to generate a set of neighboring nodes $N(s)$, and each neighboring node s' is traversed (Algorithm 1, Lines 3–4). For each s', the transition cost $C(s, s')$ from s to s' is calculated based on the DEM and motion primitives. The actual cost $g(s')$ is updated, the heuristic cost $h(s')$ from s' to the goal state s_g is calculated, and the total cost $f(s')$ is determined (Algorithm 1, Lines 5–8). If s' is a new node or $f(s')$ is improved, the $Parent(s')$ is updated to s (lines 9–10).

After traversing all s', the one with the smallest $f(s')$ is selected as the new s (Algorithm 1, Line 13). If s is equal to s_g, the global path P is generated by backtracking through the parent nodes (Algorithm 1, Lines 14–15) and returned as the output. Otherwise, the loop continues until a path is found or no more nodes can be expanded.

Algorithm 1 Improved A*-based Global Path Planning

Input: Start state s_0, Goal state s_g, Motion Primitives Library, DEM
Output: Global path P

1: Initialize current state $s \leftarrow s_0$, $Parent(s) \leftarrow \emptyset$
2: **while** $s \neq s_g$ **do**
3: $N(s) \leftarrow$ Expand state s using motion primitives
4: **for all** $s' \in N(s)$ **do**
5: $C(s, s') \leftarrow$ Compute transition cost using DEM and motion primitive from s to s'
6: $g(s') \leftarrow g(s) + C(s, s')$
7: $h(s') \leftarrow$ Compute heuristic cost using DEM and Dubins curve
8: $f(s') \leftarrow g(s') + h(s')$
9: **if** s' is a new state or $f(s')$ improves **then**
10: $Parent(s') \leftarrow s$
11: **end if**
12: **end for**
13: $s \leftarrow \arg\min_{s' \in N(s)} f(s')$
14: **if** $s = s_g$ **then**
15: $P \leftarrow$ Backtrack from s_g to s_0
16: **end if**
17: **end while**
18: **return** P

3.1. Motion primitives generator

The generation of motion primitives for planetary rovers involves designing a set of control inputs based on the rover's initial and target states, ensuring it follows the intended trajectory. Neglecting the rover's kinematic constraints may result in hazardous situations, such as collisions or rollovers. By considering the initial and goal states as boundary conditions and incorporating the rover's kinematic constraints, motion primitive generation can be formulated as a two-point boundary value problem (TP-BVP), which is solved using a nonlinearization method.

To convert the TP-BVP into a nonlinear problem with kinematic constraints, a parametric path generation model is required. The polynomial spiral, defined by the relationship between curvature and arc length, integrates kinematic constraints directly into curve generation, ensuring adherence to position,

heading, and curvature requirements. Compared to other curve parameterization methods, the polynomial spiral offers better kinematic constraint embedding and stability under perturbations [31]. The third-order polynomial spiral, with a small parameter space, satisfies the required degrees of freedom and boundary constraints. Consequently, assuming s denotes the arc length, the control input is represented as a third-order polynomial spiral

$$\kappa(s) = \kappa_0 + bs + cs^2 + ds^3 \tag{1}$$

where κ_0, b, c, and d represent the coefficients of the third-order polynomial spiral.

The generation of motion primitives involves solving the parameter vector of the spiral $\kappa(s)$. The objective function J is designed to minimize bending energy and trajectory length to create a smoother and shorter path. Hence, the problem of generating motion primitives is formulated as an optimization problem. Let $\mathbf{x}_0$ and $\mathbf{x}_f$ represent the rover's initial and final states. The problem is formulated as

$$\begin{aligned} &\text{given} \quad \mathbf{x}_0 = (x_0, y_0, \theta_0, \kappa_0) \\ &\qquad\quad \mathbf{x}_f = (x_f, y_f, \theta_f, \kappa_f) \\ &\text{minimize } J = w_1(s_f - s_0) + w_2 \int_{s_0}^{s_f} \|\kappa(\mathbf{p})\|^2 \, \mathrm{d}s \\ &\quad \text{s.t. } \mathbf{x}(\mathbf{p}) = \mathbf{x}_0, \text{ for } s = s_0 \\ &\qquad \mathbf{x}(\mathbf{p}) = \mathbf{x}_f, \text{ for } s = s_f \\ &\qquad s_0 \le s_f \le s_{\max} \end{aligned} \tag{2}$$

where x and y denote the rover's position, while θ represents its orientation. s_0 and s_f represent the initial and final arc length, respectively, which correspond to the start and end points of the trajectory. The variable $\mathbf{p} = (b, c, d, s_0, s_f)$ represents the optimized parameter vector, while w_1 and w_2 denote the weights of the optimization indicators, which satisfy $w_1 + w_2 = 1$.

The continuous problem is discretized to simplify the solution. At each individual point, a nonlinear equation is created. Assuming that x_k and u_k represent the difference state and the curve derivative at s_k, respectively, while N denotes the total number of discrete points, the problem is reformulated as

$$\begin{aligned} &\underset{x_0, u_0, \ldots, x_N, u_N}{\text{minimize}} \; J = w_1(s_f - s_0) + w_2 \sum_{k=0}^{N-1} L(x_k, u_k) \\ &\quad \text{s.t.} \quad x_{k+1} - f(x_k, u_k) = 0, k = 0, \ldots, N-1, \\ &\qquad\quad h(x_k, u_k) \le 0, k = 0, \ldots, N-1, \\ &\qquad\quad r(x_0, x_N) \;\; = 0 \end{aligned} \tag{3}$$

where f represents the system's state difference equation, while h and r denote the inequality and equality path constraint, respectively.

The direct multiple shooting method is utilized to address this problem due to its ability to handle a larger number of decision variables and equality constraints and improve the speed of convergence.

A three-dimensional motion primitive library is generated in light of the irregular terrain where the planetary rover operates. The planetary surface is characterized by continuous terrain in the horizontal (x and y axes) dimensions, but exhibits discontinuities in the vertical (z axis) dimension. Therefore, the motion primitives library is generated using the rover's state (x, y, z, θ). Assuming that the planetary rover is located at the position (0, 0) with the heading angle ranging from 0 to 360°, and the maximum climbing angle ϕ_m of the planetary rover is discretized to generate motion primitives under various constraints.

Assuming the rover is on a flat surface with a zero slope, the initial state is denoted as (0, 0, 0, 0). The target state $(x_i, y_i, z_i, \theta_i)$ is selected from discrete heading points on a circle centered at the rover's initial position with a radius R. Thus, $x_i = R\cos\theta_i$, $y_i = R\sin\theta_i$, $z_i = 0$, $\theta_i = i\frac{2\pi}{k}$. Here, k denotes the index of the discrete heading angle, $i = 0, 1, \ldots, k$.

Fig. 2. The 2D motion primitives generated from the initial state (0, 0, 0, 0).

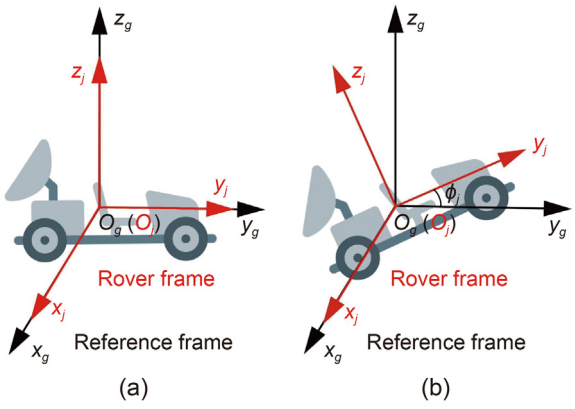

Fig. 3. (a) Reference and rover frames when slope = 0. (b) Reference and rover frames when slope = ϕ_j.

Given the same initial state (0, 0, 0, 0) and different target states $(x_i, y_i, z_i, \theta_i)$, all motion primitives are derived for the rover under terrain slope 0 by Eq. (2), as shown in Fig. 2. In this paper, no reverse motion primitives are generated.

The climbing angle for planetary rovers is discretized within the range of 0 to ϕ. Assuming the reference frame is denoted as $x_g y_g z_g$. It coincides with the rover frame located on a flat surface. As shown in Fig. 3, when the slope is ϕ_j, the rover frame $x_j y_j z_j$ can be perceived as being rotated by ϕ_j around the x_g-axis of $x_g y_g z_g$.

