


RESEARCH ARTICLE

An autonomous navigation approach for unmanned vehicle in outdoor unstructured terrain with dynamic and negative obstacles

Bo Zhou¹ , Jianjun Yi^{1,*}, Xinke Zhang¹, Liwei Chen¹, Ding Yang¹, Fei Han^{2,3} and Hanmo Zhang^{2,3}

¹School of Mechanical and Power Engineering, East China University of Science and Technology, Xuhui District, Shanghai 200237, China, ²Shanghai Key Laboratory of Aerospace Intelligent Control Technology, Shanghai 201109, China and ³Shanghai Aerospace Control Technology Institute, Shanghai 201109, China

*Corresponding author. E-mail: jjyi@ecust.edu.cn

Received: 3 June 2021; **Accepted:** 20 December 2021; **First published online:** 27 January 2022

Keywords: unstructured terrain, negative obstacles, unmanned vehicles, trajectory optimization

Abstract

At present, the study on autonomous unmanned ground vehicle navigation in an unstructured environment is still facing great challenges and is of great significance in scenarios where search and rescue robots, planetary exploration robots, and agricultural robots are needed. In this paper, we proposed an autonomous navigation method for unstructured environments based on terrain constraints. Efficient path search and trajectory optimization on octree map are proposed to generate trajectories, which can effectively avoid various obstacles in off-road environments, such as dynamic obstacles and negative obstacles, to reach the specified destination. We have conducted empirical experiments in both simulated and real environments, and the results show that our approach achieved superior performance in dynamic obstacle avoidance tasks and mapless navigation tasks compared to the traditional 2-dimensional or 2.5-dimensional navigation methods.

1. Introduction

In recent years, the positioning and navigation technology of unmanned vehicles in structured environments has achieved a remarkable result [1]. However, in the scenarios such as planetary exploration, search and rescue robots, and agricultural robots, the ability to navigate and explore autonomously in unstructured environments without GPS, road signs, and artificial landmarks is still the key challenges of field robots [2]. The process from active environment perception and pose estimation back to autonomous navigation need different technologies. In the environment perception and pose estimation task, the simultaneous localization and mapping (SLAM) technology is one of the most effective methods [3], which can be divided into laser-slam [4, 5] and visual-slam [6, 7] according to the difference of sensors. And three-dimensional (3D) LIDAR such as Velodyne VLP-16 is widely used in outdoor environments, as it can not only provide an enormous amount of accurate distance data of 360 degrees around the vehicle, but also is insensitive to the changes in illumination. For autonomous navigation task, the real-time status of the vehicle including position and pose as well as the surrounding environment should be considered. The former can be obtained by SLAM algorithm, while the later should be represented by an appropriate method. Finally, a trajectory planning method is adopted to generate a set of executable and collision-free motion commands to achieve the predetermined goal.

Several surveys and researches related to the navigation of ground vehicles in the unstructured environments have been conducted [8, 9]. As the assumption that robots operate on flat ground is not valid and the terrain obstacles are difficult to determine clearly, the conventional two-dimensional (2D) navigation approach is not suitable anymore. In the unstructured environment, most studies are based on the analysis of 2D elevation maps (the so-called 2.5D maps), where 3D information grabbed from stereo

vision or point clouds is mapped to a 2D occupancy grid map. This approach typically adopted the terrain traversability analysis (TTA) [10] or obstacle avoidance. The TTA refers to how difficulty when a ground vehicle traverses a certain defined terrain, which is deeply different from the obstacle avoidance approach. As in most cases, there may have no so-called obstacles, but the robot must choose a feasible path through the special terrain [11].

Although the TTA based on the 2.5D approach performs well in certain outdoor scenario, the major limitation of these approaches is related to the need of on-board real-time computing. As each cell of the map or the local map must be processed in this method, which limits the ability to re-plan the path to avoid moving obstacles.

Another limitation is the negative obstacles, such as the gentle slope, depression, and holes in the ground. Because of the limitation of field of view of the sensor, there will be a blind area for observation in the lower part of the structure. In this case, a conservative or an aggressive navigation approach will both lead to a failure. The former is prone to fall into false obstacles in the early stages of navigation, while the latter is prone to navigate into the blind area caused by the negative obstacles [12]. In addition, some negative obstacles are possible to pass from a certain direction or at a specific velocity due to the nature of rugged terrain; however, this possibility is ignored in 2D and 2.5D approaches.

To solve the problem mentioned above, this paper proposes an autonomous 3D navigation system suitable for the off-road environment. In this system, an octomap [13] is used to store and maintain the environment map. An efficient octree-based terrain constraint analysis module is designed to assist the sampling-based path search front-end and the passable corridor constraint-based trajectory optimization back-end. Meanwhile, a local navigation target exploration method suitable for sparse terrain information is proposed to perform a mapless navigation with negative obstacles.

1.1. Contributions

Compared with previous researches on unmanned vehicles, the contribution of this paper is to propose a 3D spatial navigation system for unmanned vehicles suitable for complex unstructured terrain environments with the aid of an octomap representation, considered with the performance limitations of unmanned vehicles. This method differs from other 3D navigation and exploration methods in that:

- An efficient front-end path search method and a passable corridor generation method, both of which based on terrain constraints, are proposed to generate safe trajectories. It is worth noting that they are specifically optimized for sparse terrain information in the off-road environments to enable superior dynamic re-planning capability of the proposed navigation method.
- The proposed approach is extended to mapless navigation tasks with negative obstacles by an efficient local target exploration algorithm.
- Our approach is demonstrated to be superior to traditional 2D or 2.5D navigation methods by various empirical experiments in simulations and real-world scenarios.

1.2. Organization

The rest of the paper is organized as follows: the navigation approach, the establishment of the navigation map, and the path planning method are briefly introduced in Section 2. The algorithm architecture and the composition of the hardware system of this paper are summarized in Section 3. In Section 4 and Section 5, we introduce in detail the 3D navigation algorithm for unmanned vehicles proposed in this paper. Experiment and comparison about the proposed framework are presented in Section 6. Finally, Section 7 presents our conclusions and our plans for future work.

2. Related work

There are two main broad categories for autonomous navigation in unstructured environments, namely the sense-plan-act paradigm and end-to-end paradigm [14]. Obviously, these two categories are deeply different. The end-to-end paradigm directly maps the sensor data and the vehicle state information into navigation control actions, which integrating perception and control of navigation frameworks. The deep reinforcement learning method [15] and the deep inverse reinforcement learning method [16] are the most popular methods in this approach. In this case, the reward function is learned from the expert demonstrations or defined by humans, which then is used to generate navigation action sequences. Although this method has many excellent properties, like dealing with negative obstacles [12], they also have some drawbacks as well. The ability of compute in real time for on-board and the generalization ability of the model are the major limitations.

In this paper, we mainly focus on the sense-plan-act paradigm, where TTA is the most important research interest. More specifically, the TTA methods can be divided into two categories: the regression-based method and the classification-based method [11].

The regression-based TTA focuses on estimating the traversal cost of the terrain area. A method is adopted by Wulfmeier et al. to learn a function from the actions of the human experts to represent the traversal cost, which can be optimized by massive well-designed and hand-crafted cost map. However, it only works well in semi-structured scenarios and meanwhile a person is needed to show and teach in similar scenarios [17]. Oliveira utilized deep learning method to predict the traversal cost. Here, inertial measurements gained from IMU and point clouds gained from a 3D LiDAR are used to measure the navigation cost [18]. But he maps the 3D point clouds directly into a 2D grid, making them lose terrain features. The probabilistic energy cost map is proposed by Quann et al. to measure the traversal cost [19], which is built from the pose of robot, the terrain slope, and the satellite imagery. However, this method relies heavily on the vehicle modeling information, which makes it difficult to perform in different type of vehicles.

