TiFA: A Terrain-informed Navigation Framework for Articulated Tracked Robots in Rescue Missions

Yifei Wang[1], Weifan Zhang[1], Yuxiang Li[1], Jiancheng Wang[1] and Haoyao Chen[1]

Abstract

Articulated tracked robots are well-suited for post-disaster rescue missions, but their navigation in large-scale, complex environments presents significant challenges. Current methodologies often face limitations, including high computational demands, inadequate real-time capabilities, and compromised safety during cross-story maneuvers. This paper introduces a novel navigation framework that addresses these challenges through a hierarchical global planner, a manifold-optimization-based local planner, and an on-manifold model predictive controller. The framework leverages various forms of terrain information extracted from the environment in all planning and control stages to facilitate global path generation, local path optimization, and robust whole-body motion control. With coherent interaction among different modules, the framework ensures reliable navigation performance. Extensive ablation simulations and comparative real-world experiments demonstrate that the proposed framework significantly improves the success rate, safety, and efficiency over state-of-the-art methods, while replacing any key method within our framework leads to a noticeable degradation in performance. Specifically, in large-scale scenarios with complex terrain spanning multiple acres, the hierarchical global planner can generate feasible paths within seconds, reducing the time cost by 78.81%. The manifold-optimization-based local planner effectively ensures obstacle avoidance while fully meeting the maneuvering safety requirements in various typical challenging terrains. The holistic controller enabled the robot to stably and reliably track paths on steep ($\geq 30^{\circ}$) and rugged terrain.

Keywords

Search and Rescue Robots, Motion and Path Planning, Whole-Body Motion Planning and Control

1 Introduction

In recent years, various robots have been developed to mitigate potential harm to rescuers while improving the efficiency and success rate of disaster rescue missions (Delmerico et al. 2019). Based on their mechanical structure and operational modes, rescue robots can be broadly categorized into aerial and ground robots. Aerial robots, predominantly quadrotor drones, are characterized by their lightweight design and high maneuverability, enabling efficient environmental exploration and perception supported by relatively mature navigation strategies (Faessler et al. 2016; Bircher et al. 2018). Ground robots, in contrast, can carry larger payloads, enabling direct interaction with disaster environments and undertaking more complex rescue tasks.

Ground mobile robots with various configurations have been developed to navigate unstructured environments, including wheeled (Chiu et al. 2005), tracked (Schwarz et al. 2017; Choi et al. 2019), and bio-inspired legged robots (Hutter et al. 2017). In disaster-stricken areas, the terrain is often highly complex and varied, featuring rubble, steep slopes, and staircases, which pose significant challenges to the mobility of ground robots. Consequently, different locomotion paradigms exhibit distinct strengths and limitations. Wheeled robots exhibit both high stability and strong controllability, but struggle to maneuver over uneven terrains such as staircases and rubble. Legged and other bio-inspired robots exhibit superior mobility but are challenging to control and suffer from lower stability. In contrast, tracked robots can achieve a favorable balance between high maneuverability and outstanding controllability in complex terrains. This balance is particularly evident in articulated tracked robots incorporating bio-inspired auxiliary flipper designs, which simultaneously enhance mobility and stability. Therefore, articulated tracked robots have become the mainstream choice for small-scale ground rescue robots (Delmerico et al. 2019). This paper addresses the critical issues and challenges in the autonomous navigation of such robots within complex, three-dimensional rescue environments (see Fig. 1).

Articulated tracked robots (ATRs) are well-suited for rescue tasks but face substantial navigation challenges in large-scale 3D rescue scenarios with multi-floor structures, significant elevation changes, and rugged terrains. The sparse and complex topological connectivity among traversable areas (e.g., staircases between collapsed floors) greatly reduces navigation efficiency. Compounding this, given the ATR's diverse motion DOFs and its SE(3) centroid state in 3D space, it is insufficient to address such a high-dimensional and complex navigation problem using traditional 2D planar navigation methods. Furthermore, the traversability depends heavily on both the configuration of the ATR and the terrain, significantly complicating the

[1] Harbin Institute of Technology Shenzhen, P.R. China

planning process. Moreover, certain terrain features, such as steep slopes and staircases, demand careful assessment of the robot's capabilities and constraints. The robot has to leverage terrain information to navigate safely. Existing research heavily depended on vanilla feedback control methods to maneuver ATRs in complex terrains (Yuan et al. 2019; Chen et al. 2023). However, these methods face substantial limitations when applied to complex multi-input, multi-output systems like ATR. And they also struggled to achieve whole-body maneuvers and safe path tracking while considering the robot's kinematic constraints.

Figure 1. The articulated tracked robot navigating on the spiral staircase (a), the rubble (b), and the outdoor grassland (c).

To address the aforementioned challenges, we propose a tightly integrated navigation framework for ATRs that leverages terrain information across different planning and control stages. A hierarchical global planner rapidly generates feasible robot centroid paths using discretized terrain maps, significantly improving planning efficiency in large-scale 3D environments while ensuring reliable reachability. This initialized path is then refined by a manifold-optimization-based local planner, which formulates a multi-objective problem to improve path quality under kinematic and terrain-specific constraints, enhancing safety and adaptability in complex terrains. Finally, the on-manifold model predictive controller utilizes only the local centroid path and local terrain perception to achieve real-time, whole-body coordination of the chassis and flippers. This control strategy avoids high-dimensional planning and enables robust execution under rugged terrain conditions. By decoupling the navigation process while ensuring close inter-module synergy, the framework balances completeness and optimality, thereby ensuring both computational efficiency and system robustness. The main contributions of this paper can be summarized as follows:

- We introduced a hierarchical global planner (**CT-DC**) based on a discrete terrain information grid map to overcome the time-consuming bottleneck of feasible path generation for ground robots in large-scale, complex 3D multi-floor environments. Compared to the existing SOTA works, the planning time for homotopic feasible paths was reduced by 78.81% at most.
- We designed a manifold-optimization-based local planner (**ManiOpt**) to ensure the safety and adaptability of the robot across different terrains. The feasibility and smoothness of paths in various complex terrains outperformed the control group.
- We developed an on-manifold model predictive controller (**M^{2}PC**) to achieve the holistic control of ATR on rugged terrains. This controller ensured close adherence between the robot's body and the terrain under various constraints, enhancing stability and reliability.
- We conducted systematic ablation simulations and real-world experiments to validate each module's contribution and the framework's overall performance, providing valuable insights and benchmarks for future research on ATR navigation in rescue scenarios.

2 Related Works

This section reviews relevant works on autonomous ground navigation in 3D unstructured environments, which have inspired our research.

2.1 Navigation of normal ground robots in unstructured environments

For ground robots navigating unstructured environments, assessing the ground's traversability is crucial. Krüsi et al. (2017) pioneered the use of planar fitting near surface points in point cloud maps to assess traversability, which was further used as a basis for state validity checks in sampling-based planners to generate initial 3D paths. Similarly, PUTN (Jian et al. 2022) employed a dimensional augmentation strategy during sampling, where poses are first sampled in 2D space and then projected onto the ground to recover 3D poses. This approach reduced invalid sampling and enhanced sampling efficiency. Xu et al. (2023) extended this idea by iteratively optimizing the robot's pose in SE(2) space to obtain the corresponding pose in SE(3) space. Their work provided a more detailed and specific assessment of ground traversability, and the polynomial trajectory optimization improved planning efficiency in complex terrains. To address navigation in multi-floor environments, Wang et al. (2023) identified traversable areas directly in large-scale point cloud maps, eliminating the need for the 2D-to-3D recovery process.

Extracting more detailed maps with terrain information from point clouds is also a common approach. Wang et al. (2017) converted point clouds into multi-layer maps by abstracting spatial features and using slope information to create traversability maps for practical navigation. In contrast, STEP's abstracted map incorporated more realistic environmental elements such as different terrains and obstacles, allowing the robot to find feasible paths in complex mining environments (Fan et al. 2021).

Moreover, learning-based methods can also assist robots in obtaining traversability maps in complex

environments. Weerakoon et al. (2022) and Hu et al. (2021) employed reinforcement learning to identify traversable areas for robots and then used sampling-based planning methods to find feasible paths within these areas. Cai et al. (2023) proposed a more interpretable learning-based method that takes semantic and elevation maps as inputs to a neural network, producing a traction distribution map to guide robot navigation in outdoor environments. However, learning-based path planning methods often exhibit limited robustness and require multiple training rounds for specific scenarios, making them challenging to deploy in practical rescue missions.

2.2 Navigation of ATRs in unstructured environments

Tracked robots equipped with auxiliary flippers have more adaptability in unstructured environments. However, algorithms designed for wheeled and traditional tracked robots can not be directly adapted to solve the ATR navigation problem.

Yuan et al. (2019) proposed a stability evaluation criterion, suggesting that tracked robots with auxiliary flippers should maintain maximum contact with the ground during maneuvers in complex environments to ensure stability. In their work, feasible ATR chassis poses were generated from discrete terrain point clouds, and the optimal path waypoints (including auxiliary flipper joint states) were selected from the corresponding possible configurations. A feedback controller was then used to execute the robot's path. Similarly, Chen et al. (2023) introduced an evaluation system incorporating sequential considerations and pose prediction to ensure the robot's motion stability and continuity. Xu et al. (2024) further classified ATR configuration states into several forms and achieved segmented whole-body motion planning and control through mixed optimization.

Gianni et al. (2016) proposed an ATR tracking control framework specifically designed for rescue scenarios. The framework utilized the robot's centroid path and abstracted terrain planes to design the configuration of the auxiliary flippers. A complex kinematic solver and a feedback controller were then employed to achieve whole-body control of the robot.

Except for traditional control methods, reinforcement-learning-based control for ATRs has been validated in scenarios where sufficient prior knowledge is available (Mitriakov et al. 2021; Pan et al. 2023). These approaches abandon traditional methods' manually defined, often intricate evaluation criteria and instead rely on the explore-exploit mechanism of reinforcement learning. Through extensive offline simulations, exclusive control strategies were trained for ATRs to perform tasks such as stair climbing and many others. To reduce the learning cost for robots, Azayev and Zimmermann (2022) utilized human supervision to guide the motion planning and control of the robot. Human-in-the-loop methods enabled the robot to achieve adaptive whole-body control through learning-based methods. However, these techniques require substantial training and complex environmental abstractions, making them challenging to deploy in complex rescue scenarios.

Previous research has commonly employed terrain point clouds to guide ground robots in unstructured environments. However, they often lacked sufficient abstraction and management of terrain information, leading to longer path generation times and more limited applicability. The safety and adaptability of robot paths across various terrains have also not been adequately addressed. Additionally, existing ATR control methods remain relatively simple and do not fully consider the safety of motion control in complex terrain situations. Although learning-based control methods avoided sophisticated logical inference and theoretical derivation, they entailed significantly higher deployment costs and time consumption than traditional methods. These methods also struggled to cover some common corner cases in real-world scenarios and suffer from serious generalization issues. Thus, this paper proposes a terrain-informed navigation framework that effectively addresses the limitations of existing approaches and offers practical solutions for deploying autonomous ATRs in complex 3D disaster environments.

3 Problem Statement

Fig. 2 illustrates the typical mechanical design of an ATR. The ATR's tracked chassis has a centroid frame denoted as F_b, with the front flipper frame labeled F_f and the rear flipper frame labeled F_r. The angles between these flippers and the chassis are $[\phi_f, \phi_r]$. The kinematic characteristics of the tracked chassis allow movements along the robot's longitudinal direction and in-place rotation $[v_x, \omega_z]$. The rotational speeds of the auxiliary flipper joint drives are denoted as $[\psi_f, \psi_r]$.

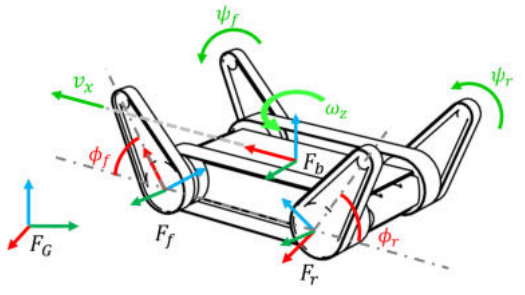


Figure 2. The typical mechanical design of an ATR, with the corresponding frame definitions and the motion space.

Let $\mathcal{X} \subset \mathbb{R}^3$ denote the robot's workspace in a 3D environment, and let $\mathcal{X}_{obs} \subset \mathcal{X}$ represent regions obstructed by obstacles. The traversable region is defined as $\mathcal{X}_{free} = \mathcal{X} \backslash \mathcal{X}_{obs}$. For ground robots, the workspace is strictly confined to the terrain surface, denoted as $\mathcal{S}_{terr} \subset \mathbb{R}^3$. Thus, the traversable subspace for the ground robot is $\mathcal{X}_{trav} = \mathcal{S}_{terr} \cap \mathcal{X}_{free}$. The navigation problem for ATRs in unstructured environments is defined as follows: Given the start and goal state $\mathbf{x}_s, \mathbf{x}_g \in \mathcal{X}_{trav}$, the navigation objective is

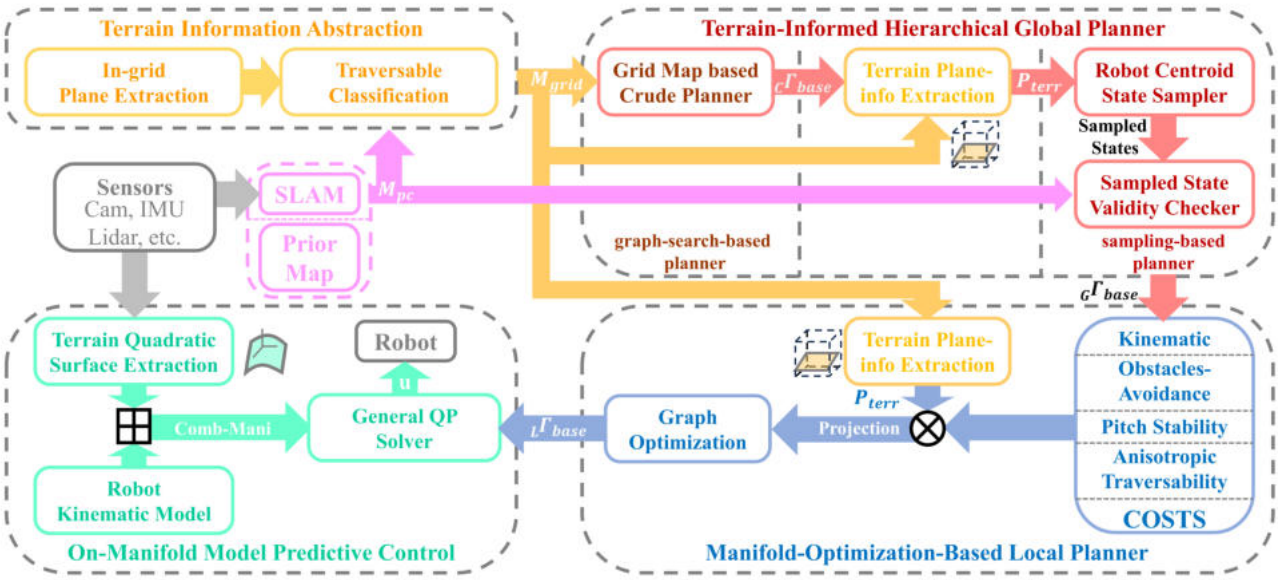


Figure 3. The proposed framework comprises several tightly integrated modules for terrain information abstraction (Section 4.1), global path generation (Section 4.2), local path optimization (Section 4.3), and whole-body control (Section 4.4) of the ATR in rescue scenarios.

to use the planning strategy π^*_{plan} to generate feasible paths Γ_{base} for the robot's centroid based on the point cloud map M_{pc}; then use the control strategy π^*_{ctrl} and the employed real-time sensor data to guide the robot from $\mathbf{x}_s$ to $\mathbf{x}_g$ along Γ_{base}. The proposed solution addresses four major challenges in ATR navigation within 3D unstructured environments:

- Traversable areas in unstructured cross-story environments are sparse and poorly connected, and assessing robot traversability over large complex terrains can be very difficult.
- In large-scale 3D environments, classical planning methods struggle to rapidly explore and evaluate terrain, resulting in extended path generation times.
- Conventional path optimization methods fail to realize obstacle avoidance while ensuring path-terrain adherence, and they cannot adjust the path according to specific terrain features.
- The complex mechanism of ATRs makes decoupled or loosely coupled control methods inadequate for achieving whole-body control, which is necessary to ensure safe maneuverability on complex terrains.

4 Framework

In this paper, we propose a tightly integrated framework that leverages terrain-aware representations across all planning and control modules (see Fig. 3) to address the challenges of the ATR navigation problem.

First, the terrain information abstraction module (Section 4.1) efficiently extracts and manages ground terrain information in the form of a 3D terrain-info grid map, based on either online point cloud maps (obtained through popular SLAM methods like FAST-LIO2 proposed by Xu et al. (2022)) or pre-acquired prior point cloud maps. This well-organized terrain information is subsequently utilized by both the global path generation and local path optimization. Next, we present a hierarchical global planner (Section 4.2) to ensure both the completeness and efficiency of path generation. We further introduce a parallelization strategy that significantly accelerates the path generation process, while improving computational efficiency. Subsequently, we propose a manifold-optimization-based local planner (Section 4.3) that refines the global path into a local path suitable for the robot's centroid tracking control. This optimized path respects the robot's kinematic constraints, performs obstacle avoidance, and satisfies a range of scenario-specific objectives. Finally, we introduce an on-manifold model predictive controller (M^2PC) (Section 4.4) specifically designed for ATRs. Leveraging the smooth and collision-free local path, M^2PC integrates the extracted terrain features with the robot's kinematic constraints to enable safe, smooth, and terrain-adaptive whole-body control during navigation execution.

The framework integrates different terrain representations at the planning and control stages to balance efficiency and optimality. Specifically, the global planner operates on the 3D terrain-info grid map enriched with discrete planar terrain approximations. The local planner also performs path optimization on the manifold spaces constructed from terrain planes in the grid map. These planar terrain models allow rapid and effective path planning even in large-scale environments. The whole-body controller uses a higher-fidelity terrain model, applying quadratic surface fitting to better capture terrain curvature, enabling more adaptive and stable motion over complex surfaces.

4.1 Discrete terrain information abstraction

In complex 3D rescue environments, ground robot path planning has to fully use the obtained terrain geometry information. However, for large-scale terrain information abstraction, general point cloud fitting methods often homogenize a lot of discontinuous terrain. For example, the floors of different disconnected rooms on the same level might have similar geometric information. Similarly, many traversable surfaces are at the same height but belong to different buildings. Moreover, many general terrain analysis methods fail to capture the topological relationships and distance information between traversable regions (Krüsi et al. 2017; Lee et al. 2022), which limits their utility for ground robot navigation in complex environments.

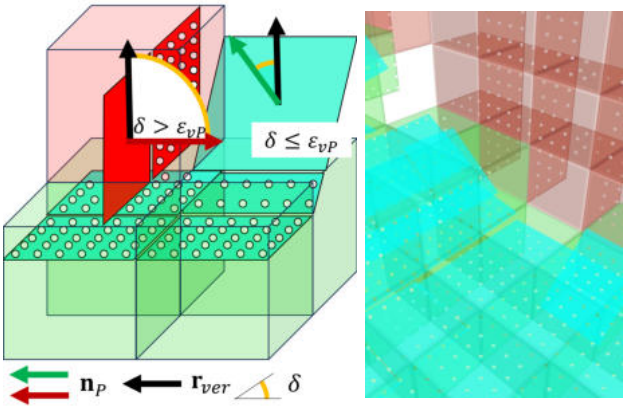


Figure 4. Grid map with terrain information. White points represent the point cloud data in the map. Cyan planes denote the traversable terrain planes extracted within the grids, while red planes denote the non-traversable obstacle planes. The geometric information of these planes is used to further assess the passability of each grid cell. Green indicates traversable grids, and red indicates non-traversable grids.