For any state $(x_0, y_0, z_0, \theta_0)$ on the motion primitive generated under terrain slope 0, the corresponding state $(x_j, y_j, z_j, \theta_j)$ under the terrain slope ϕ_j can be derived through transformation matrix T as

$$\begin{cases} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = T \begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_j & -\sin\phi_j \\ 0 & \sin\phi_j & \cos\phi_j \end{bmatrix} \begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} \\ \theta_0 = \theta_j \end{cases} \tag{4}$$

According to Eq. (4), the motion primitives under a slope of ϕ_j are generated. Therefore, the motion primitives for any discretized slope can be generated, and the 3D motion primitive library is constructed by generating motion primitives for different headings under all discretized slopes, as shown in Fig. 4. The 3D motion primitives are stored offline and can be quickly indexed for any terrain slope.

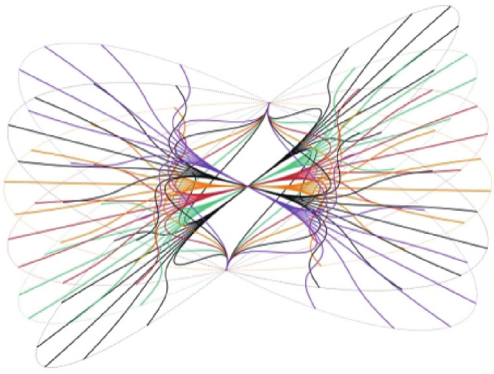

Fig. 4. 3D motion primitives generated by discretizing the slope from −30° to 30°.

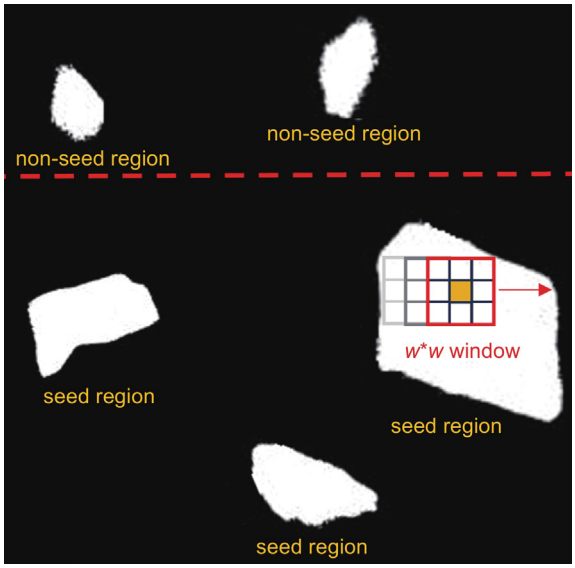

Fig. 5. Identification and traversal of seed regions.

3.2. Improved A*-based algorithm

The motion primitives define the connections between nodes in the A* search. During the search, the cost of neighboring nodes is calculated, and the node with the minimum cost is selected. The search starts from the start state and continues until the goal state is reached, at which point the optimal path is generated. The cost function for a state s is defined as

$$f(s) = g(s) + h(s) \tag{5}$$

where $g(s)$ denotes the actual cost from the start state to the current state s, while $h(s)$ is the heuristic function estimating the cost from s to the goal state.

Since the planetary surfaces are typical unstructured, it is necessary to consider three dimension terrain features in the path planning, which can be extracted from DEMs. The total cost function consists of the height cost C_{height}, slope cost C_{slope}, and roughness cost C_{rn}.

For each position (x_i, y_i) in the motion primitive, the corresponding height Δz_i, slope ϕ_i, and roughness r_i are critical for traversable cost evaluation. The height Δz_i represents the maximum vertical displacement between the grid and its neighbors, which can be formulated as

$$\Delta z_i = \max_{\substack{k,l\in\{-1,0,1\} \\ (k,l)\neq(0,0)}} |z(x_i, y_i) - z(x_i + k, y_i + l)| \tag{6}$$

where $z(x_i, y_i)$ denotes the height of the grid. The variables k and l take on values from the set $\{-1, 0, 1\}$, respectively.

The slope ϕ_i represents the degree of incline of the terrain in the grid, which can be calculated as

$$\begin{aligned} \phi_i &= \arctan\left(\sqrt{\Delta z_x{}^2 + \Delta z_y{}^2}\right) \cdot \frac{180^\circ}{\pi} \\ \Delta z_x &= \left(\frac{z(x_{i+1}, y_i) - z(x_{i-1}, y_i)}{2x_l}\right) \\ \Delta z_y &= \left(\frac{z(x_i, y_{i+1}) - z(x_i, y_{i-1})}{2y_l}\right) \end{aligned} \tag{7}$$

where x_l and y_l denote the size of the grid along the x axis and y axis respectively.

The roughness value r_i represents the extent of irregularity in the terrain surface, which is typically defined as the ratio of the surface area of a grid cell to its projected area on a horizontal plane [32]. It can be calculated based on ϕ_i as

$$r_i = \frac{1}{\cos(\phi_i \cdot \frac{\pi}{180^\circ})} \tag{8}$$

Subsequently, these values are used to calculated the height cost $C_{\text{height}}(\Delta z_i)$, the slope cost $C_{\text{slope}}(\phi_i)$, and the roughness cost $C_{\text{rn}}(r_i)$, expressed as

$$\begin{cases} C_{\text{height}}(\Delta z_i) = +\infty, \Delta z_i > \Delta z_{\max} \\ C_{\text{height}}(\phi_i) = k_{\text{height}} * \dfrac{\Delta z_i}{\Delta z_{\max}}, \Delta z_i \le \Delta z_{\max} \end{cases} \tag{9}$$

$$\begin{cases} C_{\text{slope}}(\phi_i) = +\infty, \phi_i > \phi_{\max} \\ C_{\text{slope}}(\phi_i) = k_{\text{slope}} * \dfrac{\phi_i}{\phi_{\max}}, \phi_i \le \phi_{\max} \end{cases} \tag{10}$$

$$\begin{cases} C_{\text{rn}}(r_i) = +\infty, \text{if } r_i > r_{\max} \\ C_{\text{rn}}(r_i) = k_{\text{rn}} * \dfrac{r_i}{r_{\max}}, \text{if } r_i \le \phi_{\max} \end{cases} \tag{11}$$

where k_{height}, k_{slope}, and k_{rn} denote the weight coefficients, while $\Delta z_{\max}$, $\phi_{\max}$, and $r_{\max}$ are the maximum height, slope, and roughness that the rover is capable of traversing. The infinite cost indicates an impassable terrain when the height, slope, or roughness exceeds the driving capability of the rover.

Assuming the planetary rover travels through a sequence of positions (x_i, y_i, z_i) along the motion primitive, the total height cost C_{height}, the slope cost C_{slope}, and the roughness cost C_{rn} are calculated as the sum of the costs of all positions.

The motion primitives are generated optimizing the objective function J which indicates the smoothness and length of the motion primitives. Thus, J is taken as the motion primitive cost C_J. The actual cost $g(s)$ is then expressed as

$$\begin{aligned} g(s) = g(s_{\text{parent}}) &+ w_j * C_J + w_{\text{height}} * C_{\text{height}} + w_{\text{slope}} * C_{\text{slope}} \\ &+ w_{\text{rn}} * C_{\text{rn}} \end{aligned} \tag{12}$$

where s_{parent} represents the parent node of s, while w_j, w_{height}, w_{slope}, and w_{rn} denote the weights of each cost term. This enables the concurrent consideration of the motion primitive cost and the three terrain costs during the search process.

In addition, the heuristic function $h(s)$ is reformulated as

$$h(s) = w_J * C_{J,g} + w_{ST} * C_{\text{st,g}} + w_{SL} * C_{\text{sl,g}} + w_{RN} * C_{\text{rn,g}} \tag{13}$$

where $C_{J,g}$ represents the cost of estimating the arc length from state s to the goal state, which is calculated using the Dubins curve. $C_{\text{st,g}}$, $C_{\text{sl,g}}$, and $C_{\text{rn,g}}$ represent the cost of estimating the height value, slope, and roughness from state s to the goal state, which can be estimated based on the average height value, slope, and roughness between s and the goal state.

By improving the cost calculation method and the heuristic function of the A* algorithm, the motion primitive cost and

terrain cost can be considered in the search process. This improvement can enhance the driving safety and efficiency of the planetary rover when navigating in complex terrain environments.