The classification-based TTA aims at classifying terrain areas into several classes. Many machine learning and deep learning methods are used in this field to perform a binary or multi-classification task, either through the support vector machine [20] or through the multilayer perception [21], which can obtain the geometric and visual information from the 3D LIDAR point cloud and RGB images. Martínez et al. [22] propose traversability classifiers by analyzing several supervised learning methods to extract the classification information from the 3D point cloud. However, a set of spatial features needs to be derived from each point of the acquired point cloud in order to perform terrain classification in those approaches, which makes it time-consuming. Meanwhile, the problem can also be found in terrain classification through the semantic 3D mapping approach [23].

The representation method of the environment is also a research interest in the sense-plan-act paradigm. Theoretically, various map representation ways and technologies can describe the navigation map of vehicles. Maturana et al. [24] build a 2.5D grid map to perform the semantic mapping for automatic off-road driving of all-terrain vehicles, which can provide a richer representation of the environment by encoding the semantic and geometry information extracted from 3D point cloud and RGB images. Noé et al. have used a point cloud to represent navigation maps and have used it for unmanned vehicle navigation tasks in tunnel scenarios [25]. But this approach can only be used in simulation scenarios at present because it cannot solve the interference well because of the disorder of the point cloud in the real-world scenarios and the huge time consumption of searching the point cloud scene. Fei Gao et al. directly used the point cloud in the research of the 3D navigation of unmanned aerial vehicle (UAV), which has made it possible for the use of point cloud in multi-degree-of-freedom navigation [26]. In the research of Helen et al. [27], an incremental 3D Euclidean signed distance field is used to represent the navigation map of UAV. Although it has a great obstacle avoidance performance, it is difficult to apply to the terrain obstacle representation of the unmanned vehicles. Han zhang et al. have used the octree as the representation of the navigation map and have successfully applied it to the outdoor navigation of the UAV. However, it used the remote computer to compute the navigation algorithm, which is difficult to meet the needs of autonomous vehicle navigation task in the field [28].

To sum up, this paper uses the octree to represent the environment, which was chosen for three reasons:

- It can effectively avoid noise impact on navigation in real environment.
- It can effectively represent the terrain information of the off-road environment.
- It can run effectively on the devices with limited computing performance such as unmanned vehicles.

Another important part is the design of the navigation process. In this part, a prevailing idea is to divide the entire navigation process into the front-end path finding and the back-end trajectory optimization. In recent years, there has been a lot of front-end path searching work. Graph-search-based algorithms [29] such as A-star and Dijkstra and sampling-based algorithms [30] such as probabilistic road map and rapidly exploring random tree (RRT) are the two most applicable methods. The method based on graph search can be applied to the scenes where discrete space can be constructed from the environment [31]. In contrast, the sampling-based method can use a set of nodes or other forms to discretely sample the configuration space of the vehicle, and then randomly sample the environment to obtain the final path. RRT [30] is an algorithm proposed by Steven M. LaValle and James J. Kuffner Jr to search quickly non-convex high-dimensional spaces by randomly constructing space filling tree. This algorithm can easily deal with the scenes that contain obstacles and differential motion constraints, so it is widely used in various robot motion planning scenarios. In robotics, RRT search based on high-dimensional space has been widely used [32]. Then, scholars proposed a lot of optimization to RRT. In 2000, Kuffner et al. proposed RRT-CONNECT [33], which is a two-way RRT search process whose speed and efficiency have been greatly improved. Long Chen et al. proposed double-tree RRT* [34], through which the path can be found more quickly, meanwhile its optimality is ensured compared with RRT.

Back-end trajectory optimization has also been widely studied in recent years. Although the front-end pathfinding algorithm can find a set of collision-free waypoints to the target point, it is difficult to perform on robots such as unmanned vehicles. Therefore, it is necessary to generate a smoother, safer trajectory which is a path-time sequence and should conform to the kinematics constraints of the unmanned vehicle based on the path found. Many methods are designed to optimize the trajectory, including the representation of trajectory and the design of the optimization objective function. Moritz [35] is the first to propose the method of representing the trajectory of the Frenet coordinate system, which simplifies the representation style of polynomial trajectories. W. Ding et al. used B-spline curve to represent the trajectory, well using the locality of the spline curve to complete real-time re-planning of the trajectory [36]. Ding et al. proposed a trajectory representation method, which uses the convex hull of the Bernstein polynomial to ensure the safety of the trajectory [37]. In the design of the optimization objective function, the gradient-based method [38] transforms the trajectory optimization problem into a nonlinear optimization problem affected by safety and smoothness. Mellinger et al. proposed the minimum snap trajectory optimization method for the first time and adopted a polynomial trajectory expression to convert the planning problem into a quadratic programming problem [39].

This paper uses the point cloud map and the real-time positioning information generated by the laser SLAM algorithm. In the process of navigation, the environment representation based on the octree map and the front-end path search method based on sampling are adopted. In the process of random sampling of the octree space, the TTA methods are considered. Differently, when using TTA, we do not process each map cell, but only a certain domain of the map cells sampled in the previous step which makes our method highly efficient in planning and re-planning process. Similarly, in the back-end optimization process, a traversable corridor is generated using a local TTA. A high-order Bezier curve is used to characterize the trajectory of the vehicle, as the trajectory can be constrained to a safe area using the convex hull properties of the Bezier curve and the traversable corridor. This method can realize an efficient and autonomous 3D navigation for unmanned vehicles on rugged terrain.

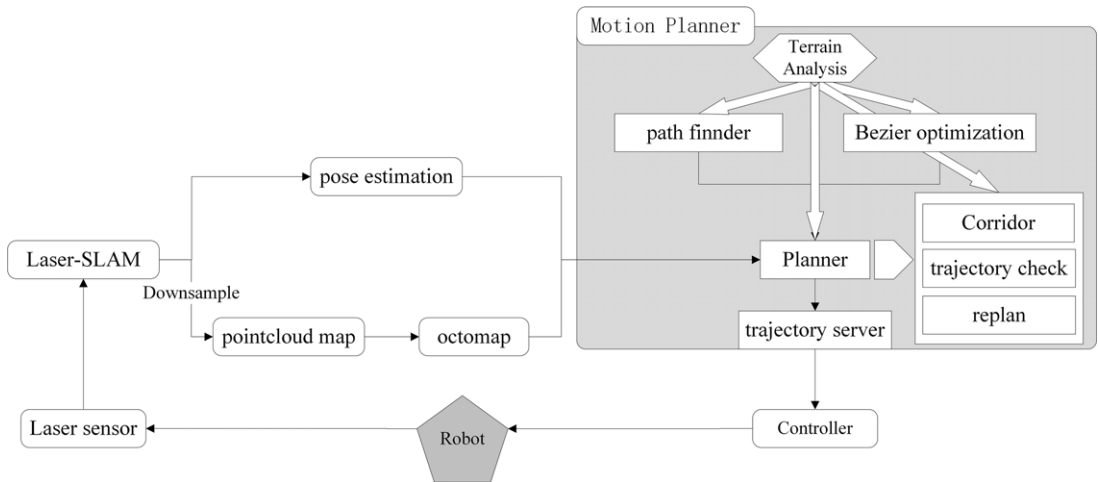


Figure 1. Architecture of the autonomous navigation system.

3. Hardware platform and system architecture

The architecture of our system is shown in Fig. 1. First, an optimized laser-slam – LOAM algorithm [40] is used for six-dimensional pose estimation and sparse point cloud map construction. In this method, a loop detection method based on location [41] and based on scanning context [42] is combined with the LOAM algorithms to reduce the mapping and the pose estimation error of the vehicle. Then, the navigation process is divided into front-end path search and back-end trajectory optimization. The planner module manages the entire system, whose function includes data reception, task allocation, corridor generation, trajectory inspection, and re-planning. It receives the environment navigation maps and the pose estimation from the vehicle as well as the goal given by the mission planner. Then it assigns the calculation task to the path finder module and Bezier optimization module. After that, it analyzes the calculation results and passes them to the trajectory server module.