We adopted a grid-based terrain information abstraction method to address the abovementioned issue. The robot's workspace $\mathcal{X}$ is divided into a 3D grid map M_{grid} with a certain resolution. Points from the point cloud map M_{pc} are assigned to corresponding grid cells, where terrain planes are extracted using the RANSAC method. If the point cloud within a grid cell is sufficiently dense and geometrically consistent, a terrain plane $P_{terr} : ax + by + cz + d = 0$ is extracted and stored. As shown in Fig. 4, grid cells are classified as traversable if the angle δ between the P_{terr}'s normal vector $\mathbf{n}_P$ and the vertical vector $\mathbf{r}_{ver}$ is below a pre-defined threshold ε_{vP}. Cells containing too many outliers or non-compliant planes are classified as non-traversable. Parallel processing accelerates this abstraction, and the resulting grid map provides spatial topological terrain information for robot path planning.

4.2 Terrain-informed hierarchical global planner

Although ground robots typically operate near the seemingly low-dimensional ground surfaces in 3D rescue environments, their centroid poses in the global coordinate frame reside in SE(3) space. In addition to the extra DOFs introduced by the ATR's flippers, global path generation for the rescue ATR is inherently a high-dimensional problem.

The combined effects of the high dimensionality of the robot's state, the complexity of 3D environments, and the workspace constraints of the robot collectively make it extremely challenging to generate feasible paths for ATR using a direct, monolithic global planning approach. To address these issues, we first decouple the ATR's flipper motion from the planning stage, instead relying on the robustness of subsequent controllers within the framework to achieve reliable flipper regulation and control. Meanwhile, we introduce a hierarchical global planner that employs a bi-level strategy to alleviate these inherent challenges effectively. Hierarchical strategies are commonly used to address large-scale planning problems. For example, TARE (Cao et al. 2021) generated a crude path based on the topological connectivity of the graph map and then refined this path within specific regions to meet the navigation requirements. However, such methods have to handle the connection between paths in different regions carefully; otherwise, gaps between path waypoints may occur, leading to fluctuations in path resolution that can negatively impact navigation.

In this paper, unlike traditional methods, the lower-level planner does not correct or optimize the crude path generated by the upper-level planner. Instead, the terrain information near the crude path was utilized to guide the lower-level planner's operation space.

As illustrated in Fig. 3, the upper-level planner quickly generates a crude path, ${}_C\Gamma_{base}$, using a graph-search method on the previously established terrain information grid map. At this level, the robot's orientation is temporarily disregarded, reducing the search space dimensionality and enabling fast, approximate path generation while preserving path-finding completeness. This path, derived from grid topology and connectivity, is neither smooth nor capable of passing feasibility checks in complex environments. Additionally, the inaccessible areas in large-scale rescue scenarios negatively impact the robustness of the graph search, leading to redundant global paths. However, the region ${}_C\Gamma_{base}$ traversed contains much feasible and valuable terrain information, which serves as a good prior for subsequent processing.

The lower-level planner utilizes a sampling-based planning (SBP) method to ensure path traversability while incorporating agnostic randomness. For ground robots engaged in rescue missions, most 3D space is non-traversable (e.g., regions far from the ground), with the robot's pose constrained near the terrain. The algorithm utilizes terrain information P_{terr} from the grid map M_{grid} near the crude path ${}_C\Gamma_{base}$ as priors for sampling, thereby limiting the sampled states to regions close to the ground while enhancing the completeness of global path generation with stochasticity. For a traversable terrain surface $P_{terr} : ax + by + cz + d = 0$ within the sampling region, the lower-level planner samples a random position $[x_\mathrm{r}, y_\mathrm{r}]$ in the x and y directions and a random yaw angle $\mathcal{Y}_\mathrm{r}$. Based on the plane equation,

an augmented random SE(3) pose can be directly computed on the terrain plane as $\mathbf{x}_\mathrm{r} = [x_\mathrm{r}, y_\mathrm{r}, z_\mathrm{r}, q_\mathrm{r}] \in \mathrm{SE}(3)$.

The dependent random position in the z direction is calculated as:

$$z_\mathrm{r} = -\frac{ax_\mathrm{r} + by_\mathrm{r} + d}{c}. \tag{1}$$

The dependent random orientation q_r is derived based on the projection of the direction vector $\mathbf{v}$ (determined by the sampled yaw angle $\mathcal{Y}_\mathrm{r}$) onto the terrain plane P_{terr}, as:

$$q_\mathrm{r} = \mathbf{Mat2Quat}([\mathbf{a}, \mathbf{b}, \mathbf{c}]), \tag{2}$$

where $\mathbf{v} = [\cos(\mathcal{Y}_\mathrm{r}), \sin(\mathcal{Y}_\mathrm{r}), 0]^\top$, $\mathbf{a} = \frac{\mathbf{v} - (\mathbf{v}\cdot\mathbf{c})\mathbf{c}}{\|\mathbf{v} - (\mathbf{v}\cdot\mathbf{c})\mathbf{c}\|}$, $\mathbf{b} = \mathbf{c} \times \mathbf{a}$, and $\mathbf{c} = \frac{\mathbf{n}}{\|\mathbf{n}\|}$. While $\mathbf{Mat2Quat}()$ denotes a transformation from a rotation matrix to a quaternion. This approach keeps the sampling pose near the likely feasible and useful terrains, reducing the sampling space and significantly decreasing the time required for feasibility checks during path generation.

The time consumption of sampling-based planning algorithms is mainly concentrated on the feasibility checks of the sampled states. In a 3D environment, a feasible state has to ensure that the ATR maintains stable contact with the ground, avoids collisions within the environment, and stays away from dangerous edges (e.g., cliffs). Our algorithm simplifies the robot model into a convex rectangular cuboid to reduce the time spent on collision detection. By combining this convex primitive with a kDTree that stores map information (Şucan et al. 2012), the efficiency of collision detection is significantly enhanced.

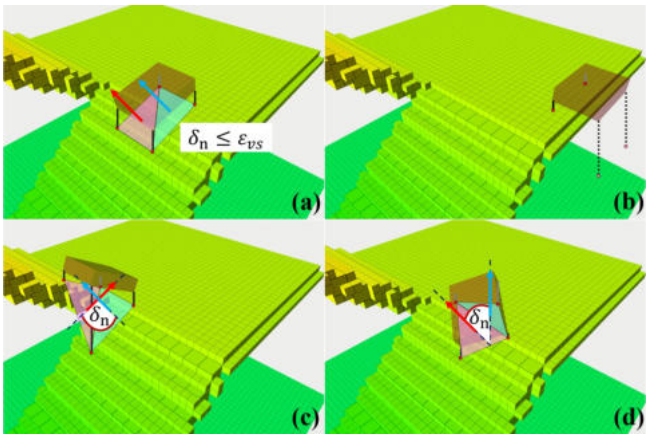


Figure 5. The typical stability evaluation cases for ground robots: (a) valid, (b) missing support, (c) uneven, (d) unstable.

Additionally, the stability can be determined by the projection of the robot's footprint on the ground. As shown in Fig. 5, since the simplified robot model is a rectangular cuboid, when its rectangular chassis footprint is vertically projected onto the ground, a 3D convex polygon is formed. Affected by the terrain, the polygon's vertices set can be expressed as $\mathcal{V} = \{\mathbf{v}_1, ..., \mathbf{v}_n\}$, where $\mathbf{v}_n := [x, y, z]$. If the number of polygon vertices n is less than 4 (as shown in Fig. 5(b)), the robot is at a dangerous edge that is not traversable. When the number of vertices equals 4, the 3D quadrilateral is then triangulated. If the maximum angle $\delta_\mathbf{n}$ between the normal vectors of any two divided triangles exceeds a certain threshold ε_{vs} (as shown in Figs. 5(c)-(d)), the contact between the robot and the ground at this location is considered unstable.

Furthermore, a divide-and-conquer approach can be employed to parallelize and accelerate the planning process in large-scale environments, as illustrated in Fig. 6. For long-range paths, the causal relationships between distant waypoints are typically weak. Therefore, the grids within the sampling region are grouped according to the graph-searched crude path. Then, perform sampling for feasible segment endpoints $\mathbf{x}_{seg}$ in the overlap areas between adjacent grouped grids. These endpoints serve as the start and goal points for the segmented paths, each of which is planned in parallel. Once all segmented paths are generated, they are stitched together to form the complete global path, ${}_G\Gamma_{base}$.

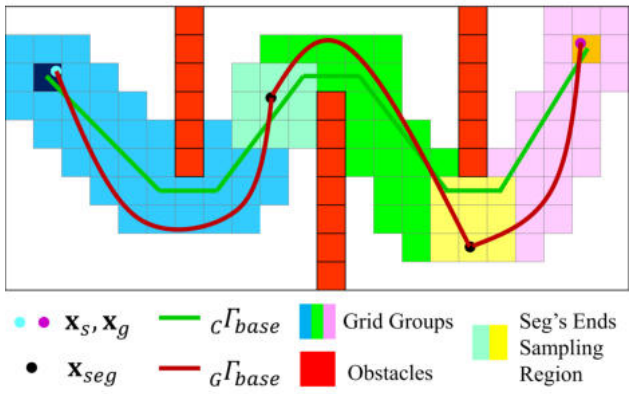


Figure 6. Hierarchy global planner with divide-and-conquer strategy.

Thus, based on a terrain information grid map, this framework introduces a novel and fast hierarchical global path generation method, further enhancing planning efficiency by adopting a divide-and-conquer strategy, ensuring that the robot can quickly generate a feasible path in complex 3D environments.

4.3 Manifold-optimization-based local planner

For ATR navigation in large-scale environments, global paths often lack the precision required to accommodate the robot's motion characteristics and adaptability to complex, fluctuating terrains. To ensure smooth, feasible, and terrain-adaptive local paths, this paper adopts a manifold-optimization-based local planner that exploits the terrain manifold to efficiently refine coarse global paths under various constraints.

4.3.1 Mathematical representation of SE(3)-path optimization.

For optimization-based 3D ground robot path planning algorithms, the basic form of the optimization

problem can be expressed as:

$$\begin{aligned}\mathbf{x}^* &= \arg\min_{\mathbf{x}\in \mathrm{SE}(3)} \mathbf{F}(\mathbf{x}), \\ &= \arg\min_{\mathbf{x}\in \mathrm{SE}(3)} \sum_{i=0:N-1, s\in\mathbb{S}} \mathbf{e}_s(\mathbf{x}_i, \mathcal{E}_s)^\top \mathbf{\Omega}_s \mathbf{e}_s(\mathbf{x}_i, \mathcal{E}_s),\end{aligned} \tag{3}$$

where $\mathbf{x} = (\mathbf{x}_0^\top, \ldots, \mathbf{x}_{N-1}^\top)^\top \in \mathbb{R}^{6N}$ represents the vector of all SE(3) poses in the path Γ_{base}; each waypoint is characterized by $\mathbf{x}_i := [x, y, z, \mathcal{R}, \mathcal{P}, \mathcal{Y}]$; the parameter i denotes the index of the waypoint; $\mathbf{e}_s$ denotes the cost function for various optimization objectives; $\mathcal{E}_s$ represents external elements related to the optimization objectives; the type of cost function is denoted by $s \in \mathbb{S}$, and $\mathbf{\Omega}_s$ indicates the weight of the corresponding cost function in the optimization process. For the application scenario of ATR in a rescue mission, the costs in SE(3) path optimization can be categorized into five types.

Kinematic Cost: The robot's mobility is constrained by its inherent kinematic limitations. The optimization has to incorporate a cost function that accounts for kinematics to produce a smooth optimized path. The kinematic model of a tracked chassis can be simplified to a differential drive model with nonholonomic motion constraints. The kinematic cost function, denoted as $\mathbf{e}_k(\mathbf{x}_i)$, can be expressed in the following form (Rösmann et al. 2017):

$$\mathbf{e}_k(\mathbf{x}_i) = \left\| \begin{bmatrix} \cos(\mathbf{x}_i^{\mathcal{Y}}) + \cos(\mathbf{x}_{i+1}^{\mathcal{Y}}) \\ \sin(\mathbf{x}_i^{\mathcal{Y}}) + \sin(\mathbf{x}_{i+1}^{\mathcal{Y}}) \\ 0 \end{bmatrix} \times d_{i,\,i+1} \right\|_2, \tag{4}$$

where $\mathbf{x}_i^{\mathcal{Y}}$ represents the yaw angle at waypoint i, and $d_{i,\,i+1} = [\mathbf{x}_{i+1}^x - \mathbf{x}_i^x, \mathbf{x}_{i+1}^y - \mathbf{x}_i^y, 0]^\top$ which represents the augmented XOY position difference between adjacent waypoints.

Obstacle Avoidance Cost: The robot has to maintain a safe clearance from obstacles $\mathcal{X}_{obs}$ to avoid collisions. For real-time and reliable path planning, onboard sensors continuously gather data to identify traversable areas and obstacles. Obstacles are represented as convex hulls, simplifying distance calculations while expanding the safety margin between the robot and the obstacles. Then, the cost function is defined as the reciprocal of the distance, formulated as follows:

$$\mathbf{e}_o(\mathbf{x}_i, \mathcal{X}_{obs}) = \frac{1}{d(\mathbf{x}_i, \mathcal{X}_{obs})} = \frac{1}{\|\mathbf{x}_i - \mathcal{X}_{obs}\|_2}. \tag{5}$$

Ground Adherence Cost: In a 3D environment, the waypoints $\mathbf{x}_i$ of a ground robot need to remain closely adhered to the ground $\mathcal{S}_{terr}$. The cost can be represented as $\mathbf{e}_t(\mathbf{x}_i, \mathcal{S}_{terr})$. Here, we use the 3D planes P_{terr} from the previously generated terrain information grid map M_{grid} as ground information in the cost function. As Eq. (6) shows, the cost function is defined based on the distance between the chassis's footprint and the ground, measured as the sum of distances from the four corner points $\mathbf{x}_{i,n}$ of the footprint to the ground, encouraging the path to adhere to the terrain.

$$\begin{aligned}\mathbf{e}_t(\mathbf{x}_i, \mathcal{S}_{terr}) &= d(\mathbf{x}_i, \mathcal{S}_{terr}) = \sum_n d(\mathbf{x}_{i,n}, P_{terr}) \\ &= \sum_n \frac{|a\mathbf{x}_{i,n}^x + b\mathbf{x}_{i,n}^y + c\mathbf{x}_{i,n}^z + d|}{\sqrt{a^2+b^2+c^2}},\end{aligned} \tag{6}$$

where a, b, c, d are the plane equation parameters of P_{terr}.

Pitch Stability Cost: In rugged terrain, maneuvering requires constraining the robot's pitch angle to ensure safety and stability. Moving along a straight path on excessively steep slopes may result in insufficient traction or cause the robot to slip. The robot should follow a winding path, prioritizing safety over path length to mitigate these risks. Thus, the method constrains the robot's pitch angle within a safe range of $[-\varepsilon^{\mathcal{P}}, \varepsilon^{\mathcal{P}}]$. The nonlinear cost function is formulated as follows:

$$\mathbf{e}_{\mathcal{P}}(\mathbf{x}_i) = \begin{cases} 0, & -\varepsilon^{\mathcal{P}} \le \mathbf{x}_i^{\mathcal{P}} \le \varepsilon^{\mathcal{P}} \\ \mathbf{x}_i^{\mathcal{P}} - \varepsilon^{\mathcal{P}}, & \mathbf{x}_i^{\mathcal{P}} > \varepsilon^{\mathcal{P}} \\ -\varepsilon^{\mathcal{P}} - \mathbf{x}_i^{\mathcal{P}}, & \mathbf{x}_i^{\mathcal{P}} < -\varepsilon^{\mathcal{P}}. \end{cases} \tag{7}$$

Anisotropic Traversability Cost: In most cases, a robot's maneuverability on normal terrain is consistent and isotropic. However, stair-like terrain, commonly found in buildings, introduces anisotropy in traversability. Empirical evidence indicates that tracked robots exhibit optimal performance when traversing perpendicularly to the edges of stairs. Because the chassis's sinuous motion would make it difficult for the tracks to overcome the structural pressure at stair edges, and the chassis may be lifted by the edges, resulting in a temporary loss of maneuverability. Thus, we define an anisotropic traversability cost $\mathbf{e}_a(\mathbf{x}_i, \mathcal{S}_{terr})$ based on terrain characteristics, to ensure the maximum maneuverability of the ATR on stairs while maintaining safety.

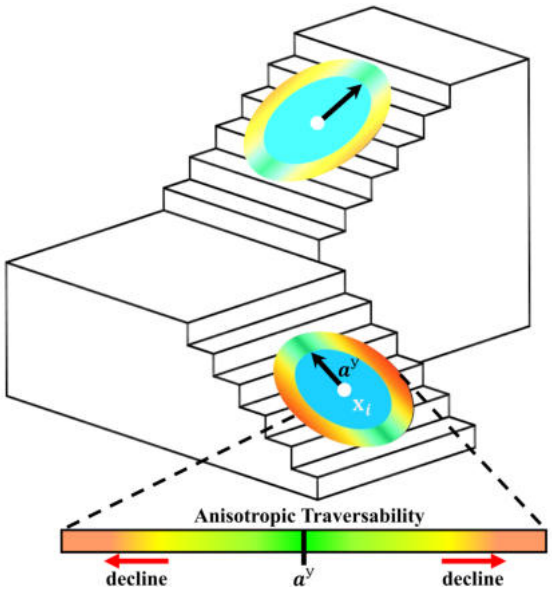


Figure 7. The anisotropic traversability on the staircases.

The staircases' pose and the steps' geometric attributes can be precisely identified using LiDAR

point clouds or RGB-D images (Perez-Yus et al. 2017; Westfechtel et al. 2018; Sriganesh et al. 2023). In this framework, terrain information in the grid map is updated and semantically annotated in real-time. If the traversability of the terrain P_{terr} in the grid is anisotropic, the direction $a^{\mathcal{Y}}$ with the highest traversability (e.g., the direction perpendicular to the stair treads, as shown in Fig. 7) is selected for anisotropic traversability labeling of the plane P_{terr} in that grid cell under the global frame F_G. During path optimization, anisotropic traversability information is queried based on the grid containing the waypoint $\mathbf{x}_i$. The cost function is expressed as:

$$\mathbf{e}_a(\mathbf{x}_i, \mathcal{S}_{terr}) = |\sin(\mathbf{x}_i^{\mathcal{Y}} - a^{\mathcal{Y}})|, \tag{8}$$

to minimize the angular difference between the waypoint's yaw angle $\mathbf{x}_i^{\mathcal{Y}}$ and the anisotropic traversability direction $a^{\mathcal{Y}}$ while maximizing the robot's traversability at each waypoint, ensuring that the pose of the waypoints satisfies the conditions for optimal maneuverability.

4.3.2 *Mathematical representation of path optimization on the terrain manifold.*

Multi-objective SE(3)-path optimization may fail due to disturbances in certain optimization directions. For instance, the optimized path might deviate from the ground to avoid obstacles, failing to meet ground adherence. This issue arises because the ground adherence cost is a soft constraint in the optimization problem (3), and significantly increasing its cost weight does not guarantee path feasibility and may lead to a weight imbalance in the optimization problem. To address this, we propose a path optimization method based on terrain manifolds to integrate the ground adherence cost as an implicit hard constraint within the manifold.

Considering the ground adherence, the waypoints can be bijectively transformed between the terrain manifold space and SE(3) space within a small range. The bijective transformation allows the path optimization problem in SE(3) to be mapped to a path optimization problem on the terrain manifold.