The path planning aims to find an optimal path to guide the rover from its start state to the goal state. To accomplish this objective, an appropriate motion primitive is essential to be selected at each state to minimize the total cost which represents the transitions between adjacent states.

During the planning search, s_0 is initially added to the OPEN list, which contains all states that have been found but not yet expanded. In addition, the CLOSED list contains all states that have been expanded. At each iteration of the proposed A*-based algorithm, the cost of each state s_i in the OPEN list is calculated by summing up $g(s_i)$ and $h(s_i)$. The state s' with the lowest cost is removed from the OPEN list and added to the CLOSED list. Then, state s' is expanded and all its successors are added to the OPEN list based on the definition of the motion primitive library. The search terminates when the goal state is retrieved from the OPEN list. The optimal path is found by tracing back through the parent–child relationships.

4. Multi-objective arc-based local motion planning

Effective local motion planning ensures planetary exploration rovers follow the global path while adapting to environmental changes. Global path planning is performed offline and often uses low-resolution maps, making it difficult to detect small obstacles. Local motion planning algorithms allow rovers to detect hazards in real-time and respond promptly, ensuring safe arrival at the goal.

This paper proposes a multi-objective arc-based local motion planning method. An optional arc path set is designed based on the kinematic constraints of the planetary rover. The optimal arc path is selected by an evaluation function through optimizing multiple objectives. The rover motion corresponding to the chosen arc path is used as the control commands, allowing the rover to accurately track the global path while avoiding obstacles.

The algorithm is presented as Algorithm 2. The inputs consist of the global path P, the optional arc path set A, and RGB-D images, while the output is the optimal control commands (v^*, ω^*). Initially, the start and end points of the global path are set as the initial state s_0 and the goal state s_g, respectively. The current state s is initialized to s_0, and the obstacle list OBS is set to empty (Algorithm 2, Lines 1–2). While $s \neq s_g$, the algorithm iteratively executes the following operations: The local goal g_l is computed based on P and s, and the optional arc path set A is generated (Algorithm 2, Lines 3–4). Subsequently, obstacles are perceived and located using RGB-D images, and the OBS list is updated accordingly (Algorithm 2, Line 5). For each path a in A, the safety score $d(v, \omega)$, goal distance score $g(v, \omega)$, and path matching score $p(v, \omega)$ are computed. The total evaluation score $G(v, \omega)$ is calculated as a weighted sum of these scores using the weight parameters α, β, and γ (Algorithm 2, Lines 7–12).

After evaluating all paths, the one with the highest evaluation score is selected as the optimal path a^*, and the corresponding control commands (v^*, ω^*) are then extracted (Algorithm 2, Lines 12–13). The rover moves according to (v^*, ω^*), and s is updated accordingly (Algorithm 2, Line 14). The loop continues until s reaches s_g. At this point, the rover successfully achieves the mission goal, and the algorithm terminates.

Algorithm 2 Multi-objective Arc-based Local Motion Planning

Input: Global path P, RGB-D images
Output: Optimal control (v^*, ω^*)

1: Initialize start state $s_0 \leftarrow P[0]$, goal state $s_g \leftarrow P[\text{end}]$
2: Initialize rover state $s \leftarrow s_0$, obstacles list $OBS \leftarrow \emptyset$
3: **while** $s \neq s_g$ **do**
4: $g_l \leftarrow$ Obtain the local goal using P and s
5: $A \leftarrow$ Generate optional arc path set
6: $OBS \leftarrow$ Perceive obstacles using RGB-D images
7: **for all** $a \in A$ **do**
8: $d(v, \omega) \leftarrow$ Compute safety score using OBS
9: $g(v, \omega) \leftarrow$ Compute goal distance score using g_l
10: $p(v, \omega) \leftarrow$ Compute path matching score using P
11: $G(v, \omega) \leftarrow \alpha \cdot d(v, \omega) + \beta \cdot g(v, \omega) + \gamma \cdot p(v, \omega)$
12: **end for**
13: $a^* \leftarrow \arg\max_{a \in A} G(v, \omega)$
14: $(v^*, \omega^*) \leftarrow$ Extract control commands from a^*
15: $s \leftarrow$ Update rover state using (v^*, ω^*)
16: **end while**
17: **return** $\emptyset$

4.1. Obstacle perception

Real-time local motion planning is necessary for planetary rover to avoid small obstacles, such as rocks, while it is moving along the global path. Vision-based obstacle detection method using stereo images exhibits high accuracy and real-time performance which is well suited for local path planning [33]. The vision-based obstacle detection method is modified in this paper for RGB-D cameras.

An RGB image is first converted to grayscale and segmented using the mean-shift algorithm, which is a clustering algorithm based on kernel density estimation. Then, the image can be segmented into multiple regions with similar features by considering the spatial and grayscale information of the pixels. Subsequently, background regions are removed from the images, such as ground, sky, etc. Finally, a series of regions representing potential obstacles are generated after segmentation which can be used for obstacle perception and recognition.

The RGB image and the depth image are corresponding in the pixel level. Each pixel in the depth image is projected onto the RGB image, thereby assigning a corresponding depth value to each pixel in the RGB image. This process allows for further processing of the segmented RGB image based on the depth values. Assuming the maximum depth threshold is T_1, the segmented regions with average depth less that T_1 will be marked as seed regions. Other segmented regions are marked as non-seed regions. As shown in Fig. 5, the white areas represent segmented regions. The areas above the red dashed line have an average depth less than T_1, classifying them as non-seeded regions, while those below the dashed line have an average depth greater than T_1, classifying them as seeded regions.

All the points in the seed region are transformed to the world coordinate system, nominally a global frame. For a pixel $p(u, v)$ with depth z in the pixel coordinate frame, the corresponding world coordinates $P(X_w, Y_w, Z_w)$ are calculated as

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z(u-u_0)\mathrm{d}x/f \\ z(v-v_0)\mathrm{d}y/f \\ z \\ 1 \end{bmatrix} \tag{14}$$

where R and t denote the rotation matrix and translation vector from world coordinate to the camera coordinate, respectively. f represents the focal length of the camera, and (u_0, v_0) is the coordinates of the principal point in the image.

Roughness and elevation step values are calculated for seed regions. As shown in Fig. 5, a sliding $w * w$ pixel-sized window is set up to traverse seed regions. Roughness reflects the degree of terrain fluctuation within a window area. Points in the window are fitted to a plane, and the distance from each point in the window to the fitting plane is calculated based on its 3D coordinates

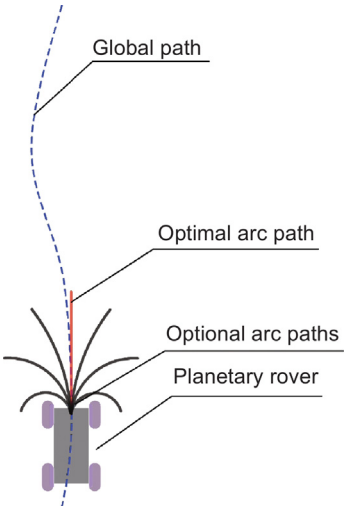

Fig. 6. Process of local motion planning.

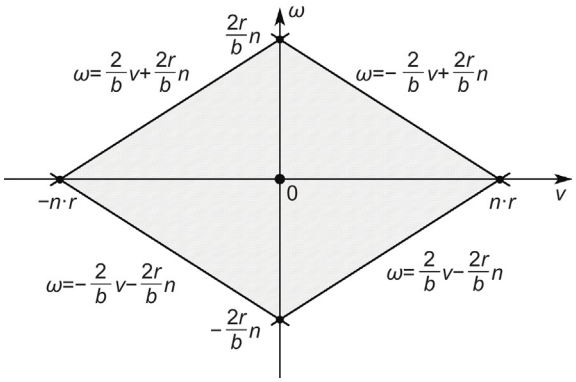

Fig. 7. The relationship between ω and v.

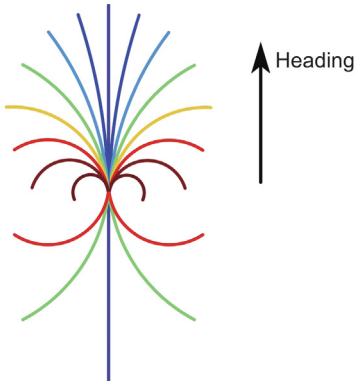

Fig. 8. Optional arc path set.