The path finder module is responsible for searching front-end paths and mapless navigation strategies. This part will be introduced in detail in Section 4. The Bezier optimization module will complete the trajectory optimization process based on the results of the front-end search and the constraints given by the planner module, which will be introduced in detail in Section 5.

The functional module terrain analysis handles the ground state analysis of the entire system and is referenced in the above three modules. This part will be described in detail in Section 4.2.

The architecture of the unmanned vehicle system is shown in Fig. 2. The system uses a four-wheel differential drive unmanned vehicle, which has individual wheel suspension system with good outdoor off-road capabilities and shock absorption performance. The 3D LIDAR Velodyne VLP-16 is used for real-time pose estimation and map construction of the vehicle. And a NVIDIA Xavier processor is used for positioning, mapping, navigation, and control algorithm. In addition, for safety in testing task, the depth camera Azure Kinect is used in this system as an emergency obstacle avoidance sensor.

4. Path planning method based on terrain constraints

4.1. Sampling space configuration

This paper uses the sampling method in octree space so that a high operation efficiency can be gained in embedded devices. Different from the path planning of drones or robotic arms, it is difficult for us to plan the movement of the vehicle on the z-axis in the path planning task in the rugged ground. However, this kind of movement is critical based on the analysis above. In this paper, we adopted a random sampling strategy to find a feasible front-end path for the ground vehicle in the octomap. The sampling space is

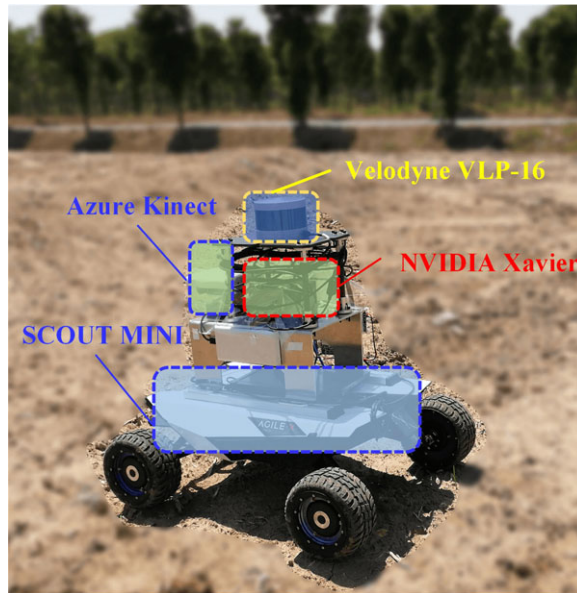


Figure 2. Overview of the unmanned vehicle system. The platform uses a Velodyne LIDAR (Velodyne VLP-16) for localizing itself and mapping the environment. The Laser-SLAM module, motion planning module, and controller are all running on the NVIDIA Xavier. And a camera (Azure Kinect) is used to avoidance in an emergency.

set as the section of the initial height of the vehicle in the octomap instead of the entire octree space. The terrain constraints are introduced into the sampling process, including ground search and the TTA. For each sample point, a ray is used by the algorithm to project upward or downward to find the ground. This method greatly reduces the sampling time and avoids the loss of terrain information in 2D planning at the same time.

$$\begin{cases} Node_{x,y,z} = \Gamma(p_{x,y}), \text{ if } \Gamma(p_{x,y}) \text{ true} \\ None = \Gamma(p_{x,y}), \text{ if } \Gamma(p_{x,y}) \text{ false} \end{cases} \quad (1)$$

where $Node_{x,y,z}$ is the final sampled point. $p_{x,y}$ is the random sampling point on the initial height section of the vehicle, and Γ means the mapping of ground search. If the search is successful, it returns the Node value, otherwise it returns None.

4.2. Path planning method

Because of the sparsity of the octomap, for each sampling point collected by the planner, the local octree nodes in a certain area around it will be analyzed, which can be a sphere, a cube, or a polygonal cylinder. Considering the shape of the vehicle, a cube is adopted as the candidate area. The parameters $box(x,y,z)$ are related to the size of the vehicle. The search set Ω of the sampling point $p_{x,y}$ can be written as follows:

$$\Omega = \left\{ \begin{array}{c} \Gamma(p_{ij}) \\ x - box_x/2 - padding \leq i \leq x + box_x/2 + padding \\ y - box_y/2 - padding \leq j \leq y + box_y/2 + padding \end{array} \right\} \quad (2)$$

Terrain reliability analysis: Generally, in the search set Ω around the sampling point, the more octree nodes with ground elevation information there are, the higher reliability of the sampling point at

the ground and the higher accuracy of the ground height calculation are. So the reliability of the terrain is calculated as follows:

$$\begin{aligned} \eta &= n_{node} / n_{max} \\ n_{max} &= card(\Omega) \\ n_{node} &= card(\{x | x \in \Omega, x \neq None\}) \end{aligned} \tag{3}$$

where n_{max} is the number of all the nodes in the search set Ω , and n_{node} is the number of nodes with ground elevation information.

Terrain flatness analysis: According to the analysis above, in the path planning task of the vehicle, it is important to analyze the ruggedness of the terrain. We use the elevation variance of the ground nodes to describe the terrain flatness:

$$\begin{aligned} mean &= \sum Node_z / n_{node}, Node \in \Omega \\ stddev &= \|Node_z - mean\|_2 / (n_{node} - 1), Node_z \in \Omega \text{ and } Node \neq None \end{aligned} \tag{4}$$

Terrain slope analysis: In the calculation of slope, it will take a lot of time to calculate the global gradient directly, as the constructed octomap may be discontinuous, which is determined by the sparsity of the 3D LIDAR. Furthermore, both the 2D and 2.5D methods have the gradient calculation along with the x and y direction of the navigation map, which makes traverse from a certain direction impossible when the negative obstacles exist. This paper calculates the terrain slope between the new node that is added to the path and its parent-node, instead of analyzing the terrain slope of each sampling node. In this way, it will not only reduce the cost of calculation but also provide the possibility of traversing negative obstacles from a certain direction.

$$\theta = ac \sin \left(\frac{|Node_{newz} - Node_{nearstz}|}{\|Node_{new} - Node_{nearst}\|_2} \right) \tag{5}$$

A cost function is designed to evaluate whether a new node can be added to the path and whether the relationship of path node should be changed. In this paper, we can assume that the cost function for each node can be expressed as a weighted linear combination of a set of weights and some terrain feature functions. And it can be defined as:

$$c(Node_n) = \omega_1 \sum_{i=0}^n \|Node_i - Node_{father}\|_2 + \omega_2 \sum_{i=0}^n \theta_i \tag{6}$$

Therefore, the steps of the 3D path sampling algorithm in octomap space based on the terrain analysis are shown in Algorithm 1.

4.3. Extension to mapless navigation

The autonomous navigation without a prepared map is crucial. As in most of the off-road unmanned vehicle navigation scenarios, there is no priori map information. In the case of mapless navigation, the terrain information around the navigation target point is unknown, and the map information observed in the early stage is very sparse, which is determined by the sparsity of the LiDAR point cloud and the occlusion relationship of the off-road terrain. To implement the proposed method to mapless navigation tasks, we keep exploring toward the target by selecting suitable local target points among the existing sampling points based on the Algorithm 1. Due to the number of sampling points during the path finder process may be large and only the outermost of them are beneficial for the task of exploration toward the target. Therefore, in order to avoid unnecessary calculations and select suitable local target points, we propose the following approach as shown in Algorithm 2 and Fig. 3.