On manifold $\mathcal{M}$, for a cost function $\mathbf{F} : \mathcal{M} \to \mathbb{R}$, the basic form of the manifold optimization problem is given as follows:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathcal{M}} \mathbf{F}(\mathbf{x}). \tag{9}$$

The optimization process on a manifold has to consider the gradient of the cost function on that manifold. If $\mathbf{F}$ is smooth on the manifold $\mathcal{M}$, the Riemannian gradient $\text{grad}_{\mathbf{x}}\mathbf{F}$ at point $\mathbf{x}$ is the projection of the Euclidean gradient $\nabla\mathbf{F}(\mathbf{x})$ onto the tangent space $T_{\mathbf{x}}\mathcal{M}$, denoted as $\text{grad}_{\mathbf{x}}\mathbf{F} = \text{Proj}_{\mathbf{x}}\nabla\mathbf{F}(\mathbf{x})$.

Optimizing a waypoint $\mathbf{x}$ on a manifold requires updating the point in the descent direction of the cost function while ensuring it stays on the manifold. Thus, after obtaining the descent direction $\mathbf{v}$, the algorithm uses a retraction map $R_{\mathbf{x}}(\mathbf{v})$ to update the waypoint $\mathbf{x}$ on the manifold. As illustrated in Fig. 8, the retraction map $R_{\mathbf{x}}(\mathbf{v}) : T_{\mathbf{x}}\mathcal{M} \to \mathcal{M}$ not only ensures continuity at point $\mathbf{x}$ but also satisfies the condition $R_{\mathbf{x}}(\mathbf{0}) = \mathbf{x}$. Analogous to the gradient descent method in Euclidean space, the update formula for gradient descent on a manifold is given by:

$$\mathbf{x}_{k+1} = R_{\mathbf{x}_k}(-t_k \text{grad}_{\mathbf{x}_k}\mathbf{F}), \tag{10}$$

where t_k represents the step size, and $R_{\mathbf{x}_k}$ denotes the retraction map at the point $\mathbf{x}_k$.

Optimization on manifold extensively utilizes the tangent space to perform iterations (Boumal 2023). The manifold used here is derived from the terrain planes P_{terr} stored in the grid map M_{grid}. This approach ensures that tangent spaces are identical across all points on a manifold, thereby facilitating the calculation of the Riemannian gradients $\text{grad}_{\mathbf{x}}\mathbf{F}$ and allowing for direct and efficient cost computation on the manifold. It also simplifies the mapping and inverse mapping between spaces, thereby improving optimization efficiency.

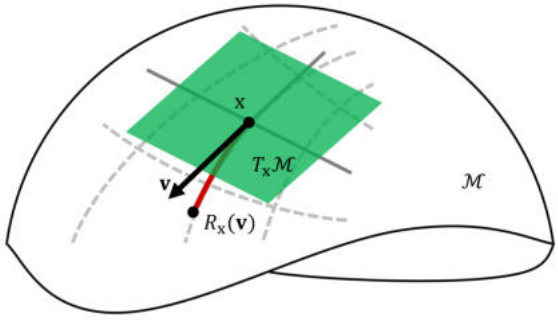


Figure 8. The retraction map $R_{\mathbf{x}}(\mathbf{v})$ on point $\mathbf{x}$ on manifold $\mathcal{M}$.

For the local planning, the information of the waypoint $\mathbf{z}_i$ on the terrain manifold is defined as $[x, y, \mathcal{Y}]$ due to the mapping relationship $\mathbf{z}_i = \mathbf{Proj}(\mathbf{x}_i, P_{terr})$, which reduces the dimensionality of the original path. Consequently, the path on the terrain manifold is defined as $\mathbf{z} = (\mathbf{z}_0^\top, \ldots, \mathbf{z}_{N-1}^\top)^\top \in \mathbb{R}^{3N}$, where N represents the number of waypoints.

For path optimization on the terrain manifold, the problem (3) is transformed into:

$$\begin{aligned}\mathbf{z}^* &= \arg\min_{\mathbf{z}\in\mathcal{M}} \mathbf{F}(\mathbf{z}), \\ &= \arg\min_{\mathbf{z}\in\mathcal{M}} \sum_{i=0:N-1, s\in\mathbb{S}} \mathbf{e}_s(\mathbf{z}_i, \mathcal{E}_s)^\top \mathbf{\Omega}_s \mathbf{e}_s(\mathbf{z}_i, \mathcal{E}_s),\end{aligned} \tag{11}$$

where the kinematic, obstacle avoidance, and anisotropic traversability costs remain unchanged and still act as soft constraints. In contrast, compared to problem (3), the ground adherence cost becomes a hard constraint due to the introduction of the manifold in problem (11). This hard constraint ensures that the robot's pose is strictly confined to the ground, preventing poor optimization directions. Considering the mathematical form of P_{terr} and the inverse mapping relationship of the manifold, the pitch angle of

a waypoint on the terrain manifold can be expressed as $\mathbf{z}_i^{\bar{P}} = \text{atan}(\frac{a+b\tan(\mathbf{z}_i^{\mathcal{Y}})}{c\sqrt{1+\tan^2(\mathbf{z}_i^{\mathcal{Y}})}})$. The basic form of the pitch stability cost also remains unchanged.

The manifold optimization (ManiOpt) resolves the issue of the path deviation from the traversable ground while significantly receding the dimension of the optimization variables and further reduces the cost type $\mathbb{S}$ in the optimization problem. This approach avoids the weight imbalance problem while improving the optimization process's efficiency.

The efficiency of ManiOpt helps prevent overshooting in path-tracking control that may occur due to delayed reference updates. Moreover, the optimized paths generated by ManiOpt reduce the risk of environmental misinterpretation during execution. For instance, ManiOpt mitigates the likelihood of holistic control failures induced by obstacle points in terrain analysis. In addition, ManiOpt's capability to incorporate terrain-specific optimization objectives in particular scenarios further decreases the complexity of ATR control. Finally, as the optimized paths remain close to the terrain surface, these paths provide high-quality information for terrain-aware motion control, thereby minimizing discrepancies between the reference terrain point cloud and the actual terrain.

4.4 On-manifold model predictive controller for ATR

The ATR should be able to adjust the flipper states adaptively, ensuring that the body adheres to the terrain and lowers its centroid to achieve stable maneuvering and better stress distribution in complex environments, thereby reducing the risk of malfunctions such as overturning or becoming trapped (Okada et al. 2009; Chen et al. 2023). Existing research often treats the flippers as a special mechanism of the chassis. After determining the centroid-joint space path for the next period, the robot utilizes a decoupled feedback control method to manage the tracks' and flippers' motion. However, suppose the flipper actuators do not perform as expected. In that case, the planned joint space path may not be accurately followed, potentially leading to terrain-configuration mismatches that could place the robot in an unsafe operational condition.

The primary concern for ATR navigation in disaster rescue missions is whether the robot can reach its assigned target locations. Inspired by some well-established navigation frameworks (Sleiman et al. 2023), the method proposed in this paper emphasizes mission-oriented whole-body control. To address the challenge of coordinated and stable ATR motion on rugged terrain, we propose an on-manifold model predictive controller (M^{2}PC) that integrates terrain information with the robot's whole-body kinematics. This controller enables the ATR's chassis to follow the optimized local centroid path holistically and stably, without requiring an explicitly organized flipper joint space path.

Therefore, M^{2}PC not only enhances whole-body control safety but also significantly reduces the dimensional burden on the planning phases within the framework. On the one hand, M^{2}PC eliminates the need for the global planner to sample flipper's joint states or perform complex feasibility checks during initial path generation. On the other hand, M^{2}PC allows the local planner to optimize paths without considering optimization terms related to the flippers, thus reducing computational costs.

As illustrated in Fig. 9, the core conception of M^{2}PC involves mathematically abstracting the terrain into a continuously differentiable manifold and then leveraging the manifold's additive properties to couple the terrain manifold with the robot's motion manifold. The coupled manifold acts like a system model that guides the model predictive control of ATRs navigating rugged terrain. Within this manifold, the proposed controller adaptively coordinates the robot's whole-body motion under terrain and robot constraints, significantly enhancing safe maneuverability in extreme environments.

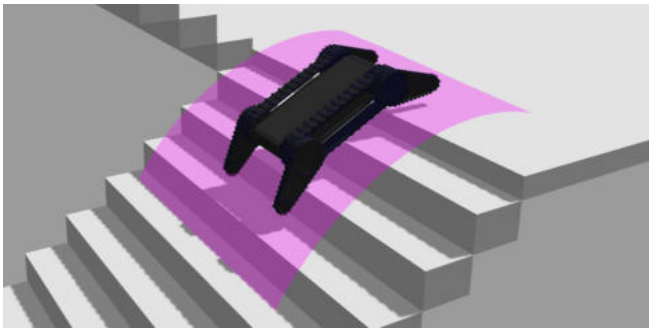


Figure 9. The holistic motion of the ATR on the manifold.

The manifold $\mathcal{M}$ is an n-dimensional topological space where each point's neighborhood is homeomorphic to an open subset of Euclidean space $\mathbb{R}^n$. And local coordinate charts made the mapping $\phi_x : \mathcal{M} \to \mathbb{R}^n$ and the inverse mapping $\phi_x^{-1} : \mathbb{R}^n \to \mathcal{M}$ between $\mathcal{M}$ and $\mathbb{R}^n$ possible. Based on the premise mentioned above, the robot system model on the manifold can be expressed as follows:

$$\mathbf{x}_{k+1} = \mathbf{x}_k \oplus \Delta t \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right), \tag{12}$$

where $\mathbf{x} \in \mathcal{M}$ represents the system state (the robot configurations on the manifold), $\mathbf{u} \in \mathbb{R}^m$ represents the control input (the chassis longitudinal and rotational velocities, as well as the flippers' rotational velocities), and k is the discrete time step. The function $\mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right) \in \mathbb{R}^l$ denotes the perturbation induced by $\mathbf{x}_k$ and $\mathbf{u}_k$ at time k, and $\oplus : \mathcal{M} \times \mathbb{R}^l \to \mathcal{M}$ is the addition operation, which adds the state perturbation in space $\mathbb{R}^l$ to manifold $\mathcal{M}$.

For the optimization on the manifold, the operation $\boxminus : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^n$ is introduced to compute the difference on the manifold $\mathcal{M}$ in its homeomorphic tangent space $\mathbb{R}^n$. Thus, the optimization problem corresponding to the on-manifold model predictive

control can be formulated as follows:

$$\begin{aligned}
&\min_{\mathbf{u}_k} \sum_{k=0}^{N-1} \left(\left\| \mathbf{x}_k \boxminus \mathbf{x}_k^d \right\|_{\mathbf{Q}_k}^2 + \left\| \mathbf{u}_k - \mathbf{u}_k^d \right\|_{\mathbf{R}_k}^2 \right) + \left\| \mathbf{x}_N \boxminus \mathbf{x}_N^d \right\|_{\mathbf{Q}_N}^2, \\
&\text{s.t. } \mathbf{x}_{k+1} = \mathbf{x}_k \oplus \Delta t \mathbf{f}\left(\mathbf{x}_k, \mathbf{u}_k\right), \quad \mathbf{x}_0 = \mathbf{x}_{\text{int}}, \\
&\quad \mathbf{u}_k \in \mathbf{U}, \mathbf{x}_k \in \mathbf{X}, \quad k = 0, \ldots, N-1,
\end{aligned} \tag{13}$$

where $\mathbf{Q}_k$, $\mathbf{R}_k$, and $\mathbf{Q}_N$ are the matrices representing the penalty weights for the state at time step k, the control input at time step k, and the terminal state, respectively; the set $\mathbf{U} = \{\mathbf{u} \in \mathbb{R}^m \mid \mathbf{u}_{min} \leq \mathbf{u} \leq \mathbf{u}_{max}\}$ defines the bounds on the control inputs; N denotes the length of the control horizon for the controller.

The optimization problem (13) is constrained by both variable bounds and the system model defined on the manifold, where the current state depends on the previous state and control inputs' projection on the homeomorphic tangent space of the manifold (Kalabić et al. 2017; Lu et al. 2023). As the robot's configuration space definition shapes the system model (12), we propose a system model incorporating terrain manifold curvature and robot mechanism to keep the ATR adhered to the ground.

The ground point clouds in the local submap are utilized for terrain manifold abstraction, and the surfaces are fitted near each path waypoint, with a quadratic form of $\mathcal{Z}_{terr} = \{(x, y, z) \in \mathbb{R}^3 \mid z = F(x, y) = [x, y]\mathbf{A}[x, y]^\top + \mathbf{B}^\top [x, y]^\top + \mathbf{C}\}$. According to the invariance theorem of surfaces (Do Carmo 2016) (the shape characteristics of a surface are the same in different coordinate systems), the frame of the surface is aligned with the global frame F_G. Given that the quadratic formed surface information, the first and second partial derivatives of the surface at the path waypoints $[x, y, \mathcal{Y}]$ can be obtained as $F_x, F_y, F_{xy}, F_{xx}, F_{yy}$. Consequently, the normal curvature K of the surface can be expressed as the ratio of the second fundamental form to the first fundamental form as follows:

$$K = \frac{\mathbf{II}\mathbf{T}}{\mathbf{I}\mathbf{T}}, \tag{14}$$

where,

$$\begin{aligned}
\mathbf{I} &= [\mathbf{E} \quad 2\mathbf{F} \quad \mathbf{G}] = \left[F_x^2 + 1 \quad 2F_xF_y \quad F_y^2 + 1\right], \\
\mathbf{II} &= [\mathbf{L} \quad 2\mathbf{M} \quad \mathbf{N}] = [F_{xx} \quad 2F_{xy} \quad F_{yy}] / \sqrt{1 + F_x^2 + F_y^2}, \\
\mathbf{T} &= \left[\cos^2(\mathcal{Y}) \quad \cos(\mathcal{Y})\sin(\mathcal{Y}) \quad \sin^2(\mathcal{Y})\right]^\top.
\end{aligned} \tag{15}$$

We emphasize that the safe path tracking of the ATR's centroid is the primary focus for maneuvering over rugged terrain, with the flipper state adjusted through adaptive coordination control to regulate the ATR's configuration. In the proposed method, the desired flipper state depends not only on the characteristics of the terrain manifold but also on the ATR's motion. Conversely, the alignment between the ATR's configuration and the terrain manifold should influence the ATR's motion to prevent reckless maneuvers in unstable configurations.

As illustrated in Fig. 10, the geometric relationship shows that the arc fitted to the ATR's configuration

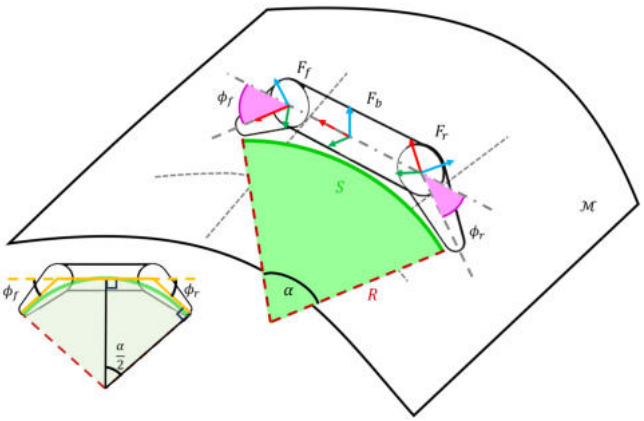


Figure 10. The geometrical relationship between the ATR configuration and the terrain manifold normal curvature.

when maneuvering over rugged terrain should continuously align with the ground's normal curvature to ensure the ATR adapts effectively to terrain variations. To ensure the alignment, the joint angles of the front and rear flippers in the chassis frame should be $\phi_i = \frac{1}{2}\alpha \approx \frac{1}{2}KS$. The robot's total span length S is approximately equal to the arc enveloped by the ATR's configuration, and K is the normal curvature of the manifold at the waypoint. According to Eqs. (14)-(15) and the invariance theorem of surfaces, the normal curvature of the surface near the ATR depends solely on its centroid's pose. The state variables for the flippers are defined as the differences between the actual and desired flipper joint angles in the chassis frame $\epsilon_i, i \in \{f, r\}$, establishing the relationship between the ATR's centroid pose and the desired configuration for coordinated control. The differences are intended to be maintained at zero in the form of:

$$\epsilon_i = \phi_i - \frac{1}{2}KS \equiv 0, \quad i \in \{f, r\}. \tag{16}$$

Since the robot is constrained to the terrain surface, not only does its centroid position adhere to the terrain manifold, but the rotational axis of its body frame also aligns with the normal vector at that position of the terrain manifold. Thus, the robot's chassis position and orientation on the terrain manifold can be represented by $\mathbf{p} \in \mathcal{Z}_{terr}$ and $\mathbf{R} \in \mathrm{SO}(2)$.

In summary, the state variables and control variables of the M^{2}PC are set as follows:

$$\begin{aligned}
&\mathcal{M} = \mathcal{Z}_{terr} \times \mathrm{SO}(2) \times \mathbb{R}^1 \times \mathbb{R}^1, \quad \dim(\mathcal{M}) = 5, \\
&\mathbf{x} = [\mathbf{p} \quad \mathbf{R} \quad \epsilon_f \quad \epsilon_r] \in \mathcal{M}, \\
&\mathbf{u} = [v_x \quad \omega_z \quad \psi_f \quad \psi_r] \in \mathbb{R}^4,
\end{aligned} \tag{17}$$

where v_x and ω_z are the longitudinal and rotational velocities of the robot chassis; ψ_f and ψ_r are the front and rear flipper joint velocities, respectively.

The robot's chassis motion model can be represented by (Lu et al. 2023):

$$\dot{\mathbf{p}} = \alpha \mathbf{R}\mathbf{e}_1 v_x, \quad \dot{\mathbf{R}} = \beta \omega_z, \tag{18}$$

where

$$\alpha(\mathbf{p},\mathbf{R}) = \frac{1}{\sqrt{1+(\mathbf{h}^\top \mathbf{R}\mathbf{e}_1)^2}},$$
$$\beta(\mathbf{p}) = \frac{1}{\sqrt{1+\mathbf{h}^\top\mathbf{h}}}, \tag{19}$$
$$\mathbf{h} = [F_x \quad F_y]^\top, \mathbf{e}_1 = [1 \quad 0]^\top, \text{and } \mathbf{e}_2 = [0 \quad 1]^\top.$$

And the motion relationship between $\dot{\epsilon}_i$ and the control inputs $\mathbf{u}$ can be represented as follow:

$$\dot{\epsilon}_i = \psi_i - (c_v K'_{\mathbf{p}} \alpha \mathbf{R}\mathbf{e}_1 v_x + c_\omega K'_{\mathbf{R}} \beta \omega_z) S, i \in \{f, r\}, \tag{20}$$

where $K'_{\mathbf{p}}$ and $K'_{\mathbf{R}}$ are the the partial derivatives of the normal curvature K with respect to $\mathbf{p}$ and $\mathbf{R}$. And $K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x$ and $K'_{\mathbf{R}} S\beta\omega_z$ represent the effects of the robot's longitudinal velocity and rotational angular rate on the desired flipper joint angles, respectively. Two scale factors c_v and c_ω are introduced to regulate the effect of the robot chassis actions v_x and ω_z on the model, then to enhance the stability of the control method.