(x_i, y_i, z_i). The roughness r_i of the central pixel is computed as

$$r_i = \frac{1}{N} \sum_{i \in W} \left[\frac{Ax_i + By_i + Cz_i + D}{\sqrt{A^2 + B^2 + C^2}} \right] \tag{15}$$

where W represents the window area, N denotes the number of pixels in the current window, $[A, B, C]$ is the normal vector of the plane.

The elevation step depends on the maximum height difference within the $w * w$ pixel-sized window area. The elevation of each pixel in the window area is traversed to determine the

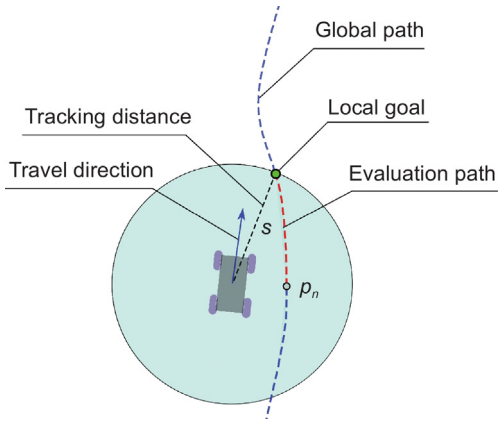

Fig. 9. Generation of the local goal and the evaluation path.

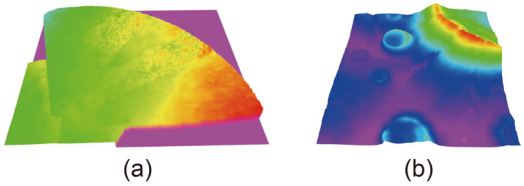

Fig. 10. Two DEMs for the global path planning test. (a) Small-scale real lunar surface data. (b) Large-scale simulated lunar surface data.

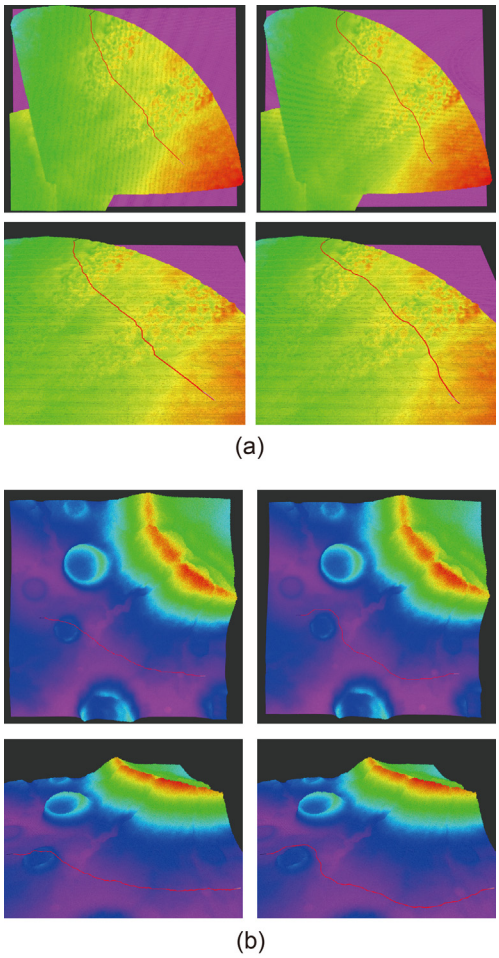

Fig. 11. Visualizations of the planning results for two algorithms. (a) Planning results on real lunar DEM. (b) Planning results on simulated DEM.

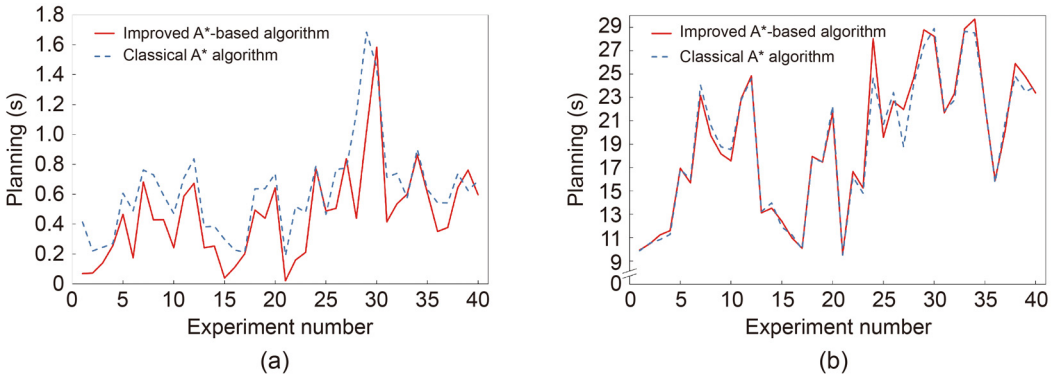

Fig. 12. Comparison of planning time and path length. (a) Comparison of planning time. (b) Comparison of path length.

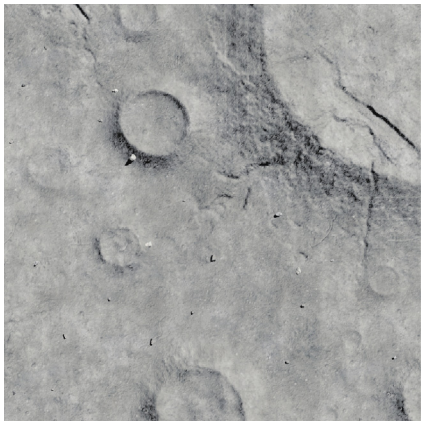

Fig. 13. Test scene based on simulated DEM and addition obstacles.

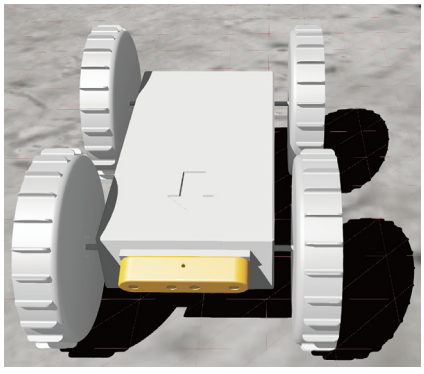

Fig. 14. The planetary exploration rover model.

maximum elevation value E_h and the minimum elevation value E_l. Consequently, the elevation step for the pixel at the center of the window is given by $E(x, y) = E_h - E_l$.

Assuming that the roughness threshold is T_2 and the elevation threshold is T_3, then the region will be marked as an obstacle when any roughness or elevation step value exceed the corresponding threshold in the seed region. The global coordinates of this region are added to the OBS list, which contains all the obstacles coordinates. For non-seed regions, the grayscale mean value is calculated. If the mean value is smaller than the predefined threshold T_4, the region is also considered as an obstacle. The global coordinates of the region are calculated using Eq. (14) and then added to the OBS list.

4.2. Sampling-based arc path generation method

The primary idea of the multi-objective arc-based local motion planning method is to select the optimal arc path from an optional arc path set based on an evaluation function. Each arc path is associated with particular linear and angular velocities of the rover. The chosen arc path is viewed as the desired trajectory for the planetary rover. Fig. 6 shows the process of local motion planning when the rover follows the global path towards the mission goal.

Let $v(t)$, $\omega(t)$ and $\theta(t)$ represent the linear velocity, angular velocity and orientation of the planetary rover at time t. Assuming the pose of the planetary rover at time t_0 in the global frame is $(x_{g0}, y_{g0}, \theta_{g0})$, $\theta(t)$ depends on θ_{g0} and the $\omega(\hat{t})$ with $\hat{t} \in [t_0, t]$.

To improve computational efficiency, $v(t)$ and $\omega(t)$ within a time interval $[t_0, t_n]$ are approximated as constant values v and ω. The rover's pose at time t_n can be formulated as

$$\begin{cases} x_{gn} = x_{g0} + v \int_{t_0}^{t_n} \cos\left(\theta_{g0} + \omega t\right) \mathrm{d}t \\ y_{gn} = y_{g0} + v \int_{t_0}^{t_n} \sin\theta\left(\theta_{g0} + \omega t\right) \mathrm{d}t \\ \theta_{gn} = \theta_{g0} + \omega\left(t_n - t_0\right) \end{cases} \tag{16}$$

Let $\Delta t = t_n - t_0$, which denotes the prediction time. Eq. (16) shows that the trajectory of the rover depends exclusively on its pose at time t_0, v, ω, and Δt. The sampling-based method is used to calculate trajectories with pairs (v, ω) of linear and angular velocities, along with a predefined prediction time Δt. In this way, an optional arc path set can be generated.