Algorithm 1: Path Finder Module

Input: threshold of the reliability of terrain η_{th} , threshold of the flatness of terrain $stddev_{th}$,
 threshold of the slope of terrain θ_{th} , size of the UGV σ , inflate step size ε , maximum
 number of iterations $iter_{max}$

Output: front-end path $V = \{ \}$

```

1  $V \leftarrow \{Node_{init}\} \leftarrow \{P_{init}, score\}$ ;
2 while  $iter \leq iter_{max}$  do
3    $iter \leftarrow iter + 1$ ;
4    $P_{new_{x,y}} \leftarrow sampleAndNew\{V, P_{goal}, m, \varepsilon\}$ ;
5    $Node_{x,y,z} \leftarrow mapProject\{P_{new_{x,y}}, m\}$ ;
6   if  $Node_{x,y,z} \neq None$  then
7      $\Omega \leftarrow searchSetGen\{V, \sigma, padding\}$ ;
8      $(\eta, stddev) \leftarrow terrainAnalysis\{V, \Omega\}$ ;
9     if  $\eta > \eta_{th}$  and  $stddev < stddev_{th}$  then
10       $\theta \leftarrow terrainAnalysis\{V, \Omega, Node_{x,y,z}\}$ ;
11      if  $\theta \leq \theta_{th}$  then
12         $c(Node_n) \leftarrow cost\{V, Node_{x,y,z}, \theta, \eta, stddev\}$ ;
13         $V \leftarrow updatePath\{V, r\}$ ;
14      else
15        continue;
16      end
17    else
18      continue;
19    end
20  else
21    continue;
22  end
23  if  $\|Node_{x,y,z}, P_{goal}\|_2 < 0.1$  then
24    return  $V$ ;
25  end
26 end

```

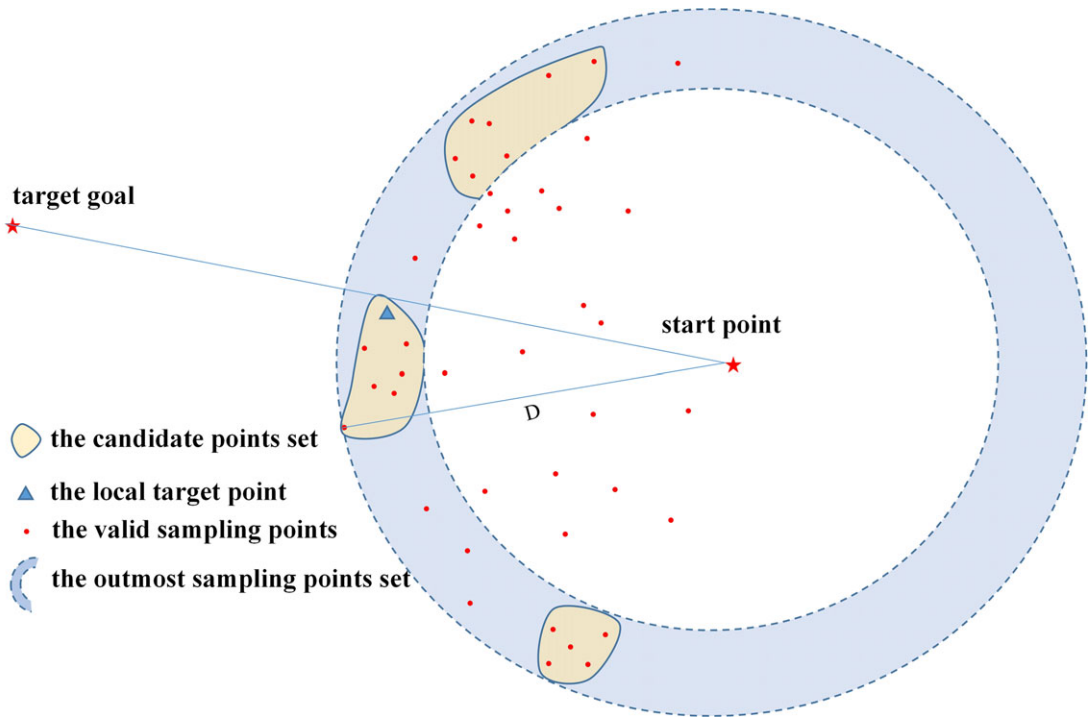


Figure 3. Local target point selection.

Algorithm 2: mapless navigation

Input: valid sampling points set Ω

Output: selected local target point $P_{localGoal}$

```

1 if PathFinderModule is failed then
2    $\Omega_{outer} \leftarrow getOutermostSamplingPoints\{\Omega\}$ ;
3    $\Omega_{cluster} \leftarrow clustering\{\Omega_{outer}\}$ ;
4    $\Theta_{CandiPts} \leftarrow findMostNum\{\Omega_{cluster}\}$ ;
5    $P_{localGoal} \leftarrow findMinCost\{\Theta_{CandiPts}\}$ ;
6   return  $P_{localGoal}$ 
7 end

```

Outermost sampling points set: we define d_i as the distance between i_{th} point(P_i) in Ω and the start point, and the D as the distance between the closest point to the end point and the start point. The outermost sampling points set can be written as follows:

$$\Omega_{outer} = \{P_i | P_i \in \Omega, 0.8 < d_i/D < 1\} \tag{7}$$

Points clustering: We perform clustering of the outermost sampling points using a method similar to that in ref. [25], which can divide the outermost region into several independent regions. As mentioned

in Section 4.2, the sampled points are the points that meet the TTA. Therefore, a region with more sampling points has better traversability, where the candidate local target point should also come from.

The local target point: In order to avoid local target points guiding the vehicle away from the direction of the final target, we finally select three sets in $\Omega_{cluster}$ with the largest number of sampled points to form the final target candidate points set $\Theta_{CandiPts}$. And a cost function is set to select the final local target point in the set of candidate target points. We seek for a point which can minimize the function as shown in the following equation:

$$P_{localGoal} = \operatorname{argmin}(\|P_i - P_{goal}\|_2 + c(P_i)), P_i \in \Theta_{CandiPts} \quad (8)$$

where $c(P_i)$ is the cost of the point P_i to start point defined in Eq. (6).

5. Path optimization method

Although the path finder module can get a path to the target in real time, the kinematics and the dynamics models of the vehicle are not considered in the planning process, which makes the planned path difficult to execute for vehicles in real environment. Based on the analysis above, there has been a lot of research on the trajectory optimization in structured scenarios. This paper focuses on the trajectory optimization methods in the unstructured and rugged terrain. We take the minimum snap as the objective function to generate a smooth trajectory and build a corridor based on the terrain to constrain the final optimized trajectory.

5.1 Traversable corridor generation

Based on the waypoints obtained in Section 4, we analyze the traversable state of the area around the waypoints to determine whether there are steep slopes and pits around the sampling point, thereby providing reasonable constraints for trajectory optimization. According to the searched waypoints, we initialize a cube which is based on the size of the vehicle and then expand the cube in the order of X-Y-Z with a fixed step ε . The expansion scales of $\varepsilon_x, \varepsilon_y$ are equal, but the expansion in the z direction ε_z is far less than the $\varepsilon_x, \varepsilon_y$. This is because it is primarily designed to eliminate collisions between the vehicle and obstacles on the Z-axis caused by the undulations of the terrain. Then it maximizes the cube by analyzing the collision in Z direction and the corresponding terrain state information in X and Y directions. The TTA method is similar to that in Section 4.2. After that, we reduce the number of traversable corridors by merging overlapping areas to reduce the computational burden of back-end optimization. And the traversable corridor generation is shown in Algorithm 3.