The state transition function for the M^{2}PC problem (13) is formulated as follows:

$$\mathbf{f}(\mathbf{x},\mathbf{u}) = \begin{bmatrix} \alpha\mathbf{R}\mathbf{e}_1 v_x \\ \beta\omega_z \\ \psi_f - c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x - c_\omega K'_{\mathbf{R}} S\beta\omega_z \\ \psi_r - c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x - c_\omega K'_{\mathbf{R}} S\beta\omega_z \end{bmatrix} \in \mathbb{R}^5. \tag{21}$$

The QP problem (13) is addressed using the OSQP (Stellato et al. 2020) open-source solver. In this problem, the linear constraint matrix is derived from the differential composition of the state transition function (21). Specifically, the expressions for the partial derivatives $\frac{\partial \mathbf{f}(\mathbf{x}\boxplus\delta\mathbf{x},\mathbf{u})}{\partial\delta\mathbf{x}}$ and $\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u}\boxplus\delta\mathbf{u})}{\partial\delta\mathbf{u}}$ are calculated within the homeomorphic tangent spaces of the manifold $\mathcal{M}$, as detailed in Eq. (22) and Eq. (23):

$$\frac{\partial \mathbf{f}(\mathbf{x}\boxplus\delta\mathbf{x},\mathbf{u})}{\partial\delta\mathbf{x}} = \begin{bmatrix} \mathbf{R}\mathbf{e}_1 v_x \frac{\partial\alpha}{\partial\delta\mathbf{p}} & \mathbf{R}\left(\frac{\partial\alpha}{\partial\delta\mathbf{R}}\mathbf{e}_1 + \alpha\mathbf{e}_2\right) v_x & \mathbf{0}_{2\times2} \\ \frac{\partial\beta}{\partial\delta\mathbf{p}}\omega_z & 0 & \mathbf{0}_{1\times2} \\ -(c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x + c_\omega K'_{\mathbf{R}} S\beta\omega_z)|'_{\mathbf{p}} & -(c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x + c_\omega K'_{\mathbf{R}} S\beta\omega_z)|'_{\mathbf{R}} & \mathbf{0}_{1\times2} \\ -(c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x + c_\omega K'_{\mathbf{R}} S\beta\omega_z)|'_{\mathbf{p}} & -(c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 v_x + c_\omega K'_{\mathbf{R}} S\beta\omega_z)|'_{\mathbf{R}} & \mathbf{0}_{1\times2} \end{bmatrix} \in \mathbb{R}^{5\times5}, \tag{22}$$

$$\frac{\partial \mathbf{f}(\mathbf{x},\mathbf{u}\boxplus\delta\mathbf{u})}{\partial\delta\mathbf{u}} = \begin{bmatrix} \alpha\mathbf{R}\mathbf{e}_1 & \mathbf{0}_{2\times1} & 0 & 0 \\ 0 & \beta & 0 & 0 \\ -c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 & -c_\omega K'_{\mathbf{R}} S\beta & 1 & 0 \\ -c_v K'_{\mathbf{p}} S\alpha\mathbf{R}\mathbf{e}_1 & -c_\omega K'_{\mathbf{R}} S\beta & 0 & 1 \end{bmatrix} \in \mathbb{R}^{5\times4}. \tag{23}$$

5 Simulation and Experiment

5.1 Preparation

We validated and quantified the framework's effectiveness by deploying the algorithms on our self-developed articulated tracked robot, **Slugcat**, as shown in Fig. 11. Slugcat features coaxial front and rear articulated flippers, with flipper lengths of approximately 350mm and a chassis length of about 600mm. The total weight of the robot is around 55.7kg. The articulated flipper joints are driven by high-reduction motors, providing strong self-supporting capability and resistance to significant external disturbances. Three MID-360 LiDARs are mounted on the robot for terrain perception and localization. The onboard computer has an Intel-i7-11700K CPU (8 cores, 3.60GHz) and an RTX-4060 GPU with 8GB of VRAM. We also built various simulation environments based on the Gazebo physics engine, including structured building environments, unstructured outdoor scenes, and ideal experimental setups. The simulated articulated tracked robot matches the real robot regarding dimensions, kinematics, and dynamic characteristics (with a realistic drive system adapted from the work of Okada et al. (2020)).

Although the proposed framework is compatible with online point cloud maps constructed by the real-time SLAM method, for rigorous comparative evaluations, we employed prior point cloud maps in all simulations (Section 5.5) and in several real-world experiments (Section 5.6.1 to Section 5.6.4). In addition, we validated the framework's performance using online point cloud maps generated by the SLAM module in an unknown environment (Section 5.6.5). In the simulations, the robot's poses were directly provided by Gazebo. In all real-world experiments, the

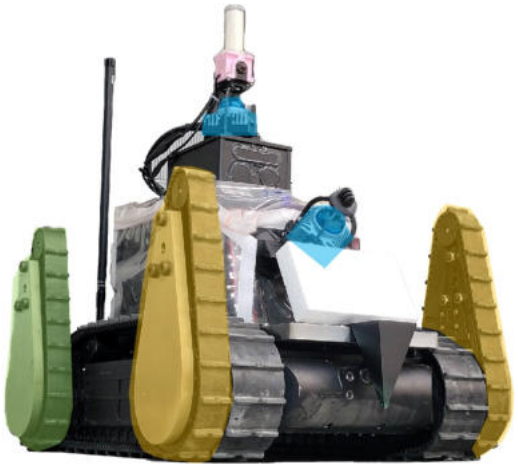
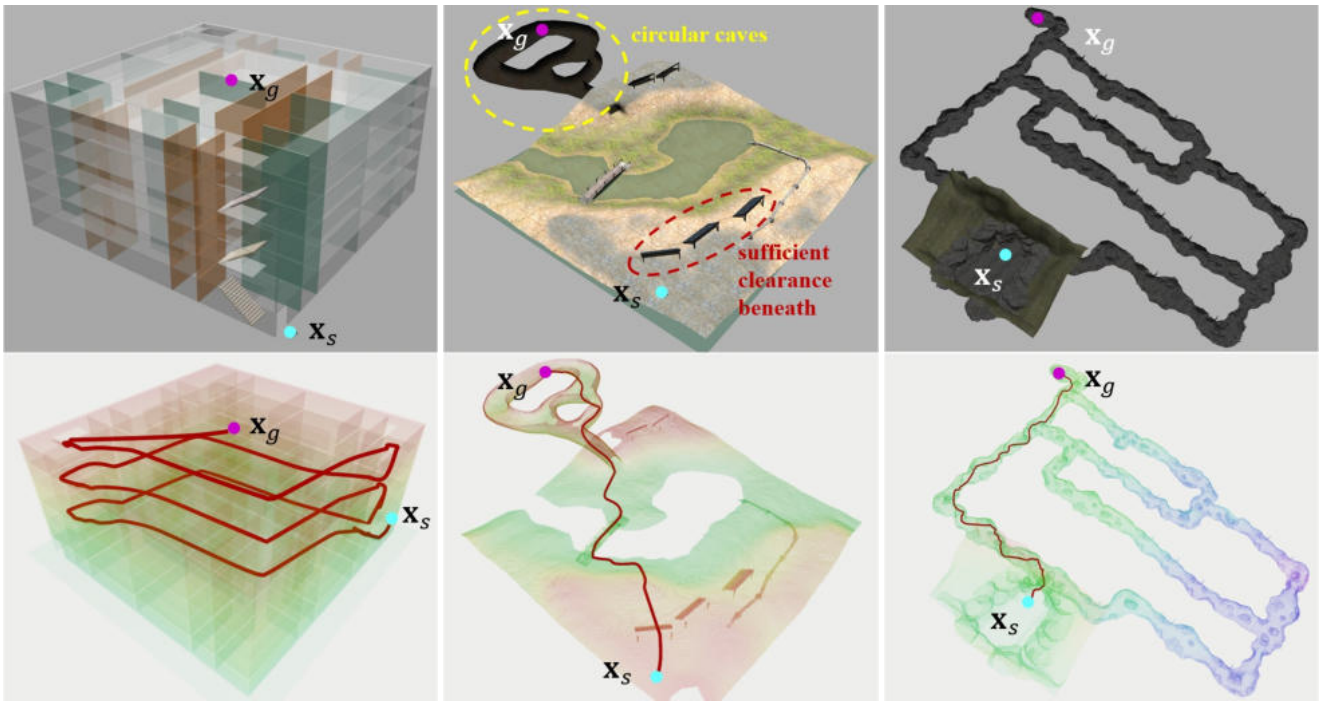


Figure 11. Slugcat: a swift articulated tracked robot for rescue missions. The orange sections represent the coaxial front flippers, the green sections represent the coaxial rear flippers, and the blue sections indicate the LiDARs.



(a) Restricted six-story building scenario **(b)** Complex wild scenario **(c)** DARPA_cave_02

Figure 12. The typical path generation results of the hierarchical global planner across various Gazebo-simulated scenarios. In the restricted building scenario shown in (a), there is only one possible path to the rooftop. In the wild scenario shown in (b), the optimal route requires crossing a narrow bridge. In the DARPA cave scenario (c), the start and goal points are separated by a considerable distance.

Table 1. Quantifications of different global path generation algorithms in the simulations: L_p denotes the average generated path length, $L_p.\sigma^2$ denotes the variance of the generated path length, and T_p denotes the average planning time. (In the DARPA_cave_02 scenario, our method reduced 78.81% of the time cost compared to the SOTA methods.)

	Six-story building			Restricted six-story building			Wild			DARPA_cave_02		
	L_p(m)	$L_p.\sigma^2$(m^2)	T_p(s)	L_p(m)	$L_p.\sigma^2$(m^2)	T_p(s)	L_p(m)	$L_p.\sigma^2$(m^2)	T_p(s)	L_p(m)	$L_p.\sigma^2$(m^2)	T_p(s)
CT-DC	63.34	4.02	**0.23**	**232.24**	92.40	**0.25**	135.81	19.45	**0.04**	**415.68**	44.91	**4.04**
Colas'	**50.93**	**0.0**	1.80	313.27	**0.0**	9.75	**111.34**	**0.0**	0.26	464.42	**0.0**	19.07
Krüsi's	97.68	753.28	3.83	294.99	113.42	23.59	157.29	212.08	0.15	626.38	358.23	22.66
PUTN	–	–	–	–	–	–	160.58	159.41	0.11	–	–	–

robot's odometry was provided by a SLAM module equipped with a re-localization function. Specifically, for the discrete terrain information abstraction module deployed throughout the work, the resolution of the 3D terrain-info grid map was uniformly set to 0.5m (considering the robot's physical size). The threshold for planar traversability determination, ε_{vP}, was set to 45°. Related experimental videos are available in the supplementary materials.

5.2 Validation of the hierarchical global planner

As shown in Figs. 12(a)-(c), we conducted efficiency evaluations in large-scale 3D simulation environments to validate the effectiveness of the hierarchical global planner, with consistent start and goal positions for every scenario, respectively. In this work, the maximum angular deviation ε_{vs} between the normal vectors of triangulated planes within a feasible ATR footprint was set to 15° (see Section 4.2, Fig. 5).

The six-story building depicted in Fig. 12(a) had a floor height of 2.5m, a floor area of 900m^2 per level, and a total volume of 1.35×10^4m^3. There were continuous staircases at the corners of the building. The start point was located at one corner outside the building, with global coordinates $(-16.0\text{m}, 14.0\text{m}, 0.1\text{m})$, while the goal point was at the center of the rooftop, with global coordinates $(0.0\text{m}, 0.0\text{m}, 15.1\text{m})$. The shortest feasible path length was approximately 60m. Additionally, the possible path solution was restricted by setting only two staircases for each floor in the opposite corners of the building, significantly increasing the length of the traversable path (with the shortest feasible path length of approximately 250m). The restricted six-story building scenario was also used to evaluate the adaptability of the proposed framework to various sampling-based planning algorithms in long-range planning scenarios.

The simulated wild scenario included ponds, mountainous areas, a narrow bridge, and mine caves. The scene depicted in Fig. 12(b) covered an area of approximately 1.0×10^4m^2. The start point was located behind the solar panel, with global coordinates $(-19.0\text{m}, 58.0\text{m}, 3.0\text{m})$, and the goal point was located in the cave at global coordinates $(30.0\text{m}, -10.0\text{m}, 4.0\text{m})$.

The shortest feasible path length was approximately 120m.

The scenario in Fig. 12(c) was adapted from DARPA's publicly available simulation scene, *DARPA_cave_02*. This scenario consisted of a massive underground cave, with the size of approximately $300\text{m} \times 300\text{m} \times 70\text{m}$, which occupied a volume of $6.3 \times 10^6\text{m}^3$. The start point was located in the open area outside the cave, with global coordinates $(0.0\text{m}, 0.0\text{m}, 0.0\text{m})$, and the goal point was located in the mine at global coordinates $(200.2\text{m}, -36.5\text{m}, -19.8\text{m})$, with the shortest feasible path length being approximately 450m. The large-scale and complex connectivity of the simulation environments presented significant challenges for the planning task. During the experiments, the upper-level planner employed the A* algorithm with a heuristic function based on Euclidean distance. The lower-level planner employed the RRT algorithm.

We compared our method with some existing works (Colas et al. 2013; Krüsi et al. 2017; Jian et al. 2022) to verify the homotopy, consistency, and efficiency of the path generation process. Each method was tested 25 times under identical conditions, and various metrics were calculated from the 20 best results, as shown in Table 1. The generated typical paths in the wild scenario are illustrated in Fig. 13.

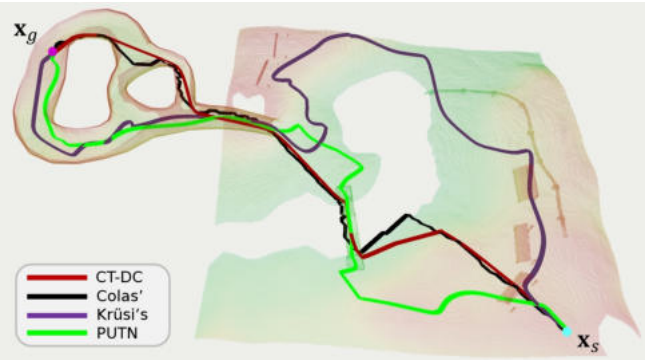


Figure 13. The typical path generation performance in the wild scenario with the proposed and the existing SOTA methods.

Colas et al. (2013) constructed a topological map for graph-search planning by identifying feasible points and connectivity in raw point clouds. Thus, its performance is highly sensitive to the scale of the environment, and the resulting paths are often rugged due to limited terrain details. Nevertheless, the deterministic nature of graph-search methods results in lower variance in path lengths, providing more consistent planning outcomes. Krüsi et al. (2017)'s work achieved smooth paths through post-processing and optimization; however, the initial solution had a significant impact on path length and trends, leading to greater variance and extended planning times, which would negatively affect execution efficiency. PUTN (Jian et al. 2022) increased the sampling step size in the SBP and used a 2D space sampler to enhance planning efficiency. However, due to the non-uniqueness of restoring 2D poses into 3D space, it cannot effectively handle path planning in multi-floor building structures. Our method (CT-DC) relies on well-organized terrain information, ensuring consistency and completeness from the upper-level planner as well as efficiency and randomness from the lower-level planner. The generated paths exhibit strong homotopy across multiple tests while maintaining superior performance in large-scale and complex environments.

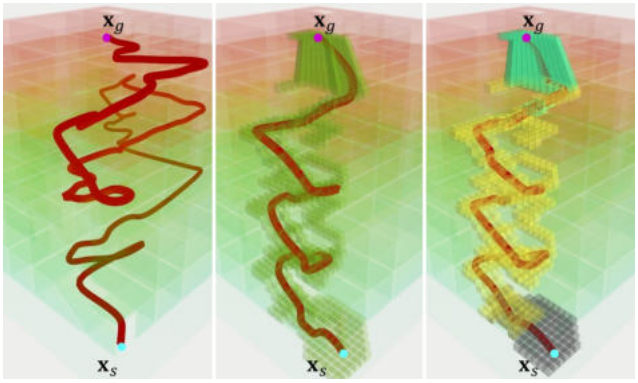


Figure 14. The typical path generation results in the six-story building with different strategies (from left to right: AT, CT, CT-DC).

As illustrated in Fig. 14, when the lower-level planner sampled all terrain information (AT) within the map, the planning process suffered from inefficiency (planning time up to 29.11s) and path redundancy. In contrast, the proposed method, which sampled only the terrain information near the crude plan (CT), significantly reduced average planning time to 0.51s by limiting the randomness of SBP. Moreover, the divide-and-conquer strategy (DC) further accelerated path generation without altering the sampling range, reducing the average planning time to 0.27s (see Table 2).

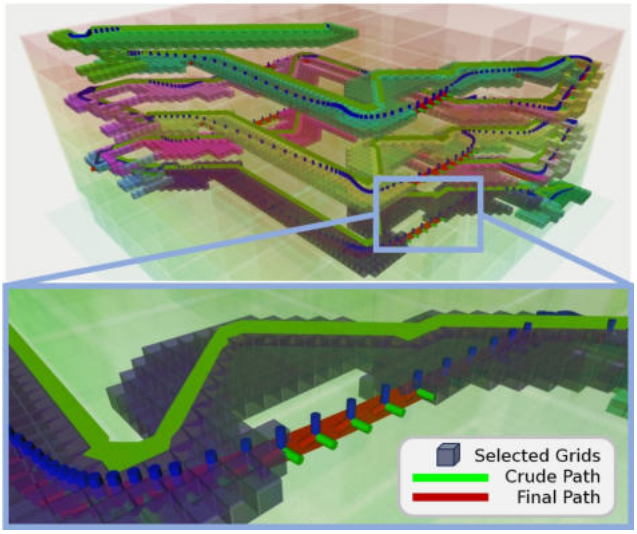


Figure 15. The contribution of agnostic randomness to path generation: In complex 3D environments where numerous impassable regions are present, a graph-search-based planner may yield suboptimal paths (depicted in green). In contrast, the sampling-based planners leverage randomness, allowing the generated paths (depicted in red) to escape the limitations of inadequate heuristic functions and thus produce more optimal solutions.

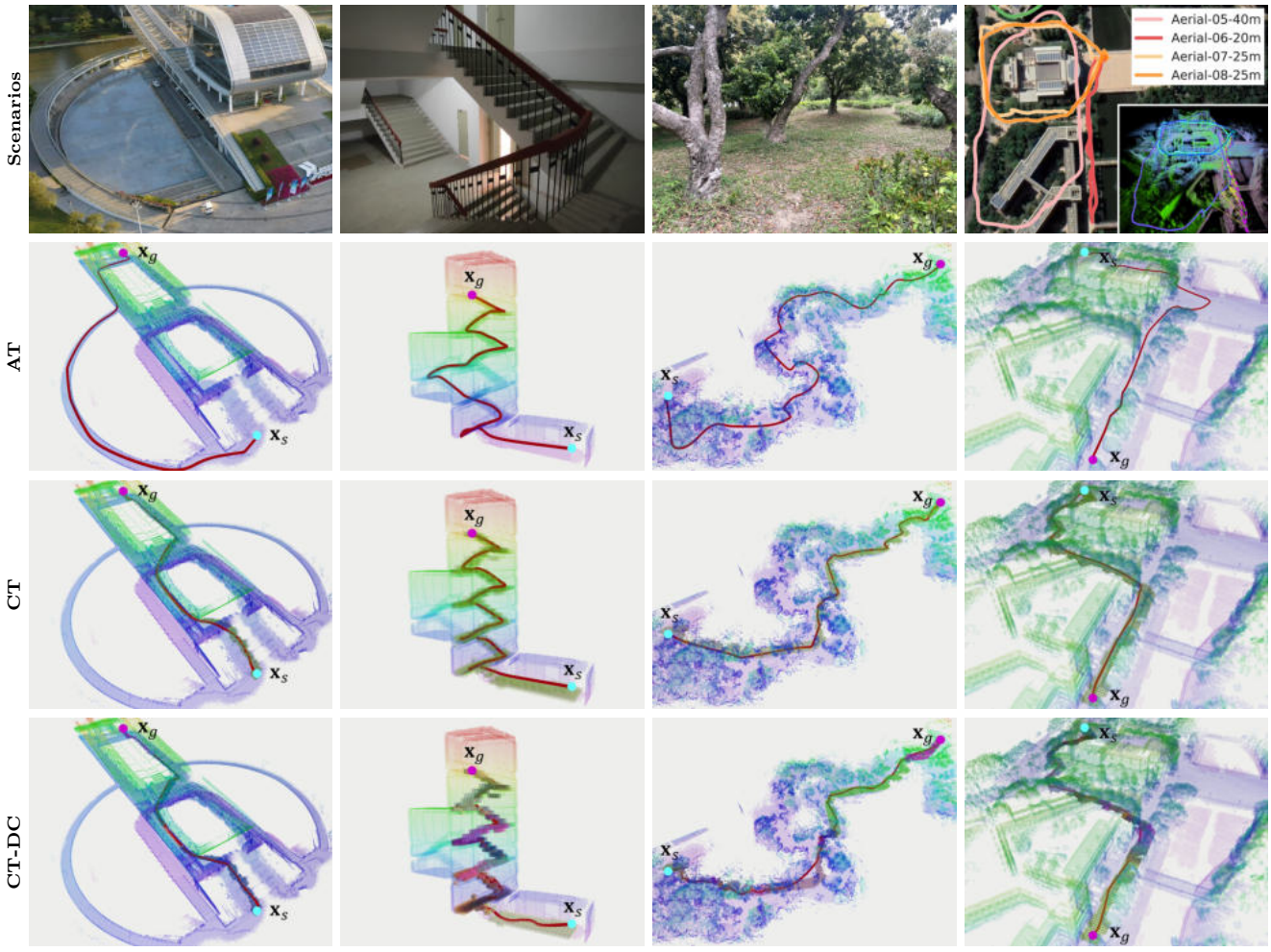


Figure 16. The typical path generation performance under ablation experiments across four real-world scenarios with distinct challenges. These challenges include: (a) confined passageways near the library, (b) staircases, (c) densely obstructed tree regions, and (d) sparse near-terrain point clouds caused by canopy and building occlusions, all of which significantly hinder ground robot path generation.