In order to ensure effective contact between the wheels and the ground, the wheel speed of the rover is typically restricted for minimizing wheel slippage and enhancing the traction and stability. Assuming each wheel's speed of the rover is limited to no more than n rps, the four-wheel differential model is simplified as a two-wheel differential model by setting the same speed for the wheels on the same side. It is important to note that the wheelbase of the simplified two-wheel differential model is referred as the virtual wheelbase, which differs from the wheelbase of the original model. This virtual wheelbase is expected to be determined through experimentation [34]. Subsequently, the relationship of the rover's linear velocity v and angular velocity ω is given as

$$\begin{aligned} \omega &= -\frac{2}{b}v + \frac{2r}{b}n_r \\ \omega &= \frac{2}{b}v - \frac{2r}{b}n_l \end{aligned} \tag{17}$$

where r represents the wheel radius, b is the virtual wheelbase, n_l and n_r are the wheel speeds of the left and right sides, which ranges from $-n$ to n.

It is evident that when n_l or n_r is constant, there exists a linear relationship between ω and v where $v \in [-n \cdot r, n \cdot r]$ and $\omega \in [-2n \cdot r/b, 2n \cdot r/b]$. The relationship between ω and v is presented in the schematic diagram, shown in Fig. 7. It represents the mobility characteristics of the planetary rover. The shaded area represents the value range for v and ω.

The linear velocity range $[-n{\cdot}r, n{\cdot}r]$ for the rover is discretized and sampled. Let v_i represent a sampled linear velocity. Through Eq. (17), the corresponding positive and negative angular velocity boundaries, denoted as ω_{pi} and ω_{ni}, are calculated for each v_i. These velocity pairs are located at the boundaries of the shaded area in Fig. 7. The maximum mobility of the rover can be achieved since the velocity pairs are generated using the maximum wheel speed.

The trajectories of the rover are generated based on velocity pairs and Δt. Non-zero linear velocity pairs correspond to either a straight line ($\omega = 0$) or an arc ($\omega \neq 0$), while zero linear velocity pairs correspond to turn-in-place motion. A resulting set of the optimal arc paths is shown in Fig. 8, where the central line represents maximum linear velocity and zero angular velocity. As the linear velocity decreases, the angular velocity increases. Turn-in-place motion is not shown.

4.3. Evaluation of arc paths

In order to select the optimal arc path as the local path, an evaluation function is used to score the optional arc paths in each cycle of local motion planning. The optional path with the highest score is the best trajectory for executing, and the corresponding velocity v^* and ω^* are the current optimal control commands for the rover. The evaluation function is defined as

$$G(v, \omega) = \alpha \cdot d(v, \omega) + \beta \cdot g(v, \omega) + \gamma \cdot p(v, \omega) \tag{18}$$

where $d(v, \omega)$, $g(v, \omega)$ and $p(v, \omega)$ are three evaluation functions. α, β, γ are the weights of these functions which satisfies $\alpha + \beta + \gamma = 1$. They can be adjusted based on environmental conditions or planning preferences.

The function $d(v, \omega)$ evaluates the safety of optional arc paths. The global coordinates of the obstacles are obtained from the OBS list mentioned in Section 4.1. Let d_o be the minimum Euclidean distance between the obstacles and all points along the arc path. The function $d(v, \omega)$ is defined as

$$d(v, \omega) = \begin{cases} 1, d_o > d_{om} \\ \dfrac{d_o}{d_{om}}, d_r < d_o \leq d_{om} \\ -\infty, d_o \leq d_r \end{cases} \tag{19}$$

where d_r is the minimum safety distance, and d_{om} is the maximum safe distance.

If $d_o \leq d_r$, the planetary rover will collide with obstacles while it is executing the current arc path. When $d_o > d_{om}$, the planetary rover is collision-free.

The function $g(v, \omega)$ is a measure of whether the optional arc path leads towards the mission goal. During local motion planning, potential deviations of the rover from the global path may happen while a global goal is used as the guide. Consequently, points on the global path are selected as local goals. The global path is initially projected onto a two-dimensional space, and the tracking distance s is set. The global path point located at a distance s in the travel direction of the rover is utilized as the local goal, as shown in Fig. 9.

The local goal is updated during each cycle of local motion planning. To ensure the local goal is ahead of all optional arc paths, s is required to exceed the length of the longest optional arc path. The function $g(v, \omega)$ is defined as

$$g(v, \omega) = 1 - \frac{d_g}{d_{gm}} \tag{20}$$

Table 1
Specific parameters of the two DEMs.

Data	Length	Width	Resolution
Real lunar DEM	33.9 m	38.0 m	0.1 m
Simulated DEM	50.0 m	50.0 m	0.1 m

where d_g represents the Euclidean distance between the endpoint of the optional arc path and the local goal, and d_{gm} is the maximum value of d_g for all optional arc paths.

The matching degree between the optional arc path and the global path is evaluated by $p(v, \omega)$. It is based on the minimum distance method, where the arc path and evaluation path are discretized, and the average Euclidean distance d_p between corresponding points is calculated. The evaluation path is the segment of the global path between the nearest global path point p_n and the local goal, as shown in Fig. 9. The function $p(v, \omega)$ is defined as

$$p(v, \omega) = 1 - \frac{d_p}{d_{pm}} \tag{21}$$

where d_{pm} is the maximum value of d_p for all optional arc paths.

The optimal arc path is selected based on the evaluation function. The rover follows the control commands corresponding to this path. This process is iterated until the rover reaches the endpoint of the global path. Through this method, the rover can effectively track the global path while ensuring safety.

5. Simulation results and discussion

Two types of simulation experiments are designed to validate the effectiveness of the proposed improved path planning and tracking control for planetary exploration rovers with traversable tolerance. These include a global path planning test and an autonomous navigation test in a mission for the planetary rover. In the global path planning test, the improved A*-based algorithm was compared with the classical A* algorithm. Its superiority and feasibility were demonstrated. In the autonomous navigation test, an exploration mission was simulated while the planetary rover was expected to travel autonomously from its initial position to the mission goal. A global path was generated by the improved A*-based algorithm. The planetary rover followed this path towards the mission goal by the multi-objective arc-based local motion planning. This test serves to validate the feasibility of the proposed method.

The two simulation experiments were carried out on a single laptop equipped with an AMD R7 5800H CPU, RTX 3070 GPU and 32 GB of RAM. The global path planning test was conducted in the Ubuntu 20.04 environment, utilizing the ROS Foxy framework. The planetary rover mission simulation was conducted in a Windows 10 environment, utilizing the Webots R2021a simulation software and Python 3.7.

5.1. Global path planning test

A comparative experiment was conducted to evaluate the proposed improved A*-based algorithm in comparison to the classical A* algorithm. The evaluation was performed using two DEMs, comprising small-scale real lunar surface data collected by Yutu-2 and large-scale simulated lunar surface data created by the researchers. The two DEMs are as shown in Fig. 10.

The specific parameters of the real lunar DEM and simulated DEM are listed in Table 1.