5.2. Trajectory optimization

In this paper, the trajectory will be expressed using a Bernstein polynomial [43], which can be written as follows:

$$P_j(t) = \sum_{i=0}^n p_{i,j} \cdot B_{i,n}(t), t \in [0, 1] \quad (9)$$

$$B_{i,n}(t) = C_n^i \cdot t^i \cdot (1-t)^{n-i} = \frac{n!}{i!(n-i)!} \cdot t^i \cdot (1-t)^{n-i}, i = 0, 1, 2, \dots, n$$

where $p_{i,j}$ is the i th control point of the j th Bezier curve, and n is the order of Bezier curve. The trajectory is divided into m segments according to the number of corridors in Section 5.1. Since the time period of the Bezier curve is $[0, 1]$, a time scale factor r needs to be set to realize any time allocation of this

Algorithm 3: CorridorInflate

Input: front-end path ϕ , size of the UGV σ , inflate step size ε , maximum number of iterations

$iter_{max}$

Output: corridor $C = \{ \}$

```

1  foreach  $i = 0, \dots, \text{sizeof}(\phi)$  do
2       $c_{init} \leftarrow \text{generateCubebyPath}(\phi, m)$ ;
3      while  $iter \leq iter_{max}$  do
4          foreach  $\{dir \in \{X_+, X_-, Y_+, Y_-, Z_+\}\}$  do
5               $c \leftarrow c_{init}$  inflate  $\varepsilon$  along  $dir$ ;
6              if ! $\text{validGround}(c, m)$  and  $\text{obstacleFree}(c, m)$  then
7                  break;
8              end
9          end
10     end
11     if  $\text{isContains}(c, c_{last})$  then
12         break;
13     end
14      $C \leftarrow c$ ;
15     return  $C$ ;
16 end

```

segment of the curve. And the piecewise Bezier curve can be represented using the following equation:

$$P^\mu(t) = \begin{cases} r_1 \cdot \sum_{i=0}^n P_{i,1}^\mu B_{i,n}(\frac{t-T_0}{r_1}), & t \in [T_0, T_1] \\ r_2 \cdot \sum_{i=0}^n P_{i,2}^\mu B_{i,n}(\frac{t-T_1}{r_2}), & t \in [T_1, T_2] \\ \vdots \\ r_m \cdot \sum_{i=0}^n P_{i,m}^\mu B_{i,n}(\frac{t-T_{m-1}}{r_m}), & t \in [T_{m-1}, T_m] \end{cases} \tag{10}$$

where T_1, T_2, \dots, T_m is the end time of each segment trajectory. $p_{i,m}^\mu$ is the i th control point of the m th segment. And μ is one of the three dimensions of X, Y, Z . r_1, r_2, \dots, r_m represents the time scale factor of each segment ($r_i = T_i - T_{i-1}, i = 0, 1, 2, \dots, m$), which scales the segment time from $[T_{i-1}, T_i]$ to $[0, 1]$.

Based on the analysis, this paper uses the minimum snap trajectory optimization method, and the cost function of the optimization can be written as:

$$C = \sum_{\mu \in \{x,y\}} \int_0^T \left(\frac{d^4 P^\mu(t)}{dt^4} \right)^2 dt \tag{11}$$

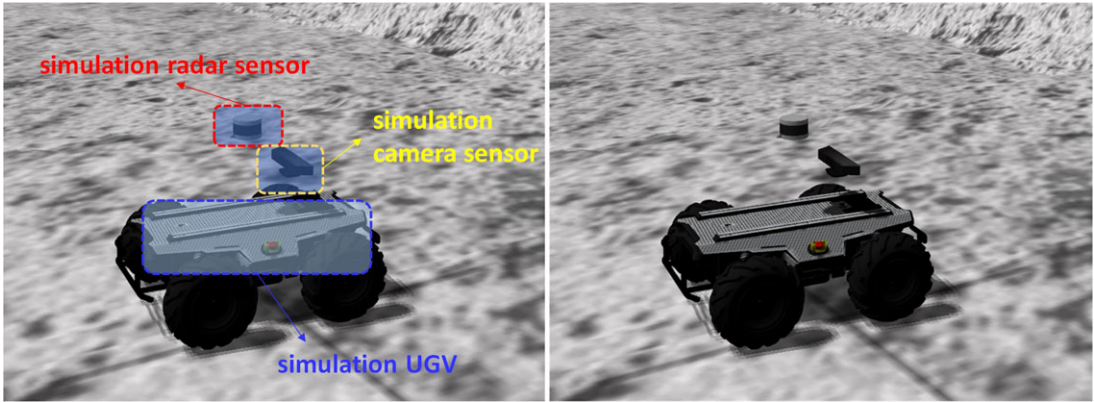


Figure 4. The simulation UGV and the simulation sensors. In our experiment, a simulation Velodyne VLP-16 and a simulation depth camera whose data format is same as that of the real sensors are used (in order to better represent the pose between the sensors, the connection between them is hidden in the simulation).

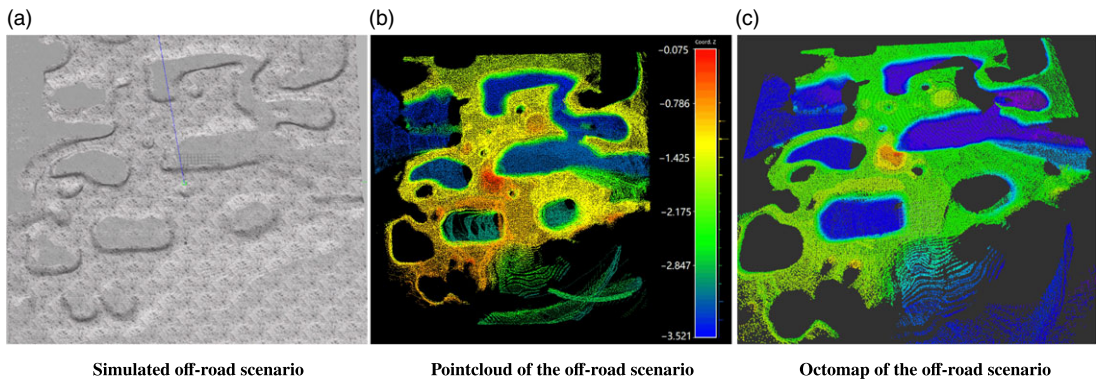


Figure 5. Off-road environment built in gazebo.

It can be represented as a quadratic matrix of $P^T Q_0 P$, where \mathbf{P} is the vector composed of all control points, and Q_0 is the Hessian matrix of the objective function. It can be proved that Q_0 is a positive semi-definite matrix, so the trajectory optimization problem can be transformed into a convex quadratic programming problem.

$$\begin{aligned}
 & \min \mathbf{P}^T Q_0 \mathbf{P} \\
 & s.t. \ A_{eq} \mathbf{P} = b_{eq} \\
 & \quad A_{ieq} \mathbf{P} \leq b_{ieq}
 \end{aligned} \tag{12}$$

For each segment of the Bezier curve, we must set a series of constraints to ensure the safety and feasibility of the optimized trajectory.

Position constraints: The optimized trajectories must pass through some specific points whose positions, velocities, and accelerations are known during the planning process, such as the start point and the goal point. According to the properties of the Bezier curve, the coefficients of the high-order derivative can be composed of the coefficients of the lower-order derivative. Take the Bezier curve in a certain