The choice of the sampling-based planner plays a crucial role in the final results. For instance, the RRT-connect method can significantly speed up path generation, while the RRT* method reduces path redundancy and improves overall path quality (see Fig. 15). Although the time required for path generation varies depending on the sampling-based algorithm used, as shown in Table 2, the proposed CT-DC strategy notably accelerates path generation.

Table 2. The average planning time (s) for different deployed sampling-based planning (SBP) methods in the six-story building scenarios (RRT-C refers to RRT-connect).

	6F building	Restricted 6F building			
	RRT	PRM	RRT	RRT-C	RRT*
AT	29.11	44.19	96.99	101.14	–
CT	0.51	7.13	7.70	8.86	–
CT-DC	**0.27**	**0.43**	**0.25**	**0.21**	**11.95**

A series of ablation studies was conducted across different planning strategies in four real-world environments to evaluate their impacts on the quality of generated paths. We performed 3D reconstructions of an exterior landscape of a library, a fire-fighting rescue stairway, and a litchi forest using a handheld MID-360 LiDAR scanner (see Figs. 16(a)-(c)). Additionally, we reconstructed a typical campus scenario at SYSU (see Fig. 16(d)) based on the airborne LiDAR data from **GrAco**'s public dataset (Zhu et al. 2023), minimizing potential traversable bias from ground-based mapping. These large-scale environments feature extensive traversable areas and complex topological connectivity. Across the four ablation experiment scenarios, the start and goal points can be connected via multiple feasible non-homotopic paths, and are separated by long navigation distances.

In addition to the challenges posed by large-scale environments and complex topological connectivity, each scenario was designed with a specific test focus. The library exterior evaluated planning ability in confined passageways. The rescue stairway scenario assessed planning capability in multi-floor environments with staircases. The litchi forest tested planning robustness in cluttered, unstructured terrain. Finally, the campus scenario examined planning performance

Table 3. Quantifications of the global path generation ablation experiments in real-world scenarios: L_p denotes the average generated path length, Smot. denotes the average generated path smoothness (a smaller number indicates a smoother path), and T_p denotes the average planning time.

	Uni-Town Library			Staircase			Forest			Campus		
	L_p(m)	Smot.	T_p(s)	L_p(m)	Smot.	T_p(s)	L_p(m)	Smot.	T_p(s)	L_p(m)	Smot.	T_p(s)
AT	223.34	**0.09**	21.33	89.62	0.15	9.73	226.81	0.16	15.17	332.68	0.17	37.22
CT	192.25	0.10	4.46	**70.27**	**0.13**	4.71	**180.14**	**0.11**	6.08	**243.44**	**0.12**	7.97
CT-DC	**191.41**	0.10	**0.13**	73.83	0.21	**0.12**	184.55	0.25	**0.11**	251.91	0.31	**0.31**

in sparse near-terrain point clouds with minimal prior traversability information.

The path quality of different strategies was evaluated across all scenarios. To quantify discrete path smoothness, a specific metric (Smot.) was adapted from the work of Dolgov and Thrun (2009) as:

$$\text{Smot.} = \frac{\sum_{i=1}^{N-2} \left\| \Delta \mathbf{x}_{i+1}^{\mathbf{P}} - \Delta \mathbf{x}_{i}^{\mathbf{P}} \right\|_2}{\sum_{i=1}^{N-1} \left\| \Delta \mathbf{x}_{i}^{\mathbf{P}} \right\|_2}, \tag{24}$$

where $\Delta \mathbf{x}_i^{\mathbf{P}} = \mathbf{x}_i^{\mathbf{P}} - \mathbf{x}_{i-1}^{\mathbf{P}}, 1 \leq i \leq N-1$ denotes the displacement at the waypoint $\mathbf{x}_i^{\mathbf{P}} = [x_i, y_i, z_i]^\top$. A larger Smot. value indicates a rougher or less smooth discrete path.

Each strategy was tested 25 times under identical conditions, and various metrics were calculated from the 20 best results. As the data shown in Table 3 and the paths shown in Fig. 16, using all terrain (AT) information for planning introduces higher randomness and leads to longer global paths. In contrast, planning with terrain information along the crude path (CT) significantly reduces path length and improves consistency. Compared to CT, the divide-and-conquer strategy (CT-DC) yields slightly longer and rougher paths with more variation. This is mainly due to randomness introduced by path segmentation (e.g., selection of the sub-paths' endpoints, $\mathbf{x}_{seg}$), which weakens the coherence between path segments. Despite this, CT-DC offers a major improvement in planning efficiency. While CT produces higher-quality paths, it is less efficient in large-scale environments (e.g., in the campus scenario, the average planning time of CT-DC is only about 3.9% of that of CT). For real-time ATR navigation in complex terrains, CT-DC offers a more practical solution, achieving significantly faster planning with only a minor loss in path quality. Within our framework, such suboptimalities can be effectively corrected by the downstream local planner and controller.

5.3 Validation of the manifold-optimization-based local planner

The path optimization problems introduced in Section 4.3 involved complex nonlinear cost functions. Thus, we employed a graph-based optimization method (Kümmerle et al. 2011) to address problem (3) and problem (11). Apart from the inherent differences in the ground adherence cost, the weights in these two optimization problems were consistent (the weight $[\mathbf{\Omega}_k, \mathbf{\Omega}_o, \mathbf{\Omega}_t, \mathbf{\Omega}_\mathcal{P}, \mathbf{\Omega}_a]$ is set as $[100, 100, 100, 100, 100]$). The robot's pitch angle safe threshold $\varepsilon^\mathcal{P}$ is set to 30°.

For clarity, the local planners guided by problems (3) and (11) are referred to as **SE(3)Opt** and **ManiOpt**, respectively. We also reproduced an open-source navigation framework, **3D2M** (proposed by Wang et al. (2023)), which demonstrates competitive performance in multi-floor 3D environments. The 3D2M framework includes a dedicated terrain analysis module capable of directly processing dense point cloud data, a graph-search-based global planner (denoted as **3D2M-G**), an optimization-based local planner (denoted as **3D2M-L**, which imposes various costs as soft constraints on polynomial-continuous trajectories), and a tracking controller for wheeled robots. In this subsection, to control the experimental variables, the comparison primarily focuses on evaluating the local planner of 3D2M (i.e., 3D2M-L), which operates based on its intrinsic terrain analysis module.

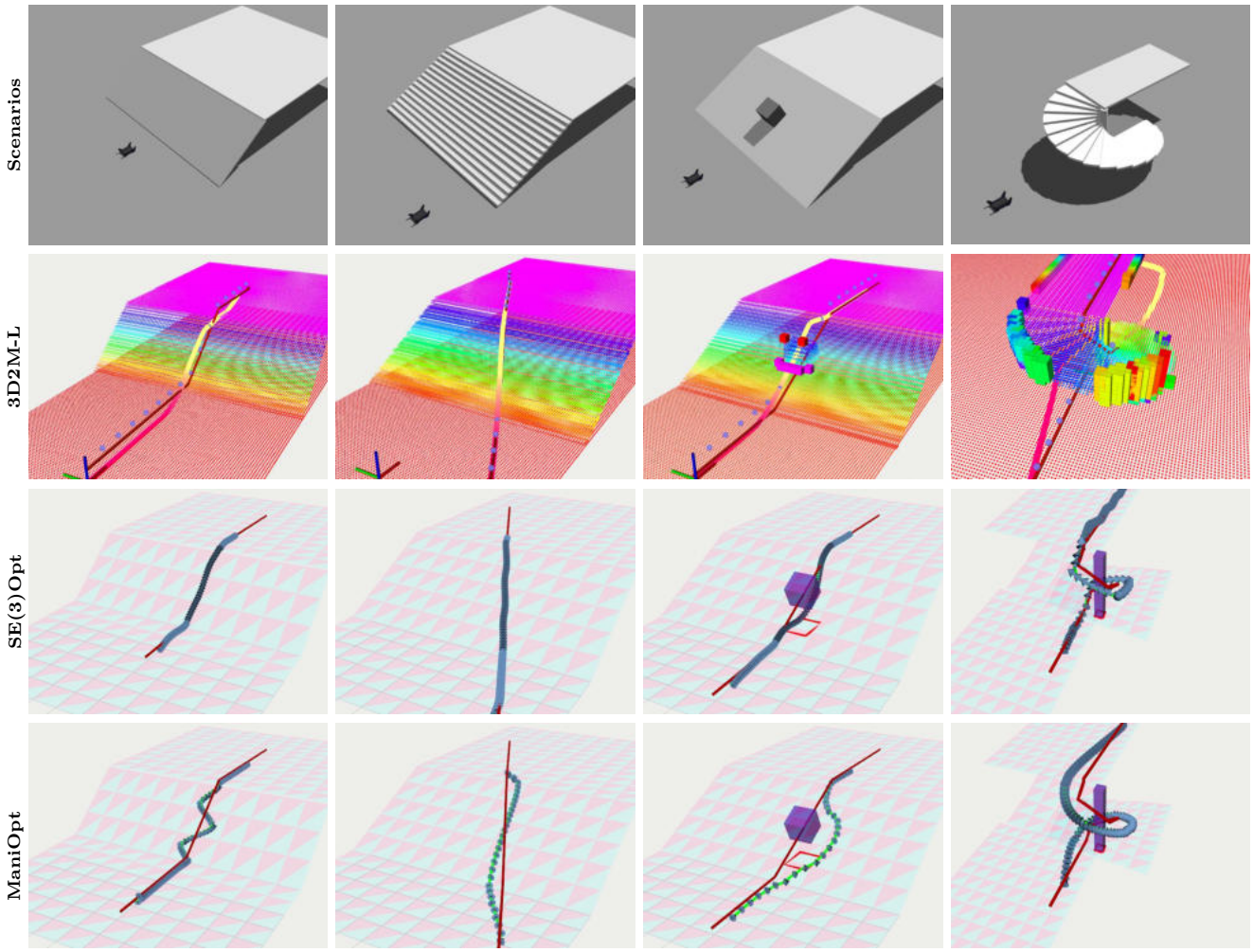
We systematically evaluate three optimization-based local planners, SE(3)Opt, ManiOpt, and 3D2M-L, across four simulated scenarios constructed in Gazebo. For consistency, all planners were provided with the same dense point cloud maps of the environment (resolution: 0.10m), prior terrain texture attributes, and identical global initial paths. A comparative summary of their performance is presented in Table 4 and visualized in Fig. 17.

Table 4. The capability comparison between different optimization-based local planners: The marker √ indicates that a safe path was successfully optimized, while the marker ○ signifies that a perilous path was produced. The marker × denotes that the local planner failed to provide a feasible path.

	Steep slope	Wide stairs	Obst. slope	Spiral stairs
ManiOpt	√	√	√	√
SE(3)Opt	○	○	×	○
3D2M-L	○	○	×	×

In the first two simulated scenarios shown in Figs. 17(a)-(b), 3D2M-L's reliance on highly accurate terrain analysis became a significant bottleneck during optimization, as steep slopes and sharp stair edges were often misclassified as non-traversable regions. Thus, the parameters of 3D2M-L required careful fine-tuning to ensure a successful process. However, when the terrain analysis thresholds were relaxed, 3D2M-L lacked a proper cost formulation to reasonably handle steep inclines and anisotropic traversability, resulting in little practical improvement in optimized trajectories over the initial paths.

In contrast, the 3D terrain-info grid map effectively addressed challenges in complex terrain representation, mitigating the roughness of the point cloud and ensuring better optimization performance. The results



(a) Steep slope **(b)** Staircase with anisotropy **(c)** Slope with obstacles **(d)** Spiral Stairway

Figure 17. The typical path or trajectory optimization results in the simulation scenarios. The red trajectories represent the pre-assigned initial paths, while the optimized trajectories of 3D2M are indicated in a gradient rainbow color. The optimized paths of SE(3)Opt and ManiOpt are indicated in blue. The obstacles in the scenarios are colored purple.

of SE(3)Opt in the first two simulations were less pronounced, primarily due to weight imbalances, which hindered the planner's ability to optimize other objectives while maintaining ground adherence. As depicted in the lower subfigure of Fig. 17(a), by introducing slight perturbations to the initial path on a steep slope (45°), the path optimized by ManiOpt is shown to be meandering and safe. In the scenario illustrated in Fig. 17(b), the initial path diagonally crossed a wide staircase with anisotropic traversability. To avoid dangerous maneuvers on the stairs, ManiOpt adaptively adjusted the path along the direction of the stairs.

As illustrated in Fig. 17(c), the presence of initial global paths that occasionally penetrate obstacles severely disrupted the optimization process. In this scenario, both SE(3)Opt and 3D2M-L failed to produce feasible paths. Although 3D2M demonstrated effective obstacle detection, its soft-constraint design and high sensitivity to initial paths led to solutions that traversed obstacles. SE(3)Opt avoided obstacles to some extent, but due to its soft-constraint formulation of ground adherence, the optimized paths deviated significantly from the actual traversable terrain. In contrast, ManiOpt, with its terrain-manifold-based hard constraints, successfully generated feasible paths that closely adhered to the terrain while avoiding obstacles.

The scene in Fig. 17(d) illustrates a typical spiral staircase scenario. The inner region of this structure features steeper slopes and pronounced changes in anisotropic traversability. In this test, the trajectory optimized by 3D2M-L deviated substantially from the spiral staircase, resulting in an entirely infeasible solution. The path optimized by SE(3)Opt is more zigzagged, closer to the central axis of the stairway, and involves perilous pitch angles. In contrast, the path optimized by ManiOpt is smoother, more gradual, and better adheres to the ground. The benefits of ManiOpt also include fewer optimization variables and cost types, leading to shorter computational times compared to SE(3)Opt. In the planning task depicted in Fig. 17(d), the optimized path spanned approximately 10m with around 30 waypoints. The average optimization time of ManiOpt was about 31.83ms, while SE(3)Opt averaged 41.22ms, saving 22.76% of the time cost.

5.4 Validation of M^2PC for ATR

This subsection evaluates the proposed M^{2}PC for path tracking of an ATR in both a Gazebo-simulated scenario and a real-world staircase setup with a slope of 33.7° (see Fig. 19). For comparison, we reproduced the control algorithm presented by Gianni et al. (2016), which utilizes feedback-based control to track the robot's centroid and flippers, with controller gains set to $K_p = 10.0$, $K_i = 0.005$, and $K_d = 0.1$. Moreover, we implemented an integrated planning-control method based on configuration geometry and terrain analysis from Chen et al. (2023)'s work for further comparison (denoted as **GEO**).

In the optimization problem of Eq. (13) in M^{2}PC for ATR, the state penalty matrices $\mathbf{Q}_k$ and $\mathbf{Q}_N$ were set to diag([20, 20, 20, 100, 100]), indicated a greater emphasis on the stability and safety of the robot's configuration compared to path tracking of the robot's centroid. The control penalty matrix $\mathbf{R}_k$ was set to diag([0.5, 0.5, 0.5, 0.5]) in the experiments.

According to the chassis and flipper size of Slugcat, the span S of the robot was set to 1.3m. Considering the robot's physical dimensions, the sensing range of its onboard sensors, and the accuracy of surface fitting, terrain manifolds in M^{2}PC are locally extracted from point clouds around each waypoint. Specifically, only the down-sampled ground points within a 1.3m radius around each waypoint were used to efficiently extract spatial uniformity and reliable local terrain manifolds for M^{2}PC.

An excessively long horizon for M^{2}PC is neither necessary nor beneficial. On the one hand, too many waypoints dilute the optimization cost near significant terrain transitions, potentially compromising robot stability. On the other hand, the limited update range of the local submap hinders the robot from accurately acquiring terrain information around distant waypoints, which can lead to inaccurate terrain estimation that may interfere with the controller's system model. Therefore, the prediction horizon of M^{2}PC was limited to 10. The influence factors c_v and c_ω in the system model (21) were set to 10.0 and 0.0, respectively, to ensure stable maneuvering on stair-like terrain, which has anisotropic traversability.

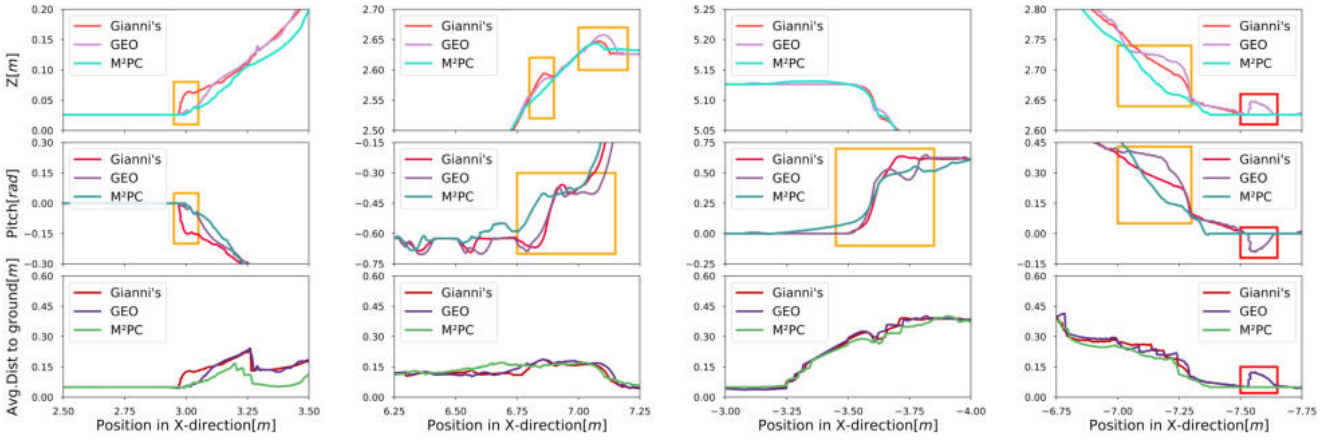
The same reference path was utilized for consistent simulations. The robot would first climb up the staircase and then descend from the other side. Considering the actuator capabilities and the transmission design of the flipper, the maximum flipper rotation velocities were set as 0.1rad/s. To evaluate how well the robot aligned with the terrain under different control methods, 100 points on the robot's base footprint were uniformly sampled, and the average distance between these points and the terrain surface was computed. Furthermore, the robot's centroid position along the z direction and the variations in pitch angle throughout the maneuver were monitored and recorded.