In this experiment, the classical A* algorithm employed an eight-connected grid for state expansion. The heuristic function

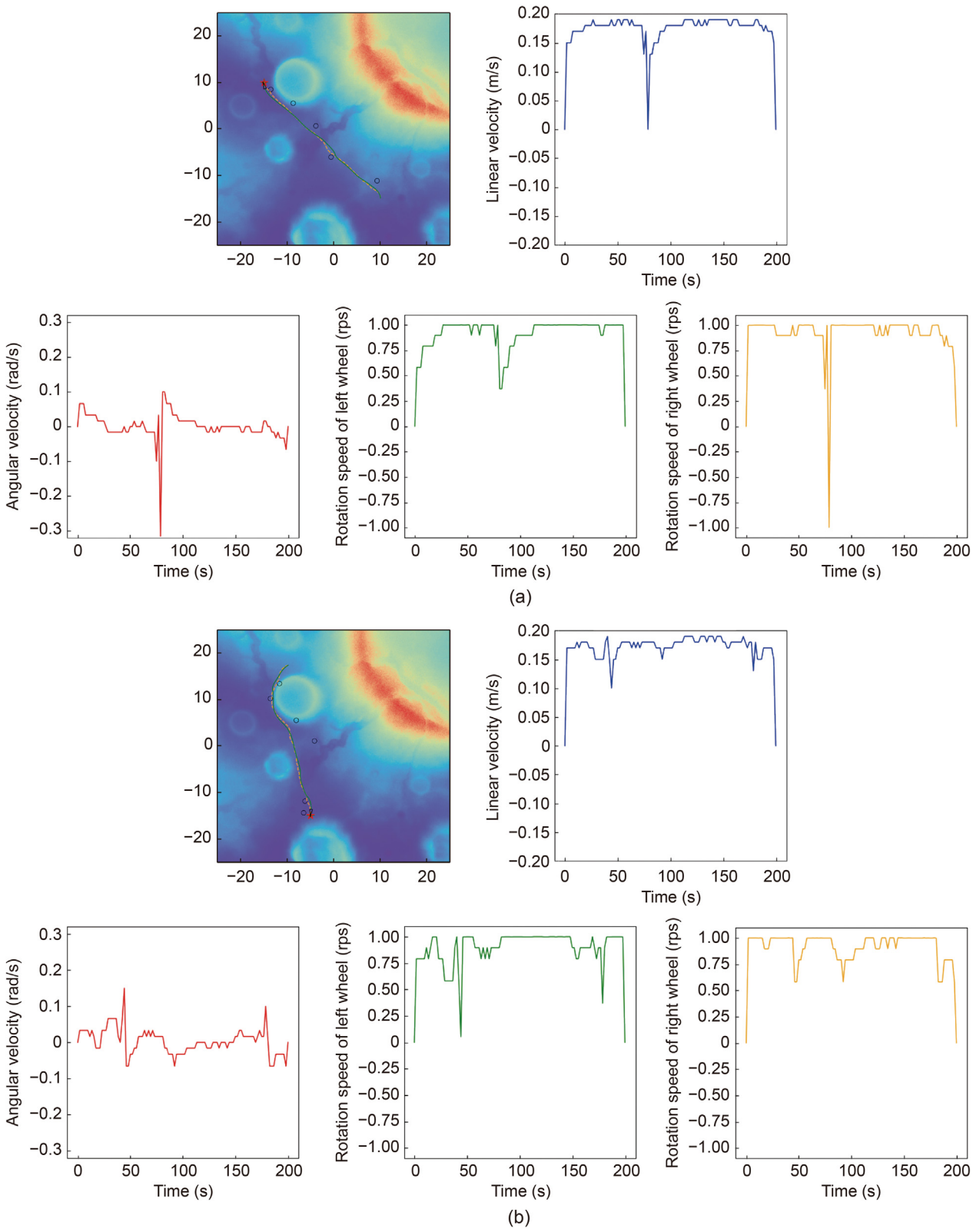

Fig. 15. (a) The result of autonomous navigation test 1 on simulated terrain. (b) The result of autonomous navigation test 2 on simulated terrain.

$h(s)$ was defined as the Euclidean distance, while the actual cost function $g(s)$ represented the distance in three-dimensional space. In addition, the improved A*-based algorithm utilized the motion primitive library for searching, as shown in Fig. 4. The motion primitives corresponding to a slope angle of 0° were identical to those found in the two-dimensional motion primitive library. The resolution of the slope angle was set at 5°, while the maximum slope angle ϕ_{max} = 30°. The cost function $g(s)$ was represented by Eq. (12), while the heuristic function $h(s)$ was denoted as Eq. (13).

The partial planning results of two algorithms operating with identical initial and goal states are shown in Fig. 11.

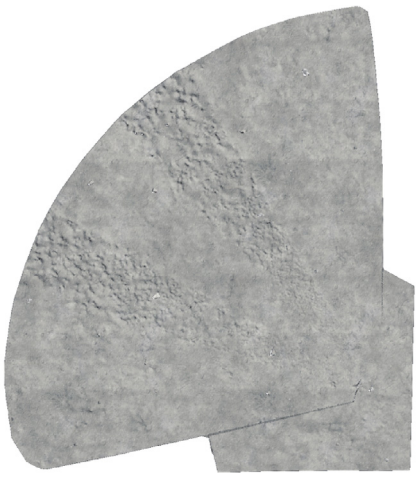

Fig. 16. Test scene based on real lunar DEM and addition obstacles.

Each subfigure in Fig. 11 presents a comparison of the planning paths generated by the classical A* algorithm (left) and improved A*-based algorithm (right). Identical paths from different perspectives are shown in the subfigures within the same column. The initial and goal states are denoted by the white and black arrows depicted in the figure, while the heading of the planetary rover is signified by the direction of the arrows. It shows that the starting and ending directions of the path generated by the improved A*-based algorithm are consistent with the initial and target orientations. However, the classical A* algorithm only takes into account the initial and target coordinates. Moreover, the path generated by improved A*-based algorithm forms a smooth curve in contrast to the path generated by the classical A* algorithm, which contains numerous sharp corners. This characteristic is anticipated to diminish the operational efficiency of the planetary rover. In Fig. 11(b), the path generated by the classical A* algorithm intersects the ring-shaped mountains and concave terrain directly, rendering it impassable for the planetary rover. This result represents a failure. In contrast, the path generated by improved A*-based algorithm effectively circumvents these obstacles, enabling the safe guidance of the planetary rover to the target.

Forty planning tasks were executed using two DEMs to compare the planning time and path length of two algorithms. The first twenty planning tasks were conducted on the real DEM, while the simulated DEM was utilized for the subsequent twenty planning tasks. The initial and goal states of each planning task were selected randomly. As shown in Fig. 12, less planning time is generally required in the improved A*-based algorithm compared to the classical A* algorithm. The planning results of the improved A*-based algorithm lead to an average time savings of 30.05% in the planning process. Due to the generation of a smoother path and the consideration of the planetary rover's attitude at the initial and goal states, the improved A*-based algorithm may result in a slightly longer path length compared to the classical A* algorithm. However, the increment is only by an average of 0.80%, which is not significant.

5.2. Autonomous navigation on simulated terrain

Complex terrains such as gullies and ring mountains are included in the simulated DEM shown in Fig. 10(b) which makes it more challenging. Consequently, this DEM was utilized to create scene in Webots simulation software for conducting the autonomous navigation test. The test scene is shown in Fig. 13. In order to validate the obstacle avoidance capability of the proposed local motion planning method, the rocks were randomly placed and used as obstacles in the scene. These rocks are unknown in the DEM. Thus, they are unable to be considered in the global path planning process.

The planetary exploration rover model is shown in Fig. 14. The rover employed four-wheel differential drive, with dimensions of 0.9 m×0.6 m×0.38 m and a wheel radius of 0.19 m. The virtual wheel spacing was experimentally determined to be approximately 1.20 m. An RGB-D camera was mounted on the front of the rover, angled downward with a pitch of −15°. This serves to improve the rover's perception ability of nearby obstacles.

In the experiment, the rover's wheel speed was limited to 1 rps. Thus, the rover's maximum linear and angular velocities are 0.19 m/s and 0.316 rad/s, respectively. The sampling-based arc path generation method described in Section 4.2 is used to generate the set of optional arc paths. The velocity combinations utilized are listed in Table 2.

Start positions and mission goals were randomly selected in this test. The improved A*-based algorithm was used to generate a global path from the start position to the mission goal. Subsequently, the multi-objective arc-based local motion planning method was employed to guide the rover in following the global path. Visualizations of two autonomous navigation tests are shown in the leftmost figures of Figs. 15(a) and 15(b). The mission goal is represented by the red star, while the obstacles are symbolized by the black circles. The global path and the rover's motion trajectory are illustrated by the orange dashed and green solid lines, respectively. The remaining four figures on the right side of Figs. 15(a) and 15(b) show the rover's linear velocity, angular velocity, and the rotation speeds of its left and right wheels over time during the two tests. In this test, the rover's autonomous navigation ability is demonstrated by effectively avoiding obstacles and successfully tracking the global path. Furthermore, the effectiveness of the proposed local motion planning method in improving the rover's mobility is also demonstrated within the wheel speed constraints. Thus, the planetary rover can be safely and efficiently guided to the mission goal on simulated terrain through the proposed improved path planning and tracking control method.