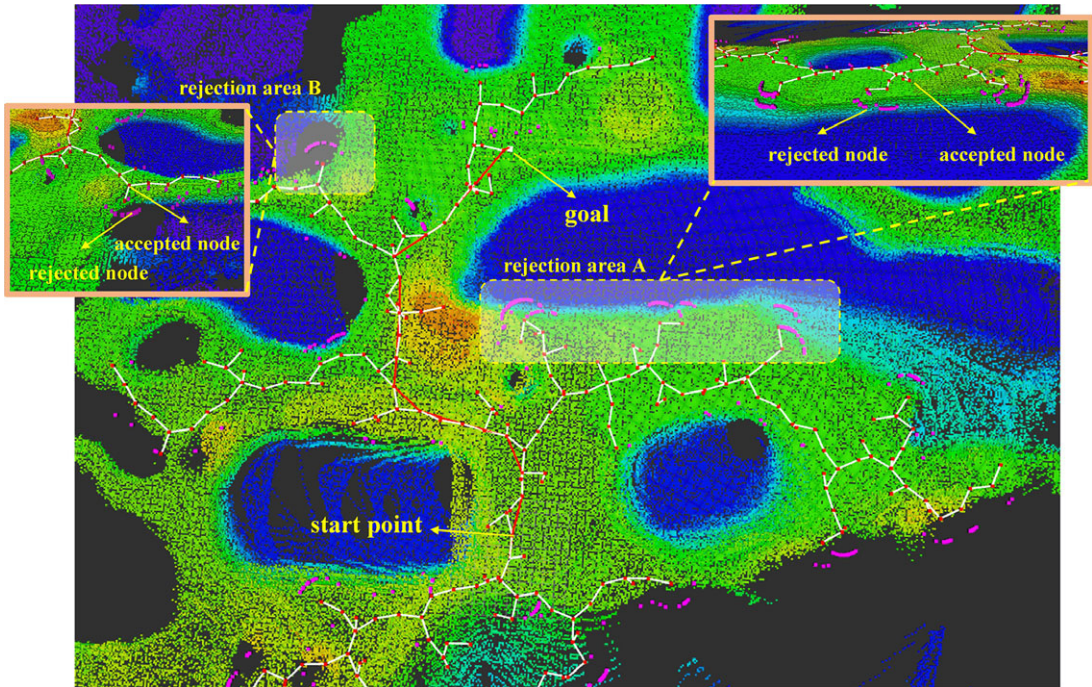


Figure 6. Unknown area, negative obstacle rejection.

dimension of a certain segment as an example:

$$\begin{aligned}
 a_i^{(0)} &= p_i \\
 a_i^{(l)} &= \frac{n!}{(n-l)!} \cdot (a_{i+1}^{(l-1)} - a_i^{(l-1)})
 \end{aligned}
 \tag{13}$$

where l is the order of derivative of the Bernstein basis. Therefore, in a certain dimension of a certain segment, the position constraint can be represented as:

$$a_i^{(l)} \cdot r^{(1-l)} = \gamma^{(l)}
 \tag{14}$$

where $\gamma^{(l)}$ can be the position, velocity, or acceleration of a specific waypoint.

Continuity constraints: Continuity constraints are mainly required for the connection points of each segment trajectory. In order to ensure the smoothness of the trajectory and the continuity of the movement, it is necessary to ensure that the l -order derivative of the connection point is continuous, where $0 \leq l \leq k - 1$. For the j th and $(j + 1)$ th segment trajectories, the last control point of the j th segment is continuous with the first control point of the $(j + 1)$ th segment in the l th order derivative.

$$\begin{aligned}
 a_{nj}^{(l)} \cdot r_j^{(1-l)} &= a_{0j+1}^{(l)} \cdot r_{j+1}^{(1-l)} \\
 a_{i,j}^{(0)} &= p_{i,j}^{(0)}
 \end{aligned}
 \tag{15}$$

Corridor constraints: In the unmanned vehicle trajectory optimization, it is necessary to ensure the safety and feasibility of the optimized trajectory. The usual approach is to process the optimized trajectory for trajectory checking and impose additional constraints on waypoints that do not meet the safety or enforceability requirement. The trajectory optimization and trajectory check process are then repeated until all waypoints on the trajectory meet safety and enforceability requirements [44]. This paper uses the convex hull property of the Bezier curve to enforce all the control points of the trajectory to be constrained within the boundaries of each direction of the corridor which is mentioned in

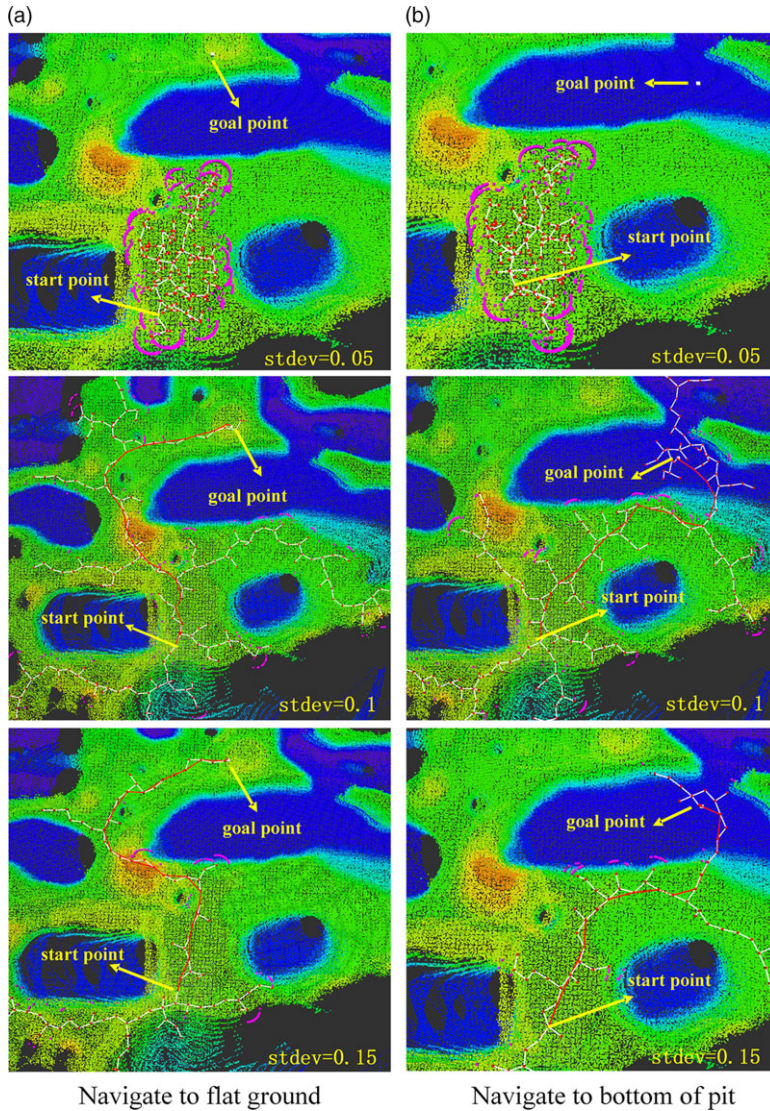


Figure 7. Result of different path-expansion boundaries and cost functions.

Section 5.1. For example, for the j th trajectory:

$$\chi_{i,j}^{\mu-} \leq p_{i,j}^{\mu} \leq \chi_{i,j}^{\mu+}, \mu \in \{x, y, z\}, i = 0, 1, 2, \dots, n \tag{16}$$

where $\chi_{i,j}^{\mu-}, \chi_{i,j}^{\mu+}$ are the upper and lower boundary of the j th corridor in a certain dimension of x, y, z .