The quantitative performances of the control algorithms in the simulated scenario featuring severe terrain variations are presented in Fig. 18. Gianni's method adjusts the flipper angles in real-time based on the terrain plane fitted from the point cloud around the robot, aiming to maintain close contact with the ground. As a result, under conditions where the flipper motion speed is limited, Gianni's method tends to suffer from delayed configuration adaptation, leading to issues such as free-spinning or robot impact against the terrain across various phases. Meanwhile, GEO's core assumption posits that ATR's stability is not solely dependent on robot-ground adherence. Considering the reference robot centroid path, GEO would sample flipper configurations near the waypoints and evaluate them based on the robot's constraints and the dense prior point cloud map. An elaborately defined cost function was then employed to select the optimal flipper configuration, and an integrated controller was utilized for general tracking. Consequently, during the transition from flat ground to stair climbing, although the robot body does not closely adhere to the terrain, the ascent in the z direction and pitch angle variation remains relatively smooth (see Fig. 18(a)). However, in the transition from stairs to elevated platforms and throughout all descent phases, GEO fails to adapt the configuration in response to rapid terrain changes, resulting in abrupt changes in the robot's posture. Moreover, during the descent from stairs to flat ground, GEO is prone to misinterpret terrain configurations, leading to flipper misoperations (see Fig. 18(d)).

Compared to the baseline methods, M^{2}PC enabled smoother maneuvering of the robot, more continuous and safer adjustments of the body pitch angle, and better contact performance between the robot and the ground. By incorporating information about surface curvature changes, M^{2}PC allowed the robot to achieve better whole-body control. Especially when the flipper angle has not yet reached the desired position, the chassis actively slows down, achieving coordinated stair climbing. This whole-body movement mitigates impacts in the z direction and suppresses rapid changes in pitch angle. Additionally, when descending, the robot adjusted the joint configuration and reduced body oscillations.

Fig. 20 illustrates the robot's performances on a lab-built staircase using various control strategies. During the ascent phase, Gianni's feedback control method lacked coordination, resulting in the robot's inability to adjust its configuration in response to terrain changes and leading to impacts with the stairs. When the robot moved onto the landing, this strategy also failed to coordinate the flippers' posture, causing the robot to experience a significant impact of up to 8G. This issue also occurred during the descent phase, where shaking in the z direction further confirms the problem. GEO exhibited similar performance, with the chassis experiencing significant vibrations throughout the stair-crossing.

In contrast, M^{2}PC effectively satisfied both terrain manifold constraints and robot kinematic limitations, thereby mitigating the aforementioned issues. During the stair-crossing, M^{2}PC adjusted the robot's configuration to ensure a smoother transition between terrains.



(a) Climbing(Floor-Stairs) **(b)** Climbing(Stairs-Floor) **(c)** Descending(Floor-Stairs) **(d)** Descending(Stairs-Floor)

Figure 18. The typical performance of ATR in crossing stairs within the simulation scenarios: In the orange box, the proposed M^{2}PC method significantly smoothed the robot's maneuvers through severe terrain changes compared to baseline methods. The malfunction effects of GEO are shown in the red boxes.

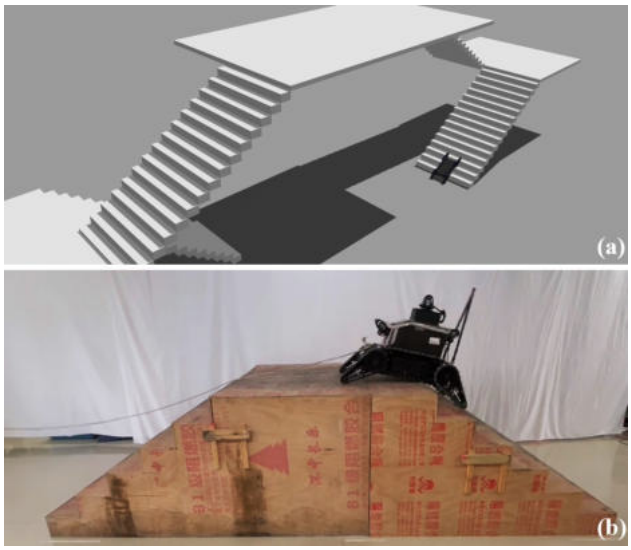


Figure 19. The ATR maneuvering in the Gazebo-simulated scenario (a) and the lab-built staircase (b).

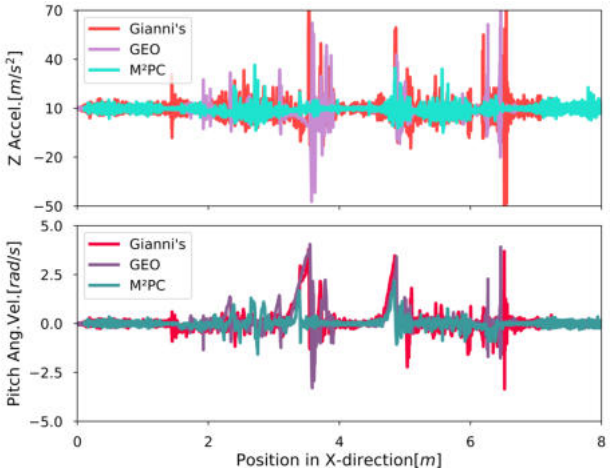


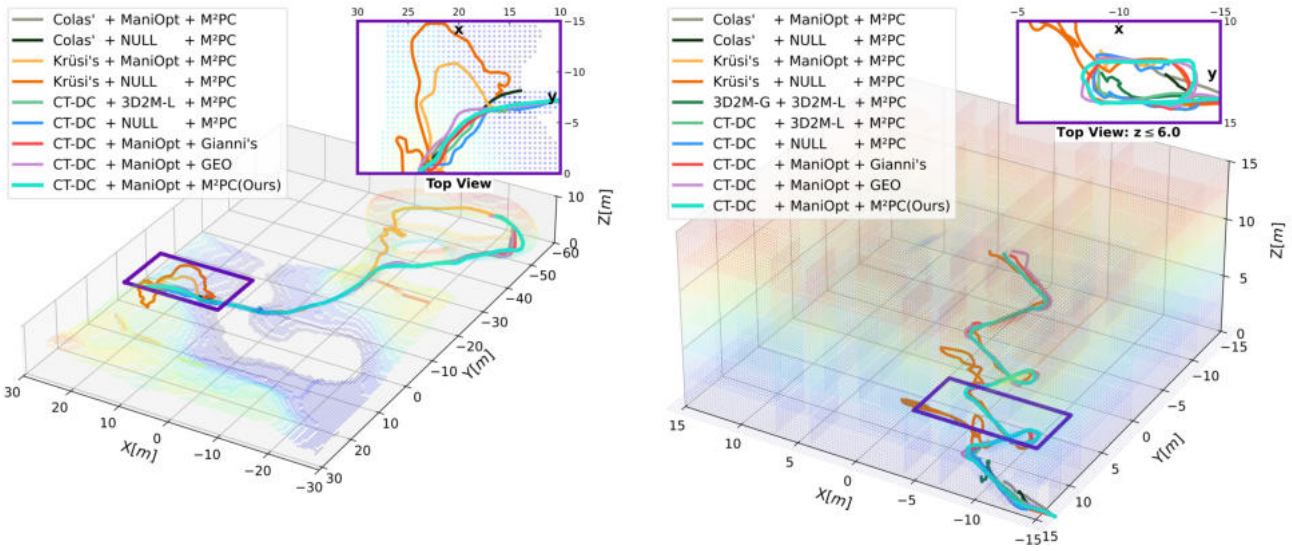
Figure 20. The typical acceleration and angular velocity profiles of the ATR on the lab-built staircase.

Compared to the baseline methods, M^{2}PC's maneuvering strategy was more conservative, safer, and prevented severe impacts.

5.5 *Validation of the framework in simulated scenarios*

We comprehensively validated the proposed framework in two challenging simulation environments. The wild simulation scenario (shown in Fig. 12(b)) features a lake, a bridge, caves, and significant ground undulation (an elevation difference of approximately 10 meters). The robot has to cross the lake to reach the goal point in the inner cave. The building simulation scenario (the simple version of the six-story building, as shown in Fig. 12(a)) features extensive and complex connectivity, requiring the robot to traverse multiple stairways to reach the goal point that is elevationally distant from the start point. Throughout the simulations, the robot's pose, flipper joint states, global point cloud maps of the environment, and the local submap surrounding the robot were directly obtained from Gazebo.

We conducted extensive ablation studies by comparing multiple combinations of the proposed methods and several state-of-the-art methods, to comprehensively evaluate the performance of the proposed framework and its key modules. Multiple times of simulations were performed for each combination across the two Gazebo-simulated scenarios, and the success rates of different algorithmic combinations were compared. The detailed quantitative evaluation results are summarized in Table 5, and the best-performing executed trajectories for each combination are illustrated in Fig. 21. In all simulations, the planning and tracking control processes were performed online. For clarity, we labeled each combination from A to J. Combinations A, B, C, D, and J were primarily used to evaluate the impact of different global path generation methods on overall navigation performance. Combinations F, G, and J focused on analyzing the contributions of various local path or trajectory optimization methods. Combinations H, I, and J were designed to assess the influence of different tracking control methods. Furthermore, due to the limited availability of comprehensive and well-established frameworks in the current literature, the



(a) General simulations in the wild

(b) General simulations in the building

Figure 21. The best-performing executed trajectories of different combinations in the Gazebo-simulated scenarios.

Table 5. Quantifications of the exhaustive ablation simulations in the Gazebo-simulated scenarios: Succ. denotes the success rate, d_{min} denotes the minimal Euclidean distance from the robot to the navigation goal, and Asc. denotes the maximum ascent in the building, L_e denotes the average executed trajectory length, and T_e denotes the average execution time.

				Wild				Building			
	Global	Local	Ctrl.	Succ.(%)	d_{min}(m)	L_e(m)	T_e(s)	Succ.(%)	Asc.(m)	L_e(m)	T_e(s)
A	Colas'	ManiOpt	M²PC	50	$\leq$ **0.2**	93.63	187.7	0	0.03	–	–
B	Colas'	NULL	M²PC	0	59.44	–	–	0	0.19	–	–
C	Krüsi's	ManiOpt	M²PC	20	$\leq$ **0.2**	120.07	246.0	0	2.74	–	–
D	Krüsi's	NULL	M²PC	0	55.19	–	–	10	**15.13**	102.42	318.5
E	3D2M-G	3D2M-L	M²PC	–	–	–	–	0	2.16	–	–
F	CT-DC	3D2M-L	M²PC	0	53.74	–	–	0	2.63	–	–
G	CT-DC	NULL	M²PC	0	54.74	–	–	0	4.97	–	–
H	CT-DC	ManiOpt	Gianni's	**100**	$\leq$ **0.2**	93.17	**184.0**	**100**	**15.13**	53.97	**128.8**
I	CT-DC	ManiOpt	GEO	90	$\leq$ **0.2**	100.02	358.3	50	**15.13**	60.26	488.8
J	CT-DC	ManiOpt	M²PC	**100**	$\leq$ **0.2**	**92.89**	185.3	**100**	**15.13**	**53.82**	145.1

3D2M framework (excluding its wheeled robot controller) was selected as a representative baseline for fair comparison at the framework level, denoted as combination E.

5.5.1 Analysis of simulation results in the wild

In the wild scenario, the tested combinations exhibited varied performances, illustrating the impact of module-level design on navigation quality. The simulation results (see Table 5 and Fig. 21(a)) indicate that five combinations completed the navigation task at least once. For combinations A and B, the global paths generated by the graph-search-based planner during the narrow bridge were almost entirely located within infeasible, obstacle-occupied regions, directly leading to inevitable collisions for combination B at the bridge. Although with the support of a relatively robust ManiOpt local planner, combination A may successfully traverse the narrow bridge and complete the navigation task, its overall success rate was relatively low, mainly restricted by the extreme proximity of the initial global paths to obstacles.

Combination C completed two navigation trials. However, due to significant randomness in its global path generation process, the executed trajectories were highly redundant, substantially prolonging task completion time and indirectly increasing the failure risk, thus lowering the overall success rate. Combination D consistently failed to complete navigation before the bridge due to the absence of local obstacle avoidance capabilities. Combination E employed the 3D2M framework (excluding its original controller). However, 3D2M's A* global planner consistently failed due to the limitations of its terrain analysis module, rendering the navigation task impossible. Combination F, which used the 3D2M local planner (3D2M-L), showed moderate performance in rugged terrain with its soft-constraint trajectory optimization method. However, it failed to handle narrow terrains, causing the robot to stall before the bridge due to unsuccessful path optimization. Combination G completely disabled local path optimization; since the CT-DC global planner still had some randomness, the robot failed to avoid the guardrails on the bridge, ultimately leading to navigation failure.

In the wild scenario, terrain variations were relatively mild, and the rigid simulated ground provided generous friction allowances. Although Gianni's feedback controller exhibited some slippage during navigation, its maneuvering capability was comparable to that of

M^2PC. Due to the conservative nature of M^2PC, the proposed framework J exhibited slightly longer execution times but shorter execution trajectories compared to combination H (which used Gianni's controller). Combination I further evaluated the performance of GEO in standard online simulations. GEO required path interpolation to smooth the flipper motions, and it took seconds to analyze the robot's configurations, leading to poor real-time control performance. It often failed to respond promptly to the current reference path, frequently exhibiting significant delays and overshoots during navigation. Overall, the proposed framework J demonstrated the best comprehensive performance, achieving the highest success rate and the shortest actual execution trajectories.

5.5.2 Analysis of simulation results in the building

Significant height differences, consecutive stairway turns, and complex connectivity in the six-story building posed even greater challenges for ATR navigation. The simulation results (see Table 5 and Fig. 21(b)) indicate that four combinations completed the navigation task at least once. The graph-search-based global planners used in combinations A, B, and E often generated paths dangerously close to stair edges or walls, increasing the risk of optimization or execution failure. Intuitively, combination C should outperform combination D; however, simulation results showed that combination C consistently failed in all 10 trials. Combination D succeeded once, but the randomness in its global path generation led to highly redundant paths with frequent detours, significantly increasing task duration.

In combination F, the 3D2M terrain analysis module failed to correctly identify obstacles at stairway corners, causing the robot to fall or get stuck. Combination G failed all 10 trials due to unrecoverable collisions. Although the CT-DC global planner considered vehicle stability on the ground and the controller generally tracked the path well, the absence of local path optimization led to occasional collisions at the stairway turns.

In Gazebo, the relaxed friction and collision criteria allowed Gianni's method to complete the navigation task despite experiencing slippage and impacts on the stairs. Compared to combination H, which deployed Gianni's controller, the proposed framework J achieved slightly longer execution times but shorter execution paths due to M^2PC's more conservative and safer control strategies. Combination I (with GEO) also failed to achieve smooth online navigation, frequently experiencing stop-and-go behavior during stair climbing, significantly prolonging task duration. Among all combinations that completed navigation in the building scenario, the proposed framework J demonstrated superior performance in both success rate and execution efficiency.

5.5.3 Discussion on simulation

As illustrated in Fig. 21, a high-quality global path generated by CT-DC provides a favorable initialization for local optimization, reduces the robot's execution distance and time, and avoids unnecessary detours. Meanwhile, the comparison of success rates among different combinations further highlights the importance of local path refinement. In both scenarios, ManiOpt serves multiple purposes, such as enabling obstacle avoidance on the narrow bridge, smoothing the path, and satisfying the anisotropic traversability constraints on staircases. Disabling local optimization or initializing it with a poor path often leads to task failure. Moreover, locally optimized paths that satisfy multiple objectives can help prevent the controller from being misled by obstacles and reduce the difficulty of control over terrain with complex traversability. For instance, if the local point cloud map for quadratic terrain analysis includes structures such as railings or walls, terrain-aware controllers may misinterpret these as drivable surfaces, resulting in control failures. Similarly, directly following overly coarse initial paths (such as winding paths over stairs) may trigger excessive motion responses from the controller. Finally, safe maneuvering of the ATR over complex and rugged terrain requires not only a controller (like M^2PC) capable of generating coordinated whole-body motions under both terrain and robot-specific constraints but also strong real-time performance at both planning and control levels to avoid overshoot caused by delayed responses and redundant motions arising from stochastic variations of the planned path.

5.6 Validation of the framework in real-world scenarios

We selected several real-world scenarios characterized by considerable challenges, to systematically evaluate the proposed framework alongside baseline methods. Each experiment presented in Section 5.6.1 to Section 5.6.4 was conducted using prior point cloud maps with an identical size and a uniform resolution of 0.1 meters. The experiment in Section 5.6.5, however, utilized online point cloud maps generated by the SLAM module. Throughout all real-world experiments, the robot's localization was provided by a modified version of FAST-LIO2 (Xu et al. 2022), which includes a re-localization function. The joint states of the flippers and the local submaps were continuously updated based on data collected from the onboard sensors.

5.6.1 Architecture complex

We conducted a series of experiments within an architecture complex (see Fig. 22). The primary objective of this experimental setup was to evaluate the practical effectiveness of various global path planning algorithms (including CT-DC, Colas' method, Krüsi's method, and parts of the 3D2M planning pipeline) in large-scale, complex-connected environments with varying heights of traversable ground. In these experiments, M^2PC is consistently used as the tracking controller. Except for the comparative combination involving the 3D2M framework, all experimental combinations employ the ManiOpt local path optimizer.

The start point for all navigation tasks are located near $(12.0\text{m}, 3.0\text{m}, 0.2\text{m})$ in the global coordinate system, with goal point near $(96.3\text{m}, -45.1\text{m}, 5.9\text{m})$. After conducting five trials for each combination, the best-performing executed trajectories were visualized in Fig. 23. The corresponding key quantitative metrics are summarized in Table 6.

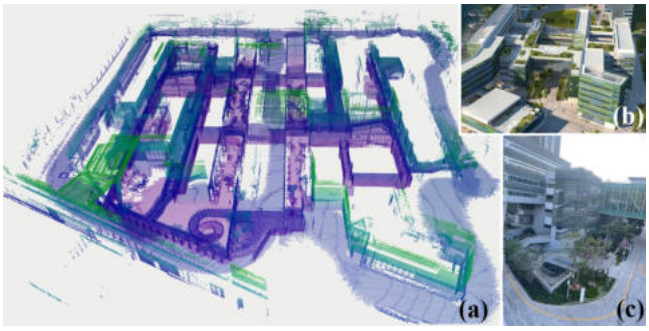


Figure 22. A typical architecture complex on a campus: (a) the reconstructed point cloud map (including the first three floors), (b) the general CG rendering, and (c) a corner of the scenario.

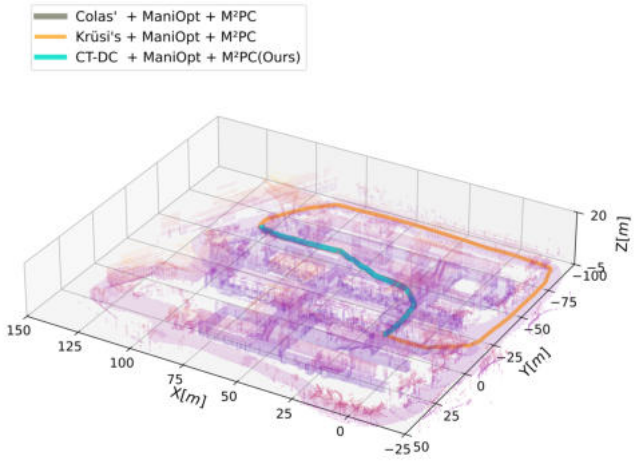


Figure 23. The best-performing executed trajectories of different combinations in the architecture complex. The deterministic nature of CT-DC and Colas' methods allowed the robot to efficiently navigate through stairways within the building and reach the goal via significantly shorter trajectories. In contrast, the stochastic Krüsi's method resulted in detoured trajectories.

Except for one failure caused by crowd interference, the proposed framework consistently outperformed the comparative combinations in terms of success rate, average executed trajectory length, global planning time at the start point, and overall navigation duration. Colas' method also achieved relatively efficient execution trajectories, but its planning time at the start point was longer due to the large-scale environment. Moreover, local path optimization occasionally failed in certain areas (like long corridors) due to poor initial paths from Colas' method, resulting in collisions with obstacles and ultimately compromising the overall navigation success rate.