5.3. Autonomous navigation test on lunar terrain

To further validate the feasibility of proposed algorithm on the actual lunar terrain, a test was conducted using the real lunar DEM shown in Fig. 10(a) derived from exploration data collected by the Yutu-2 rover. As mentioned in Section 5.2, the DEM was loaded into the Webots software, where additional rocks were introduced as obstacles to evaluate the obstacle avoidance capabilities of the local motion planning algorithm. The test scene is shown in Fig. 16.

The same planetary rover described in Section 5.2 is utilized in this test, as shown in Fig. 14. The wheel speeds are similarly restricted to 1 rps, and the speed combinations employed are listed in Table 2.

Visualizations of two autonomous navigation tests are shown the leftmost figures of Figs. 17(a) and 17(b), where the starting positions and mission goals are randomly selected. The red star in the figure represents the mission goal, while the black circles represent obstacles. The global path is illustrated by orange dashed lines, and the rover's motion trajectory is depicted by green solid lines. The remaining four figures on the right side of Figs. 17(a) and 17(b) show the rover's linear velocity, angular velocity, and the rotation speeds of its left and right wheels over time for both tests. In these tests, the proposed algorithm effectively controls the rover to avoid obstacles and successfully

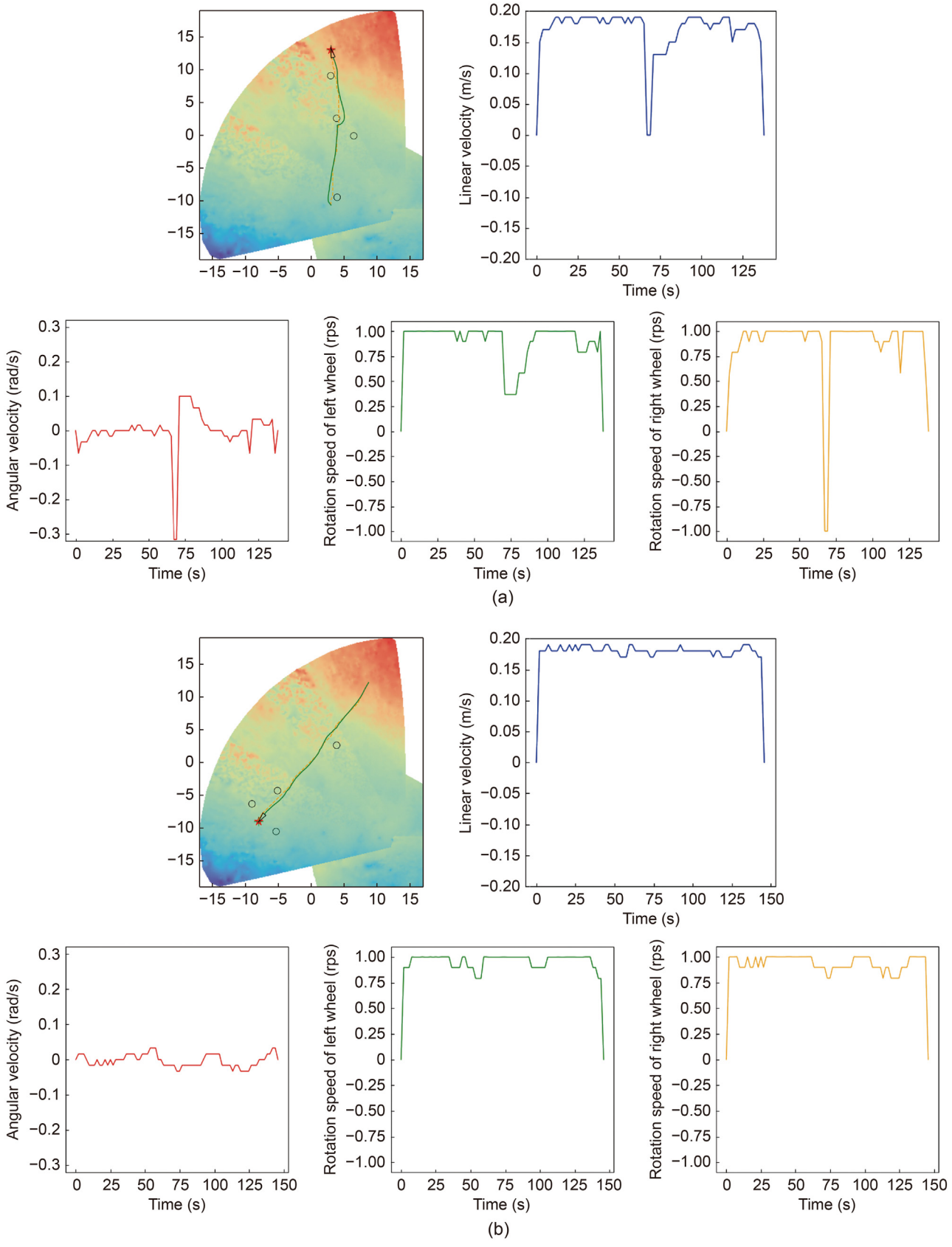

Fig. 17. (a) The result of autonomous navigation test 1 on lunar terrain. (b) The result of autonomous navigation test 2 on lunar terrain.

Table 2
Velocity Combination for optional arc path set.

Velocity type	Velocity combinations								
v (m/s)	±0.19	0.18	0.17	±0.15	0.13	±0.10	0.07	0.04	0
ω (rad/s)	0	±0.016	±0.033	±0.066	±0.10	±0.15	±0.20	±0.25	±0.316

track the global path on actual lunar terrain. In addition, the effectiveness of the proposed local motion planning method in enhancing the rover's mobility under wheel speed constraints is demonstrated. Consequently, with the proposed improved path planning and tracking control method, the planetary rover can safely and efficiently achieve its mission objectives under actual lunar surface conditions.

In addition, the proposed improved path planning and tracking control method has received the National Second Prize in the 18th "Challenge Cup" National College Student Extracurricular Academic Science and Technology Works Competition, which is about autonomous navigation capabilities of planetary rovers in lunar surface exploration missions.

6. Conclusion

The improved path planning and tracking control method is proposed in this paper to address autonomous path planning challenges in planetary rover exploration missions. With this method, a planetary rover can safely and efficiently navigate from its initial position to a mission goal. Global paths that satisfy the motion constraints of the planetary rover are generated by the proposed improved A*-based global planning method with motion primitives. The global paths adhere to terrain constraints by considering various terrain features such as roughness, slope, height, etc. The planetary rover can follow the global path to the mission goal using the proposed multi-objective arc-based local motion planning method. An optional arc path set is generated based on the mobility capabilities of the planetary rover. Under the constraints of multiple objectives, the optimal arc path is selected by the evaluation function. The rover achieves real-time obstacle avoidance and accurate tracking of the global path through optimal motion. The simulation results show that the proposed improved A*-based algorithm reduces the planning time by 30.05% and generates smoother paths compared with the classical A* algorithm. The proposed multi-objective arc-based method improves the efficiency of the planetary rover's motion, ensuring a safer and more efficient arrival at the mission goal along the global path.

CRediT authorship contribution statement

Haojie Zhang: Writing – review & editing, Conceptualization. **Feng Jiang:** Writing – original draft, Methodology. **Qing Li:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was funded by the State Key Laboratory, China (KJW6142210210308) and the National Natural Science Foundation of China (61806183).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.birob.2025.100219.