Kinematic constraints: Generally, considering the feasibility of the unmanned vehicle movement, its speed and acceleration need to be constrained within a certain workable interval. For the i th control point:

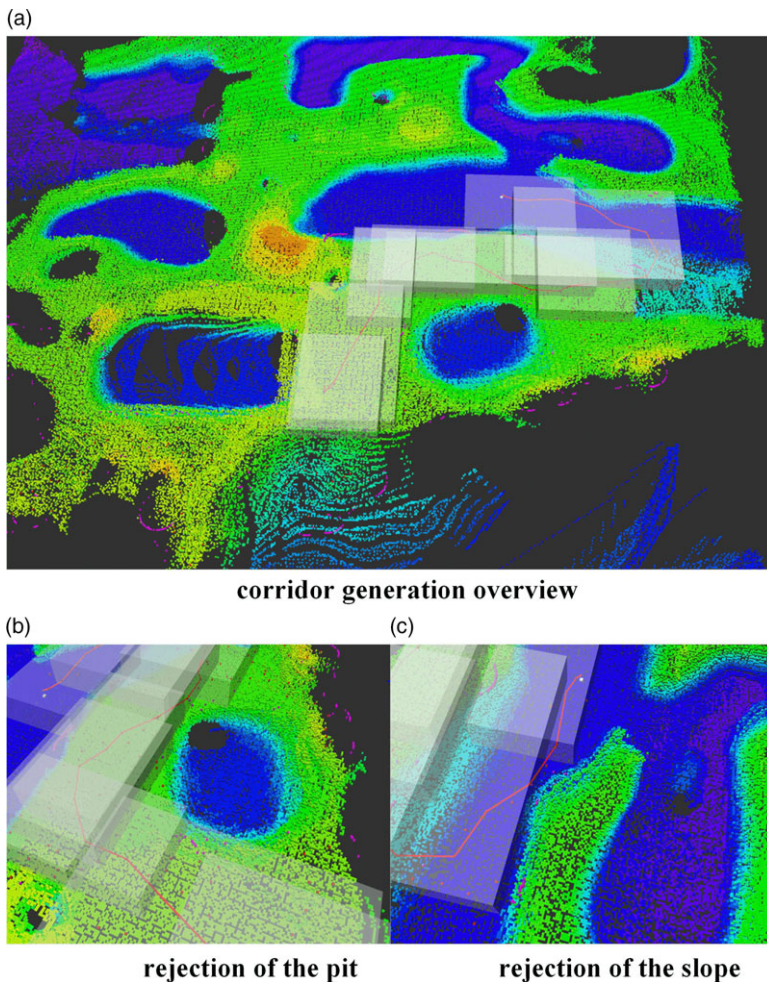
Based on the property of the Bezier curve: $a_i^{(l)} = \frac{n!}{(n-l)!} \cdot (a_{i+1}^{(l-1)} - a_i^{(l-1)})$

$$\begin{aligned} V_{\min} &\leq n(p_{i+1} - p_i) \leq V_{\max} \\ a_{\min} &\leq n \cdot (n - 1) \cdot (p_i - 2p_{i-1} + p_{i-2})/r_j \leq a_{\max} \end{aligned} \tag{17}$$

where $V_{\min}, V_{\max}, a_{\min}, a_{\max}$ are the minimum and maximum speed and acceleration of the vehicle.

Table I. Time of path finder module process ($\eta_{th} = 3$)

Parameters	Navigate to flat ground	Navigate to bottom of pit
Goal	(29.7028, -22.963, -2.09698)	(31.6588, -20.2179, -2.1243)
Stdev=0.05	/	/
Stdev=0.1	187.878 ms.	179.499 ms.
Stdev=0.15	46.2344 ms.	46.2344 ms.

**Figure 8.** Corridor generation result.

6. Experiments and results

We use robot operating system [45] and C++ 11 to implement the proposed system and algorithm in both simulation and real-world scenarios and compare them with popular 2.5D method in mapless navigation tasks and dynamic obstacle avoidance tasks. The qpOASES (Online Active Set Strategy) is used as the convex optimization solver in the trajectory optimization stage.

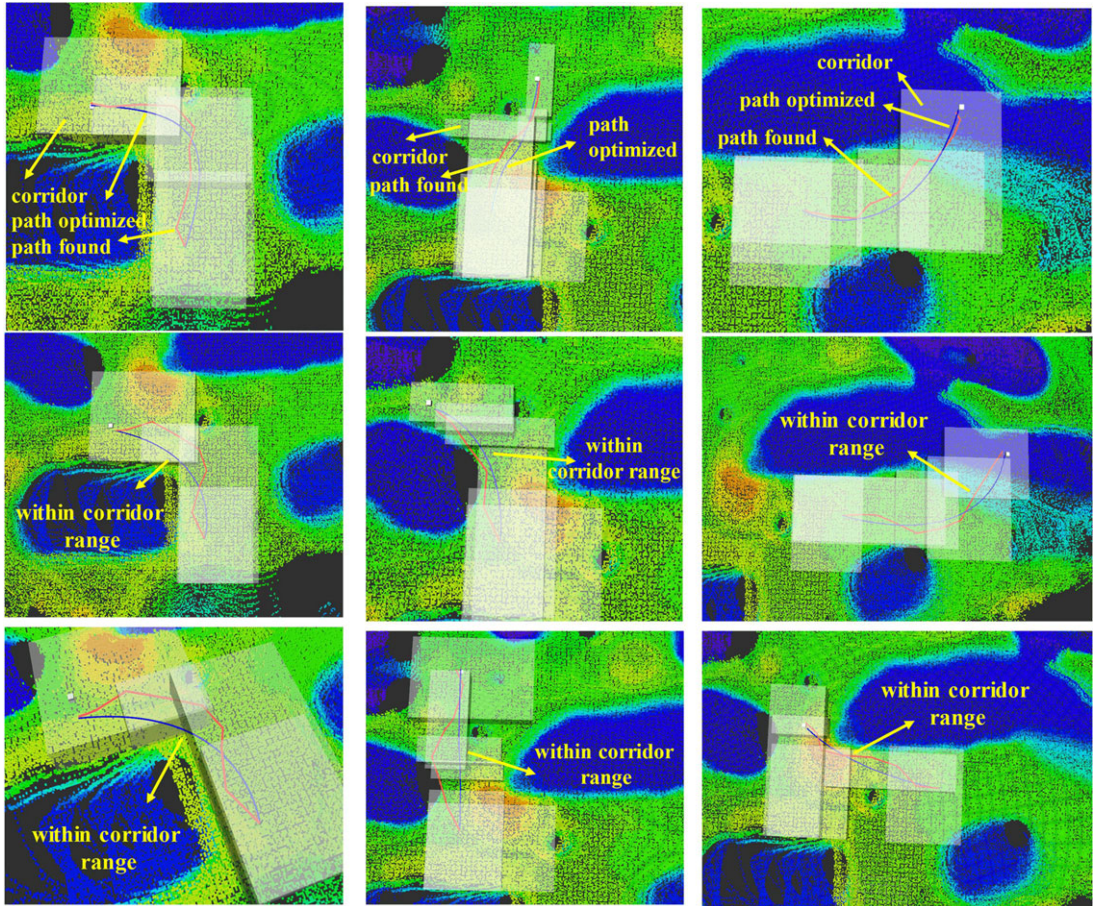


Figure 9. Trajectory generation results.