Krüsi's method, affected by the superfluous traversable areas in the environment, produced highly stochastic global paths. This randomness resulted in frequent changes in heading direction (failed four times on the staircases and one success with an offline-generated path), substantially increasing the executed trajectory length, planning time, and overall execution duration. The terrain analysis module in 3D2M imposed heavy computational demands. At practical resolutions (below 0.5m), it failed to complete terrain processing on onboard computers. At coarser resolutions (around 1.0m), the algorithm produced inaccurate traversability estimates, resulting in global planning failures.

Table 6. Quantifications of the experiments in the architecture complex: Succ. denotes the success rate, L_e denotes the average executed trajectory length, T_p denotes the average global planning time at the start point, and T_e denotes the average execution time.

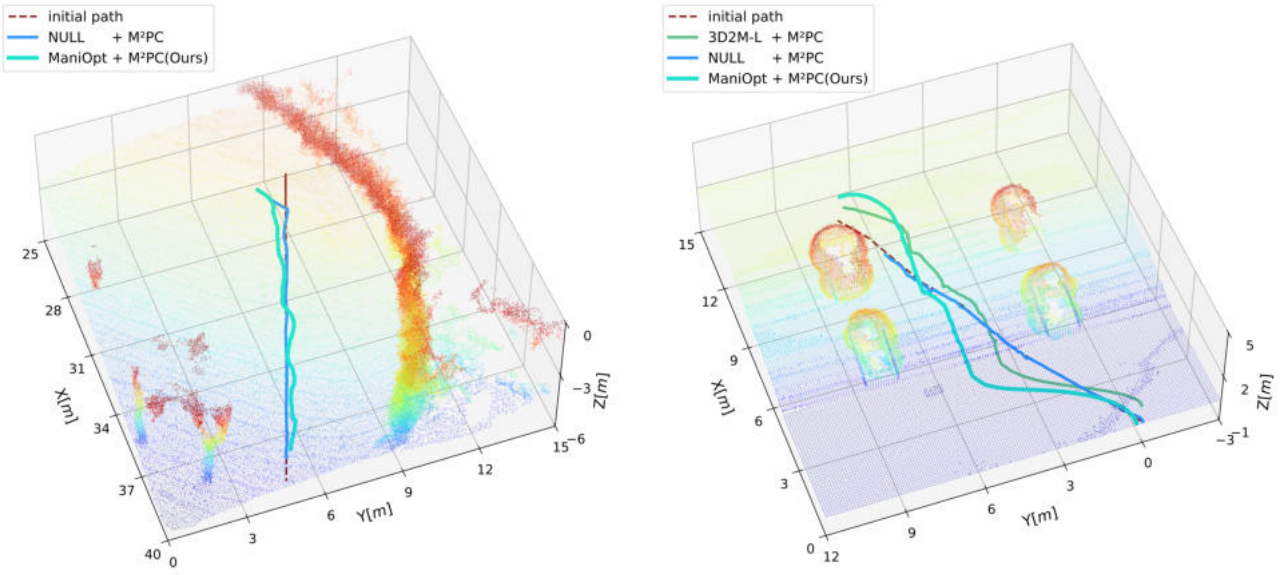
Global	Local	Ctrl.	Succ.(%)	L_e(m)	T_p/T_e(s)
CT-DC	ManiOpt	M^{2}PC	**80**	**118.7**	**0.3/310.6**
Colas'	ManiOpt	M^{2}PC	40	119.8	3.9/339.8
Krüsi's	ManiOpt	M^{2}PC	20	258.3	43.2/659.7
3D2M-G	3D2M-L	M^{2}PC	0	0.0	$\geq$ 1200/–

5.6.2 Challenging uneven terrains

We conducted a series of experiments on a steep grassy slope and a set of multi-segment staircases to further evaluate the effectiveness of different local path or trajectory optimization algorithms in real-world environments. The grassy slope, shown in Fig. 24(a), featured soft, wet terrain with a high risk of slippage, posing significant challenges to ATR navigation. This experimental setup aimed to assess how well different local optimization algorithms could refine aggressive initial paths in steep environments. The multi-segment staircases, shown in Fig. 24(b), were used to test each algorithm's ability to optimize initially direct, non-adaptive paths in terrains with pronounced anisotropic traversability.

Figure 24. Two types of terrains with distinct challenges: (a) grassland with steep slope and (b) multi-segment wide staircases.

In these experiments, M^{2}PC was consistently employed as the tracking controller. The initial paths were unified, adhered to the ground, and designated manually to ensure experimental consistency. Each algorithm was tested five times on both environments, and the best-performing executed trajectory for each was selected for visual presentation (see Fig. 25). The corresponding key quantitative metrics for each method are summarized in Table 7.



(a) Steep slope on the grassland **(b)** Multi-segment staircases

Figure 25. The best-performing executed trajectories of different combinations on the two challenging uneven terrains.

Table 7. Quantifications of the experiments on the challenging uneven terrains: Succ. denotes the success rate, d_{min} denotes the minimal Euclidean distance from the robot to the navigation goal, a_{max} denotes the maximum linear acceleration norm during the best-performing navigation, and Mal. denotes the average robot's malfunction index (a smaller number indicates better control performance).

			Grassland				Multi-segment staircases			
Global	Local	Ctrl.	Succ.(%)	d_{min}(m)	a_{max}(m/s^2)	Mal.	Succ.(%)	d_{min}(m)	a_{max}(m/s^2)	Mal.
NAN	ManiOpt	M^{2}PC	**100**	$\leq$ **0.2**	**29.92**	**0.07**	**100**	$\leq$ **0.2**	**46.28**	**0.16**
NAN	3D2M-L	M^{2}PC	–	–	–	–	40	$\leq$ **0.2**	71.40	0.33
NAN	NULL	M^{2}PC	**100**	$\leq$ **0.2**	41.89	0.19	0	3.97	–	–

To evaluate the execution performance of the robot, we defined a malfunction metric (Mal.) based on the deviation between the desired commands and actual velocity over a certain time horizon. Let $v^x_{t,cmd}$ and $v^x_{t,a}$ denote the longitudinal velocity command and the actual longitudinal velocity in the chassis coordinate frame F_b at time t, respectively. Over a T time steps of navigation on the uneven terrains, the Mal. index is computed as:

$$\text{Mal.} = \frac{1}{T}\sum_{t=0}^{T}\left|v^x_{t,a} - v^x_{t,cmd}\right|. \tag{25}$$

This metric serves as an indicator of how accurately the robot follows the intended control commands. A lower Mal. value reflects better control consistency.

On the grassland, the initial path descended rapidly, which posed stability challenges. The paths optimized by ManiOpt on the grassy slope effectively circumvented steep slopes and significantly reduced the pitch stability cost $\mathbf{e}_{\mathcal{P}}$ in the steep regions. These optimized paths adhered closely to the terrain manifold and exhibited strong disturbance rejection. More importantly, the conservative, terrain-adaptive paths generated by ManiOpt substantially lowered the risks of chassis destabilization and reduced the robot's peak centroid accelerations. These results underscore the critical importance of terrain-manifold-based hard constraints in optimizing feasible and stable local paths. Meanwhile, 3D2M-L failed to execute its optimization process, as its terrain analysis module was unable to process traversability estimation in this scenario due to the size and high resolution of the point cloud map.

On the staircases, ManiOpt employed an adaptive optimization objective tailored to the specific terrain, ensuring safe traversal on the anisotropically traversable stairs. It also enabled coordinated adjustment on flat, isotropically traversable grounds, helping to shorten the overall path length. Throughout all test runs, ManiOpt consistently avoided path-terrain detachment, demonstrating advantages in mitigating potential controller malfunctions and failures. These results strongly validate the effectiveness of the manifold optimization approach. In contrast, the 3D2M-L optimizer failed to produce a trajectory that maintains good ground adherence or accounts for anisotropic traversability, resulting in lateral slips and partial loss of control on the second-segment staircase. In experiments conducted without optimization, the robot was unable to perform adequate obstacle avoidance, resulting in perilous collisions.

5.6.3 Debris

We conducted a series of experiments at a post-demolition site to evaluate the performance of different ATR whole-body control methods in real-world scenarios, as shown in Fig. 26. The experiment area

Table 8. Quantifications of the experiments on the debris: Succ. denotes the success rate, d_{min} denotes the minimal Euclidean distance from the robot to the navigation goal, a_{max} denotes the maximum linear acceleration norm during the best-performing navigation, T_e denotes the average execution time, L_e denotes the average executed trajectory length, Smot. denotes the average executed path smoothness, and Mal. denotes the average robot's malfunction index.

Global	Local	Ctrl.	Succ.(%)	d_{min}(m)	a_{max}(m/s^2)	T_e(s)	L_e(m)	Smot.	Mal.
CT-DC	ManiOpt	M^{2}PC	**60**	$\leq$ **0.2**	**18.01**	**59.00**	**17.96**	**0.30**	**0.38**
CT-DC	ManiOpt	Gianni's	0	2.11	21.73	–	–	0.53	0.51
CT-DC	ManiOpt	GEO	0	5.98	19.22	–	–	0.58	0.51
3D2M-G	3D2M-L	M^{2}PC	0	7.24	20.90	–	–	0.54	0.47

covered approximately 400m^2 and presented significant challenges, including unstable rubble piles, soft soil slopes, and numerous hard protrusions, realistically simulating a post-disaster environment. Additionally, we compared the performance of the 3D2M framework when integrated with M^{2}PC. Each algorithm combination was tested five times, with a fixed start and goal point located near $(0.0\text{m}, 0.0\text{m}, 0.1\text{m})$ and $(16.5\text{m}, 0.0\text{m}, 3.4\text{m})$, respectively. The best-performing executed trajectories were selected for visual presentation (see Fig. 27). The corresponding key quantitative metrics for each combination are summarized in Table 8.

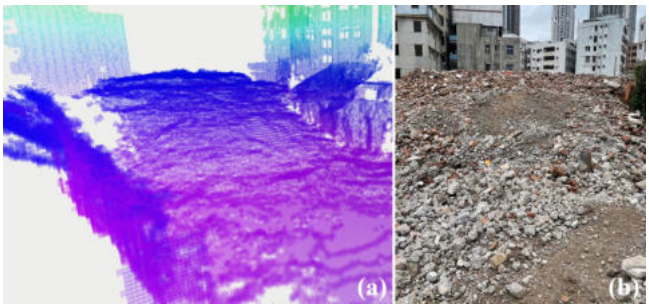


Figure 26. The post-demolition debris: (a) the reconstructed point cloud map and (b) the front side of the debris.

In the experiments, the global path generated by 3D2M was relatively straight and consistent. However, its local optimization was heavily influenced by terrain roughness analysis, resulting in a highly tortuous path. This path caused the robot to perform excessive twists over the rubble, leading to entrapment on small, rigid protrusions (see Fig. 27). The combination using Gianni's controller frequently experienced poor contact between the robot's body and the soft soil slopes, which often resulted in track subsidence. Moreover, when encountering minor rigid obstacles on the ground, Gianni's controller also tends to get stuck. Furthermore, GEO failed in all five trials due to the significant computation failure of flipper configurations in rough terrain.

In contrast, the proposed framework is the only solution capable of completing the rubble terrain navigation task. Although two failures occurred due to gravel slides and rebar obstructions, the overall performance remained robust. With the integration of the extracted quadratic-form terrain information and the robot's inherent constraints, M^{2}PC demonstrated safer control behaviors during execution, effectively reducing the risks of the robot becoming trapped in soft soil or immobilized by rubble.

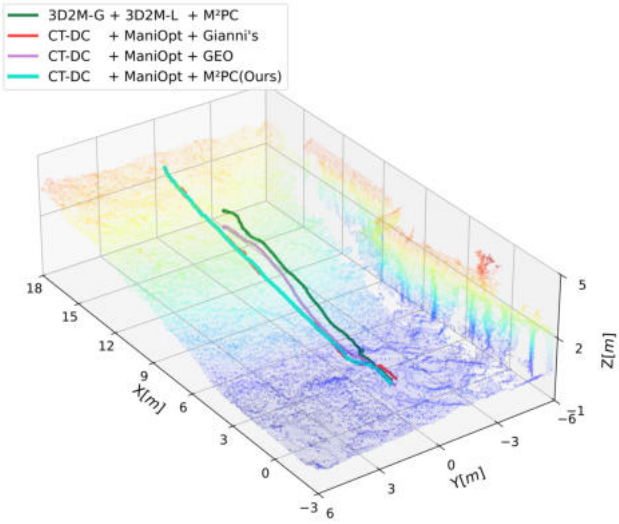


Figure 27. The best-performing executed trajectories of different combinations on the debris.

5.6.4 *Interlaced stairways*

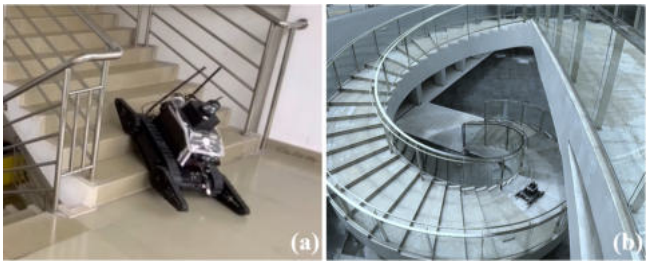
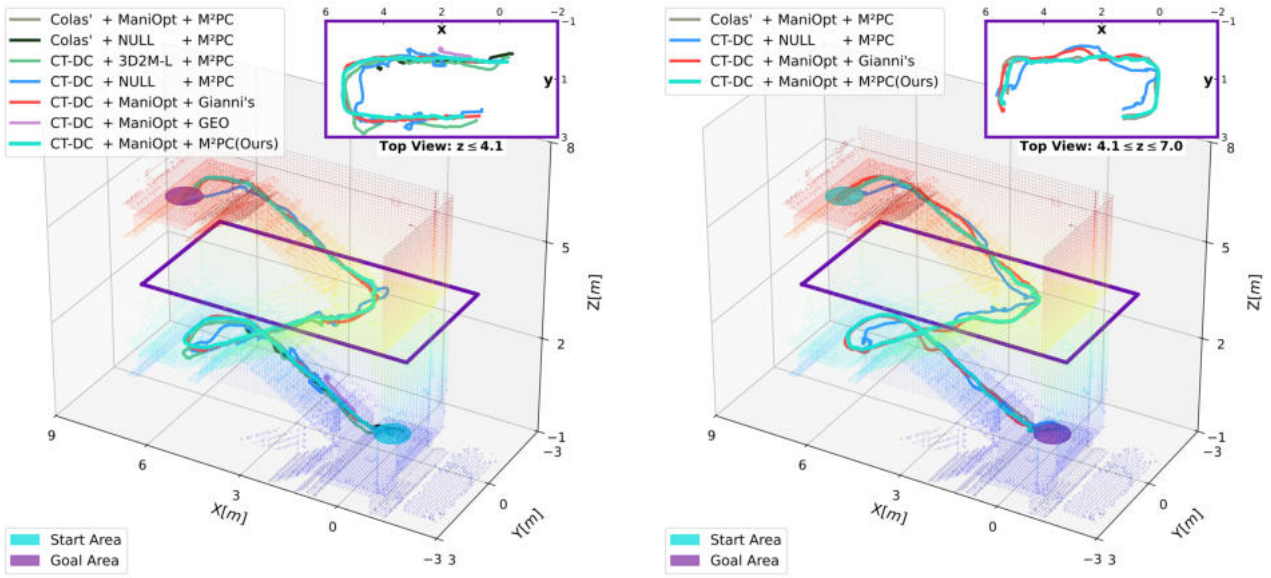


Figure 28. Two stairways with multi-floor structure: (a) rescue stairway and (b) spiral stairway.

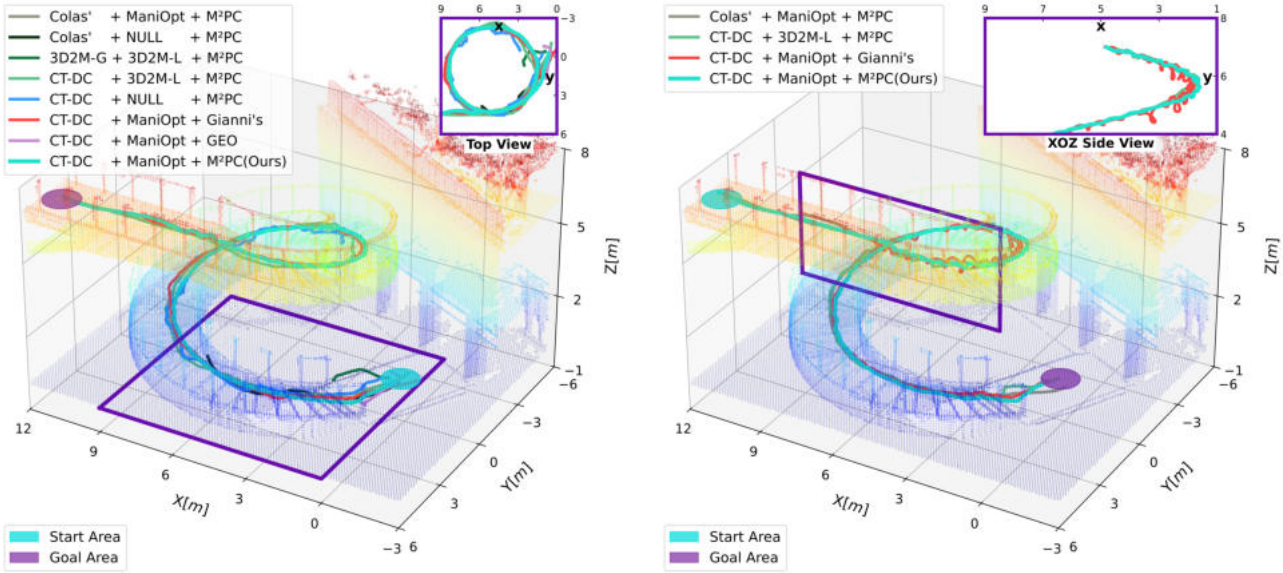
We conducted a series of experiments on a fire rescue stairway and an irregular spiral stairway (see Fig. 28), to evaluate the capabilities of different ablation combinations in multi-floor scenarios. Both scenarios consisted of multiple staircases and platforms, with a maximum traversable elevation of approximately 6 to 7 meters. The horizontal overlap of the stair structures and the complex traversability conditions provided an excellent setting for assessing the performance of each algorithm combination.

Due to the constrained connectivity of the experiment environments, Krüsi's method and the CT exhibited similar path generation patterns; thus, further comparison with Krüsi's method was not conducted. In each experiment, the robot started from a low-elevation start point, climbed the stairs to reach goal point 1



(a) Upward navigation **(b)** Downward navigation

Figure 29. The best-performing executed trajectories of different combinations on the rescue stairway.



(a) Upward navigation **(b)** Downward navigation

Figure 30. The best-performing executed trajectories of different combinations on the spiral stairway.

Table 9. Quantifications of the ablation experiments on the stairways: Succ. denotes the success rate, Asc. denotes the maximum ascent on the staircases, a_{max} denotes the maximum linear acceleration norm during the best-performing navigation, and Smot. denotes the average executed path smoothness.