References

[1] Y. Wang, W. Wan, S. Gou, M. Peng, Z. Liu, K. Di, L. Li, T. Yu, J. Wang, X. Cheng, Vision-based decision support for rover path planning in the Chang'e-4 mission, Remote. Sens. 12 (4) (2020) 624, http://dx.doi.org/10.3390/rs12040624.
[2] X. Diao, W. Chi, J. Wang, Graph neural network based method for robot path planning, Biomim. Intell. Robot. 4 (1) (2024) 100147, http://dx.doi.org/10.1016/j.birob.2024.100147.
[3] M.Q.-H. Meng, R. Song, Legged mobile robots for challenging terrains, Biomim. Intell. Robot. 2 (1) (2022) 100034, http://dx.doi.org/10.1016/j.birob.2021.100034.
[4] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Trans. Syst. Sci. Cybern. 4 (2) (1968) 100–107, http://dx.doi.org/10.1109/tssc.1968.300136.
[5] A. Stentz, The D Algorithm for Real-Time Planning of Optimal Traverses, Carnegie Mellon University, the Robotics Institute, 1994, https://api.semanticscholar.org/CorpusID:14389316.
[6] S. Dong, H. Ju, H. Xu, An improvement of D lite algorithm for planetary rover mission planning, in: 2011 IEEE International Conference on Mechatronics and Automation, IEEE, 2011, pp. 1810–1815, http://dx.doi.org/10.1109/ICMA.2011.5986254.
[7] T. Lei, T. Sellers, C. Luo, D.W. Carruth, Z. Bi, Graph-based robot optimal path planning with bio-inspired algorithms, Biomim. Intell. Robot. 3 (3) (2023) 100119, http://dx.doi.org/10.1016/j.birob.2023.100119.
[8] L. Zhang, K. Cai, Z. Sun, Z. Bing, C. Wang, L. Figueredo, S. Haddadin, A. Knoll, Motion planning for robotics: A review for sampling-based planners, Biomim. Intell. Robot. (2025) 100207, http://dx.doi.org/10.1016/j.birob.2024.100207.
[9] R. Raja, A. Dutta, Path planning in dynamic environment for a rover using A* and potential field method, in: 2017 18th International Conference on Advanced Robotics, ICAR, IEEE, 2017, pp. 578–582, http://dx.doi.org/10.1109/icar.2017.8023669.
[10] X. Yu, Q. Huang, P. Wang, J. Guo, Comprehensive global path planning for lunar rovers, in: 2020 3rd International Conference on Unmanned Systems, ICUS, IEEE, 2020, pp. 505–510, http://dx.doi.org/10.1109/icus50048.2020.9274967.
[11] M. Sutoh, M. Otsuki, S. Wakabayashi, T. Hoshino, T. Hashimoto, The right path: comprehensive path planning for lunar exploration rovers, IEEE Robot. Autom. Mag. 22 (1) (2015) 22–33, http://dx.doi.org/10.1109/mra.2014.2381359.
[12] L. Zhou, L. Yang, H. Tang, Research on path planning algorithm and its application based on terrain slope for slipping prediction in complex terrain environment, in: 2017 International Conference on Security, Pattern Analysis, and Cybernetics, SPAC, IEEE, 2017, pp. 224–227, http://dx.doi.org/10.1109/spac.2017.8304280.
[13] J. Zhang, Y. Xia, G. Shen, A novel learning-based global path planning algorithm for planetary rovers, Neurocomputing 361 (2019) 69–76, http://dx.doi.org/10.1016/j.neucom.2019.05.075.
[14] T. Tanaka, H. Malki, A deep learning approach to lunar rover global path planning using environmental constraints and the rover internal resource status, Sensors 24 (3) (2024) 844, http://dx.doi.org/10.3390/s24030844.
[15] J.J. Biesiadecki, M.W. Maimone, The mars exploration rover surface mobility flight software driving ambition, in: 2006 IEEE Aerospace Conference, IEEE, 2006, pp. 15–pp, http://dx.doi.org/10.1109/AERO.2006.1655723.
[16] S. Hayati, A. Rankin, W. Kim, P. Leger, R. Castano, K. Ali, Advanced robotics technology infusion to the NASA Mars Exploration Rover (MER) project, IFAC Proc. Vol. 40 (15) (2007) 180–185, http://dx.doi.org/10.3182/20070903-3-fr-2921.00033.
[17] J. Carsten, A. Rankin, D. Ferguson, A. Stentz, Global path planning on board the mars exploration rovers, in: 2007 IEEE Aerospace Conference, IEEE, 2007, pp. 1–11, http://dx.doi.org/10.1109/aero.2007.352683.
[18] O. Toupet, An Overview of the Mars 2020 Perseverance Rover's Enhanced Path-Planner, 2020, http://dx.doi.org/10.2014/53782.
[19] X. Yan, L. Xiang, T. Bao-Yi, M. Xiao-Yan, Autonomous local obstacle avoidance path planning of lunar surface exploration rovers, Control Theory Appl. 36 (12) (2019) 2042–2046, http://dx.doi.org/10.7641/CTA.2019.90590.
[20] S. Karaman, E. Frazzoli, Sampling-based algorithms for optimal motion planning, Int. J. Robot. Res. 30 (7) (2011) 846–894, http://dx.doi.org/10.1177/0278364911406761.
[21] J.A. Sethian, Fast marching methods, SIAM Rev. 41 (2) (1999) 199–235, http://dx.doi.org/10.1137/s0036144598347059.

[22] M.I. Jordan, T.M. Mitchell, Machine learning: Trends, perspectives, and prospects, Science 349 (6245) (2015) 255–260, http://dx.doi.org/10.1126/science.aaa8415.
[23] W. Wu, X. Xie, M. Wei, N. Liu, X. Chen, P. Yan, M. Omar, L. Xu, Planetary rover path planning based on improved A* algorithm, in: Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, August 8–11, 2019, Proceedings, Part IV 12, Springer, 2019, pp. 341–353, http://dx.doi.org/10.1007/978-3-030-27538-9_29.
[24] S. Garrido, L. Moreno, F. Martín, D. Alvarez, Fast marching subjected to a vector field–path planning method for mars rovers, Expert Syst. Appl. 78 (2017) 334–346, http://dx.doi.org/10.1016/j.eswa.2017.02.019.
[25] X. Yu, P. Wang, Z. Zhang, Learning-based end-to-end path planning for lunar rovers with safety constraints, Sensors 21 (3) (2021) 796, http://dx.doi.org/10.3390/s21030796.
[26] M. Ono, T.J. Fuchs, A. Steffy, M. Maimone, J. Yen, Risk-aware planetary rover operation: Autonomous terrain classification and path planning, in: 2015 IEEE Aerospace Conference, IEEE, 2015, pp. 1–10, http://dx.doi.org/10.1109/aero.2015.7119022.
[27] B. Rothrock, R. Kennedy, C. Cunningham, J. Papon, M. Heverly, M. Ono, Spoc: Deep learning-based terrain classification for mars rover missions, in: AIAA SPACE 2016, 2016, p. 5539, http://dx.doi.org/10.2514/6.2016-5539.
[28] J.R. Sánchez-Ibánez, C.J. Pérez-del Pulgar, M. Azkarate, L. Gerdes, A. García-Cerezo, Dynamic path planning for reconfigurable rovers using a multi-layered grid, Eng. Appl. Artif. Intell. 86 (2019) 32–42, http://dx.doi.org/10.1016/j.engappai.2019.08.011.
[29] W. Huiting, Y. Meng, L. Yuye, H. Tao, Z. Bo, Autonomous navigation path planning algorithm for rovers in lunar south pole surface, J. Deep. Space Explor. 10 (6) (2023) 598–607, http://dx.doi.org/10.15982/j.issn.2096-9287.2023.20230084.
[30] C.L. Bottasso, D. Leonello, B. Savini, Path planning for autonomous vehicles by trajectory smoothing using motion primitives, IEEE Trans. Control Syst. Technol. 16 (6) (2008) 1152–1168, http://dx.doi.org/10.1109/TCST.2008.917870.
[31] R.J. Cripps, M. Hussain, S. Zhu, Smooth polynomial approximation of spiral arcs, J. Comput. Appl. Math. 233 (9) (2010) 2227–2234, http://dx.doi.org/10.1016/j.cam.2009.10.008.
[32] W. Wong, J. Lee, Statistical Analysis of Geographic Information with ArcView GIS and ArcGIS, Wiley, 2005, https://hub.hku.hk/handle/10722/192432.
[33] Y. Wang, M. Peng, K. Di, W. Wan, Z. Liu, Z. Yue, Y. Xing, X. Mao, B. Teng, Vision based obstacle detection using rover stereo images, Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci. 42 (2019) 1471–1477, http://dx.doi.org/10.5194/isprs-archives-XLII-2-W13-1471-2019.
[34] T. Wang, Y. Wu, J. Liang, C. Han, J. Chen, Q. Zhao, Analysis and experimental kinematics of a skid-steering wheeled robot based on a laser scanner sensor, Sensors 15 (5) (2015) 9681–9702, http://dx.doi.org/10.3390/s150509681.