				Rescue stairway				Spiral stairway			
	Global	Local	Ctrl.	Succ.(%)	Asc.(m)	a_{max}(m/s^2)	Smot.	Succ.(%)	Asc.(m)	a_{max}(m/s^2)	Smot.
A	Colas'	ManiOpt	M^{2}PC	20	**6.37**	**22.17**	0.37	20	**7.13**	31.30	0.53
B	Colas'	NULL	M^{2}PC	0	1.99	30.66	0.54	0	1.97	39.87	0.31
E	3D2M-G	3D2M-L	M^{2}PC	–	–	–	–	0	0.30	50.49	0.50
F	CT-DC	3D2M-L	M^{2}PC	0	5.87	23.10	0.49	20	**7.13**	31.09	0.30
G	CT-DC	NULL	M^{2}PC	20	**6.37**	53.11	0.44	0	5.26	40.26	0.46
H	CT-DC	ManiOpt	Gianni's	60	**6.37**	80.93	0.36	40	**7.13**	74.35	0.41
I	CT-DC	ManiOpt	GEO	0	0.97	57.34	1.17	0	0.41	40.01	0.40
J	CT-DC	ManiOpt	M^{2}PC	**100**	**6.37**	23.14	**0.32**	**100**	**7.13**	**25.57**	**0.28**

on a high platform, and then descended to return to goal point 2 near the original start point. For the rescue stairway scenario, the start point and goal point 2 were located near $(0.0\text{m}, 0.0\text{m}, 0.1\text{m})$, while goal point 1 was placed at $(5.6\text{m}, 1.9\text{m}, 6.4\text{m})$, as the colored circle illustrated in Fig. 29. For the spiral

stairway scenario, the start point and goal point 2 were located near (0.0m, 0.0m, 0.1m), with goal point 1 set at (11.0m, 4.5m, 7.1m), as the colored circle illustrated in Fig. 30. Each algorithm combination was tested five times, and the best-performing executed trajectories were selected for visual presentation (see Figs. 29 and 30). The corresponding key quantitative metrics for each combination are summarized in Table 9.

During the rescue stairway experiments, only combinations A, G, H, and J completed the navigation task at least once. Our proposed framework J achieved the highest success rate, exhibiting the smoothest executed trajectory and robustness during rapid terrain shifts. For combinations A and B, the graph-search-based Colas' method generated extreme paths that were close to the guardrails. Thus, in combination B, collisions were inevitable. However, in combination A, ManiOpt was able to partially mitigate this issue (with a 20% success rate).

For combination E and F, 3D2M's terrain analysis struggled in environments containing irregular under-surfaces, such as beams on the ceiling, mistakenly classifying them as obstacles. This misinterpretation led to failures in both global path planning and local path optimization. Even initialized with a reasonable global path, the optimized trajectory in combination F tended to deviate from the ground. In combination G, without local optimization, CT-DC alone produced coarse and stochastic global paths, resulting in unsafe slippage on anisotropically traversable staircases.

Combination H, employing Gianni's controller, exhibited inadequate response, causing robot-ground impacts at every first and every final stage of staircase climbing. The two failures of combination H were attributed to the onboard computer crash due to these impacts. Meanwhile, the combination I with GEO required excessively high resolutions in both local path and map representations. In practical online navigation, GEO consistently failed to process valid flipper configuration analysis, resulting in flipper misbehavior and subsequent entrapment of the robot on stair edges, accompanied by high-frequency oscillations of the body.

The spiral stairway provides slightly more navigable space compared to the tightly confined scissor-style rescue stairway. However, it introduces significant challenges due to steep tangential inclinations on the inner arc. In the experiments, four combinations completed at least one navigation test. The proposed framework J again demonstrated the best overall performance. Although the global paths generated by Colas' method in combinations A and B were feasible, they remained risky. Successful navigation was only possible when coupled with ManiOpt to refine the path and avoid collisions.

Due to the refractive properties of glass, parts of the spiral stairway's glass guardrails were missing from the point cloud map. 3D2M failed to detect the hazardous obstructions along both sides of the ground-adjacent staircases, leading its global planner to erroneously produce paths that passed through the missing glass walls, resulting in navigation failure. This failure in obstacle detection also impaired its local trajectory optimization. Even when using global paths generated by CT-DC, 3D2M-L's local optimizer frequently produced trajectories too close to the handrails, which again led to several unsuccessful navigation attempts.

Combination H, which utilized Gianni's controller, also completed the task. However, as the robot tended to follow paths along the outer arc of the stairway, frequent critical impacts (even causing SLAM drift, see Fig. 30(b)) between the robot chassis and the staircase were observed on the larger inter-step distance staircases. That is because this type of control strategy is incapable of accommodating long-term processing and lacks the responsiveness required to adapt to rapidly changing terrains. Combination I with GEO exhibited similar issues to those in the rescue stairway scenario, struggling with the configuration analysis of flippers and failing to maintain smooth and stable motion.

5.6.5 *Unknown construction site*

We conducted a real-world demonstration at an abandoned construction site (see Fig. 31), to validate the framework's ability to navigate using real-time online point clouds generated by the SLAM module in an unknown environment. The site, approximately 1000m^2 in size, featured various obstacles, construction debris, and large traversable areas connected by staircases.

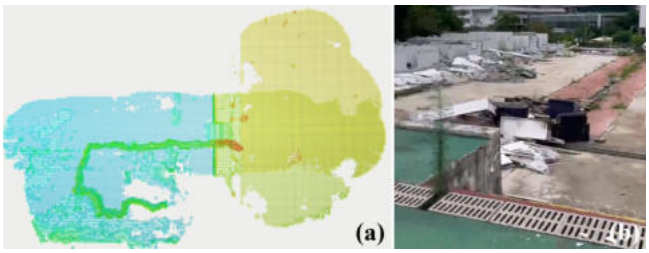


Figure 31. The abandoned construction site: (a) the online point cloud map from the SLAM module and (b) a corner of the abandoned site.

During the test, the robot explored the entire area by following goal points manually assigned in real time. The terrains encountered by the robot were representative of typical rescue environments. The total duration of the demonstration was approximately 10 minutes, and the robot's executed trajectory is illustrated in Fig. 32.

The results show that the proposed framework demonstrated strong stability during online planning tasks in unknown environments and adapted well to real-time point cloud maps provided by the SLAM module. Additionally, the framework exhibited high efficiency in both short-range and long-range planning and maintained robust whole-body control throughout the navigation process, successfully traversing various challenging terrains to complete the mission.

5.6.6 *Discussion on real-world experiments*

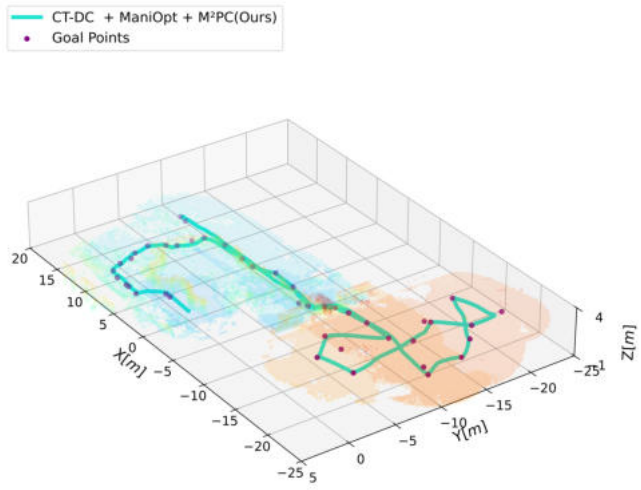


Figure 32. The executed trajectory of the proposed framework in an unknown complex scenario.

Through extensive multi-scenario validations, we systematically demonstrate the synergistic effectiveness and interdependence of the modules within the proposed framework. The global path generation (CT-DC), the local path optimization (ManiOpt), and the holistic control (M^2PC) cooperate efficiently across diverse and challenging environments, where the absence of any single module leads to a significant degradation in overall framework performance. Furthermore, through independent evaluations under unified conditions, we quantitatively verified the rationality of global path generation, the stability of local path optimization, and the robustness of the controller, clearly illustrating the critical contributions of each subsystem to the overall navigation performance. The adaptability of the framework to auxiliary modules (such as the SLAM module) and varying map sources further confirmed the effectiveness of the modular decoupling design and the necessity of systematic integration.

6 Conclusion

This paper presents a novel navigation framework for articulated tracked robots (ATRs) operating in rescue scenarios, leveraging terrain information throughout the planning and control processes. The framework consists of three tightly integrated components: a hierarchical global planner for efficient path generation in large-scale, unstructured environments; a manifold-optimization-based local planner that refines the global path according to kinematic and scenario-specific objectives; and an on-manifold model predictive controller that enables robust centroid path tracking through coordinated whole-body motion on the combined manifold spaces. Together, these components form a cohesive system that addresses the core challenges of ATR navigation across diverse and complex environments.

Extensive simulations and real-world experiments across complex scenarios demonstrate that the proposed methods outperform state-of-the-art methods. In general, the proposed framework consciously separates the notions of completeness and optimality across different levels of planning and control. The hierarchical global planner (CT-DC) emphasizes completeness, ensuring the ability to generate feasible paths in large-scale and complex environments in time, even if the resulting path is suboptimal. This guarantees robust reachability and serves as a reliable initialization for downstream modules, while the downstream local planner and controller can effectively correct suboptimality. In contrast, the manifold-optimization-based local planner (ManiOpt) focuses on improving the quality of the initial path by formulating an optimization problem that integrates multiple objectives. The refined reference path provided to the controller satisfies obstacle avoidance requirements, the robot's kinematic constraints, and various terrain-specific demands. Furthermore, by relying solely on the robot's centroid reference path and its perception of the terrain within the horizon, the proposed M^2PC can achieve whole-body coordinated motion in complex terrains. This design allows the framework to avoid high-dimensional state searches and feasibility checks in the global planning stage and significantly reduces the need for flipper-specific objectives in the local planning stage. In this way, the framework effectively balances completeness and optimality across different levels while carefully designed modularization reduces computational load and system fragility.

Although effective, the proposed framework could benefit from learning-based enhancements of terrain models. In future work, we plan to explore more efficient map representations, such as implicit neural networks, and further extend the framework toward data-driven end-to-end navigation across diverse robot platforms, potentially enabling multi-robot collaborative navigation.

Acknowledgements

Our sincere appreciation to Mr. Lingxu Chen, Mr. Jidong Huang, Dr. Zhihao Wang, Dr. Yu Wang, and any other personnel who assisted in the experiments carried out in this paper.

Supplemental material

The videos that demonstrate the results of our simulations and real-world experiments can be found in the supplementary materials and https://www.bilibili.com/video/BV1WQBfBTE7w/.

References

Azayev T and Zimmermann K (2022) Autonomous state-based flipper control for articulated tracked robots in urban environments. *IEEE Robotics and Automation Letters* 7(3): 7794–7801. DOI:10.1109/LRA.2022.3185762.

Bircher A, Kamel M, Alexis K, Oleynikova H and Siegwart R (2018) Receding horizon path planning for 3d exploration and surface inspection. *Autonomous*

Robots 42(2): 291–306. DOI:https://doi.org/10.1007/s10514-016-9610-0.

Boumal N (2023) *An introduction to optimization on smooth manifolds*. Cambridge University Press. DOI:10.1017/9781009166164. URL https://www.nicolasboumal.net/book.

Cai X, Everett M, Sharma L, Osteen PR and How JP (2023) Probabilistic traversability model for risk-aware motion planning in off-road environments. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 11297–11304. DOI:10.1109/IROS55552.2023.10341350.

Cao C, Zhu H, Choset H and Zhang J (2021) Tare: A hierarchical framework for efficiently exploring complex 3d environments. In: *Robotics: Science and Systems*, volume 5.

Chen B, Huang K, Pan H, Ren H, Chen X, Xiao J, Wu W and Lu H (2023) Geometry-based flipper motion planning for articulated tracked robots traversing rough terrain in real-time. *Journal of Field Robotics* 40(8): 2010–2029. DOI:https://doi.org/10.1002/rob.22236.

Chiu Y, Shiroma N, Igarashi H, Sato N, Inami M and Matsuno F (2005) Fuma: environment information gathering wheeled rescue robot with one-dof arm. In: *IEEE International Safety, Security and Rescue Rototics, Workshop, 2005.* pp. 81–86. DOI:10.1109/SSRR.2005.1501261.

Choi B, Lee W, Park G, Lee Y, Min J and Hong S (2019) Development and control of a military rescue robot for casualty extraction task. *Journal of Field Robotics* 36(4): 656–676. DOI:https://doi.org/10.1002/rob.21843.

Colas F, Mahesh S, Pomerleau F, Liu M and Siegwart R (2013) 3d path planning and execution for search and rescue ground robots. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 722–727. DOI:10.1109/IROS.2013.6696431.

Delmerico J, Mintchev S, Giusti A, Gromov B, Melo K, Horvat T, Cadena C, Hutter M, Ijspeert A, Floreano D, Gambardella LM, Siegwart R and Scaramuzza D (2019) The current state and future outlook of rescue robotics. *Journal of Field Robotics* 36(7): 1171–1191. DOI:https://doi.org/10.1002/rob.21887.

Do Carmo MP (2016) *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications.

Dolgov D and Thrun S (2009) Autonomous driving in semi-structured environments: Mapping and planning. In: *2009 IEEE International Conference on Robotics and Automation*. pp. 3407–3414. DOI:10.1109/ROBOT.2009.5152682.

Faessler M, Fontana F, Forster C, Mueggler E, Pizzoli M and Scaramuzza D (2016) Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics* 33(4): 431–450. DOI:https://doi.org/10.1002/rob.21581.

Fan DD, Otsu K, Kubo Y, Dixit A, Burdick J and Agha-Mohammadi AA (2021) Step: Stochastic traversability evaluation and planning for risk-aware off-road navigation. *arXiv preprint arXiv:2103.02828* .

Gianni M, Ferri F, Menna M and Pirri F (2016) Adaptive robust three-dimensional trajectory tracking for actively articulated tracked vehicles. *Journal of Field Robotics* 33(7): 901–930. DOI:https://doi.org/10.1002/rob.21584.

Hu H, Zhang K, Tan AH, Ruan M, Agia C and Nejat G (2021) A sim-to-real pipeline for deep reinforcement learning for autonomous robot navigation in cluttered rough terrain. *IEEE Robotics and Automation Letters* 6(4): 6569–6576. DOI:10.1109/LRA.2021.3093551.

Hutter M, Gehring C, Lauber A, Gunther F, Bellicoso CD, Tsounis V, Fankhauser P, Diethelm R, Bachmann S, Blösch M et al. (2017) Anymal-toward legged robots for harsh environments. *Advanced Robotics* 31(17): 918–931. DOI:10.1080/01691864.2017.1378591.

Jian Z, Lu Z, Zhou X, Lan B, Xiao A, Wang X and Liang B (2022) Putn: A plane-fitting based uneven terrain navigation framework. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 7160–7166. DOI:10.1109/IROS47612.2022.9981038.

Kalabić UV, Gupta R, Di Cairano S, Bloch AM and Kolmanovsky IV (2017) Mpc on manifolds with an application to the control of spacecraft attitude on so(3). *Automatica* 76: 293–300. DOI:https://doi.org/10.1016/j.automatica.2016.10.022.

Krüsi P, Furgale P, Bosse M and Siegwart R (2017) Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments. *Journal of Field Robotics* 34(5): 940–984. DOI:https://doi.org/10.1002/rob.21700.

Kümmerle R, Grisetti G, Strasdat H, Konolige K and Burgard W (2011) G2o: A general framework for graph optimization. In: *2011 IEEE International Conference on Robotics and Automation*. pp. 3607–3613. DOI:10.1109/ICRA.2011.5979949.

Lee S, Lim H and Myung H (2022) Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3d point cloud. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 13276–13283. DOI:10.1109/IROS47612.2022.9981561.

Lu G, Xu W and Zhang F (2023) On-manifold model predictive control for trajectory tracking on robotic systems. *IEEE Transactions on Industrial Electronics* 70(9): 9192–9202. DOI:10.1109/TIE.2022.3212397.

Mitriakov A, Papadakis P, Kerdreux J and Garlatti S (2021) Reinforcement learning based, staircase negotiation learning: Simulation and transfer to reality for articulated tracked robots. *IEEE Robotics & Automation Magazine* 28(4): 10–20. DOI:10.1109/MRA.2021.3114105.

Okada Y, Kojima S, Ohno K and Tadokoro S (2020) Real-time simulation of non-deformable continuous tracks with explicit consideration of friction and grouser geometry. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 948–954. DOI:10.1109/ICRA40945.2020.9196776.

Okada Y, Nagatani K and Yoshida K (2009) Semi-autonomous operation of tracked vehicles on rough terrain using autonomous control of active flippers. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 2815–2820. DOI:10.1109/IROS.

2009.5354549.

Pan H, Chen B, Huang K, Ren J, Chen X and Lu H (2023) Deep reinforcement learning for flipper control of tracked robots. *arXiv preprint arXiv:2306.10352* .

Perez-Yus A, Gutierrez-Gomez D, Lopez-Nicolas G and Guerrero JJ (2017) Stairs detection with odometry-aided traversal from a wearable RGB-D camera. *Computer Vision and Image Understanding* 154: 192–205. DOI: https://doi.org/10.1016/j.cviu.2016.04.007.

Rösmann C, Hoffmann F and Bertram T (2017) Kinodynamic trajectory optimization and control for car-like robots. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 5681–5686. DOI:10.1109/IROS.2017.8206458.

Schwarz M, Rodehutskors T, Droeschel D, Beul M, Schreiber M, Araslanov N, Ivanov I, Lenz C, Razlaw J, Schüller S, Schwarz D, Topalidou-Kyniazopoulou A and Behnke S (2017) Nimbro rescue: Solving disaster-response tasks with the mobile manipulation robot momaro. *Journal of Field Robotics* 34(2): 400–425. DOI:https://doi.org/10.1002/rob.21677.

Sleiman JP, Farshidian F and Hutter M (2023) Versatile multicontact planning and control for legged loco-manipulation. *Science Robotics* 8(81): eadg5014. DOI: 10.1126/scirobotics.adg5014.

Sriganesh P, Bagree N, Vundurthy B and Travers M (2023) Fast staircase detection and estimation using 3d point clouds with multi-detection merging for heterogeneous robots. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 9253–9259. DOI: 10.1109/ICRA48891.2023.10160258.

Stellato B, Banjac G, Goulart P, Bemporad A and Boyd S (2020) OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation* 12(4): 637–672. DOI:10.1007/s12532-020-00179-2.

Wang C, Meng L, She S, Mitchell IM, Li T, Tung F, Wan W, Meng MQH and de Silva CW (2017) Autonomous mobile robot navigation in uneven and unstructured indoor environments. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 109–116. DOI:10.1109/IROS.2017.8202145.

Wang J, Xu L, Fu H, Meng Z, Xu C, Cao Y, Lyu X and Gao F (2023) Towards efficient trajectory generation for ground robots beyond 2d environment. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7858–7864.

Weerakoon K, Sathyamoorthy AJ and Manocha D (2022) Sim-to-real strategy for spatially aware robot navigation in uneven outdoor environments. *arXiv preprint arXiv:2205.09194* .

Westfechtel T, Ohno K, Mertsching B, Hamada R, Nickchen D, Kojima S and Tadokoro S (2018) Robust stairway-detection and localization method for mobile robots using a graph-based model and competing initializations. *The International Journal of Robotics Research* 37(12): 1463–1483. DOI:10.1177/0278364918798039.

Xu L, Chai K, Han Z, Liu H, Xu C, Cao Y and Gao F (2023) An efficient trajectory planner for car-like robots on uneven terrain. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 2853–2860. DOI:10.1109/IROS55552.2023.10341558.

Xu W, Cai Y, He D, Lin J and Zhang F (2022) Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics* 38(4): 2053–2073. DOI:10.1109/TRO.2022.3141876.

Xu Z, Chen Y, Jian Z, Tan J, Wang X and Liang BL (2024) Hybrid trajectory optimization for autonomous terrain traversal of articulated tracked robots. *IEEE Robotics and Automation Letters* 9(1): 755–762. DOI: 10.1109/LRA.2023.3337593.

Yuan Y, Wang L and Schwertfeger S (2019) Configuration-space flipper planning for rescue robots. In: *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. pp. 37–42. DOI:10.1109/SSRR.2019.8848978.

Zhu Y, Kong Y, Jie Y, Xu S and Cheng H (2023) Graco: A multimodal dataset for ground and aerial cooperative localization and mapping. *IEEE Robotics and Automation Letters* 8(2): 966–973. DOI:10.1109/LRA.2023.3234802.

Şucan IA, Moll M and Kavraki LE (2012) The Open Motion Planning Library. *IEEE Robotics & Automation Magazine* 19(4): 72–82. DOI:10.1109/MRA.2012.2205651.