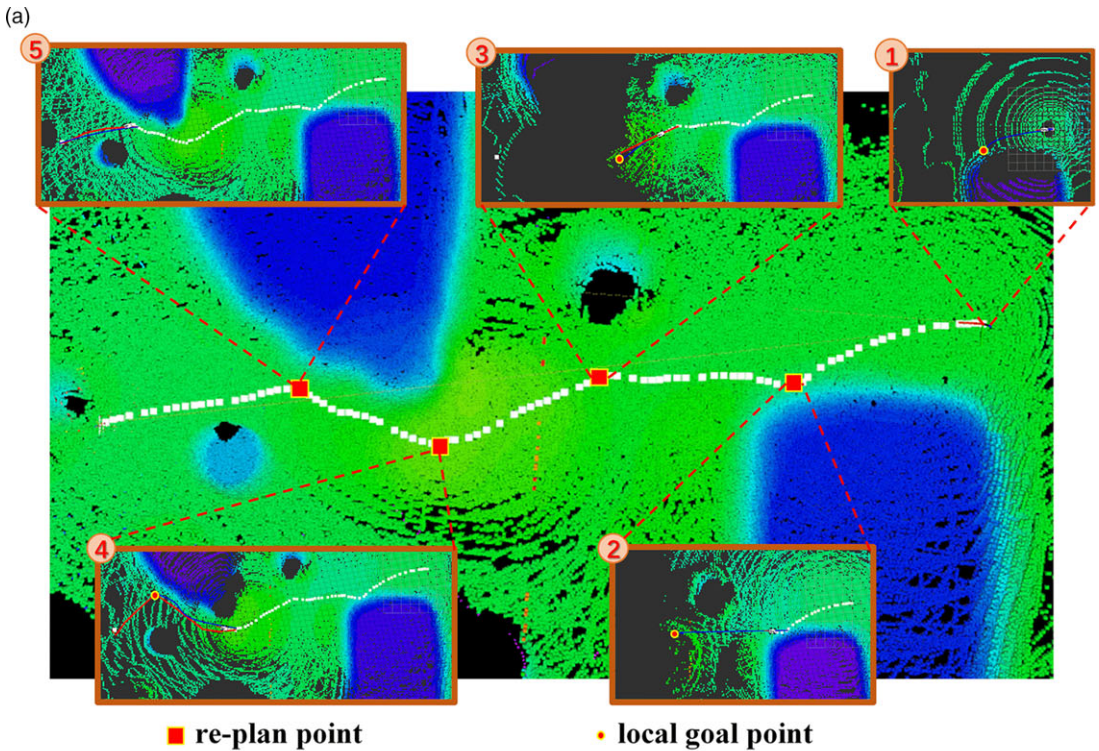
6.1. Simulation experimental setup

6.1.1 Simulation environment

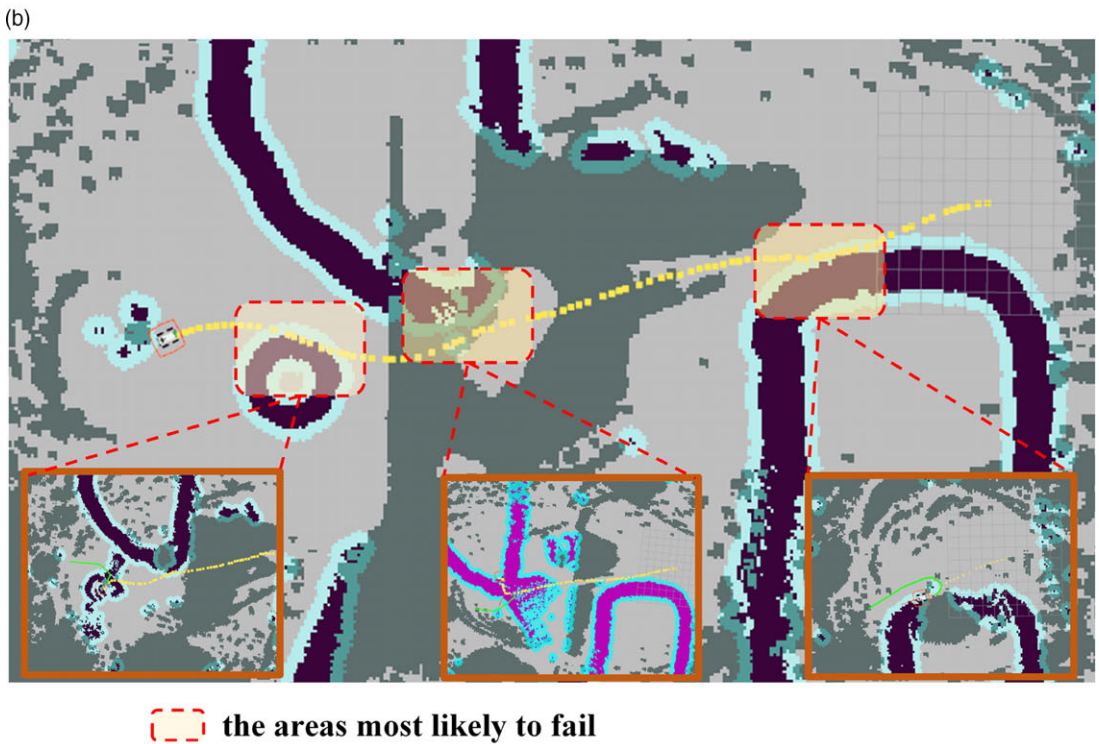
First, we present several simulation experiments of the proposed local target selection, path search, and trajectory optimization algorithms using Gazebo. In order to make the simulation more meaningful and simplify the development and debugging process in the real environment, in this paper a URDF unmanned vehicle simulation model is used, whose dynamic performance and motion restrictions are similar to those of real vehicles. A 3D LIDAR and binocular camera are added to the model for context awareness, as is shown in Fig. 4. An off-road environment is built in the software platform **UNITY** and then imported into Gazebo to gain gravity and physical properties, as is shown in Fig. 5(a). The size of the environment is 200 m \times 200 m, with multiple pits and steep slopes of different diameters distributed, and the maximum undulation of the terrain is about 3 m.

6.1.2. Result of the proposed method

Octomap building: The laser slam algorithm is used to build the sparse point cloud map of the constructed off-road scenario and get the six-dimensional pose estimation of the vehicle. In order to reduce the interference of the real field environment on the point cloud and the huge computational cost of navigation directly on the point cloud map, the map representation method of the octomap is adopted in this paper. Point cloud within a certain range centered on the vehicle location will be added to the



An example of mapless navigation with proposed method



An example of mapless navigation with GridMap method

Figure 10. Comparison results with the GridMap method in mapless navigation task.

Table II. Navigation time distribution

Item	Start point	Goal point	Path finder time (ms)	Corridor generation time (ms)	Trajectory generation time (ms)
1	(0,0,0)	(13.799, 8.528, 0.275)	49.4692	94.7934	6.15753
	(0,0,0)	(16.879, -11.780, 0.275)	111.831	84.3121	3.55139
	(0,0,0)	(18.891, -18.532, -0.863)	91.2956	80.8881	3.98212
2	(17, 12, -0.4)	(33.812, 7.228, -0.025)	160.776	105.67	7.00782
	(17, 12, -0.4)	(30.205, 19.921, -0.025)	205.101	102.424	16.119
	(17, 12, -0.4)	(39.474, 12.839, -0.890)	104.207	100.128	11.3094
3	(17, -9, -0.4)	(27.595, -25.934, -1.412)	74.6397	106.143	6.2942
	(17, -9, -0.4)	(25.825, -34.094, -1.351)	140.462	125.864	9.49656
	(17, -9, -0.4)	(24.136, 10.18, 0.425)	92.5626	91.1293	5.85881

Table III. Comparison results with the GridMap method in mapless navigation task

Method	Navi. time (s)	Traj. Len. (m)	Re-planning times	Success rate (%)
GridMap method	53.4	38.5616	18	8/20(40.0)
Proposed method	43.6	42.1011	4.5	18/20(90.0)

octomap where the terrain information will be maintained and updated, which allows the vehicle to have lower map maintenance costs when real-time navigation tasks are performed. And we use the color to distinguish the elevation, as is shown in Fig. 5(c). The global map coordinate system is established based on the location of the centroid when the vehicle is started. A warmer color shows a higher elevation of the terrain, indicating a larger value of Z-axis, which can be slopes. Conversely, colder colors refer to lower elevation, which can be pits.

Path finding result: In this paper, we analyze the terrain information effectively based on the octomap, which described in Section 4, and evaluate the front-end path search method using the simulation environment mentioned above. We set the navigation destinations at the bottom of the pit or a relatively flat area and change the boundary threshold parameters to test the efficiency of the path finding algorithm and the feasibility of the path.

By setting the boundary threshold reasonably, the path searched by the front-end pathfinder module performs well in avoiding unknown areas, negative obstacles like pits, and steep slopes. Figure 6 shows the successful front-end path finding process, where the cube is the sampled node. The white line is the sampled path, and the red line is the final searched path. As shown in the picture, rejection area A is an example of avoiding negative obstacles like pits and steep slopes. The red nodes are the accepted nodes during the path search process, and magenta nodes are the rejected nodes which do not match the terrain analysis results during the process. After the terrain analysis, some nodes that cannot be safely reached by the vehicles will be refused to join the path, thereby ensuring the successful avoidance of the dangerous areas such as pits and steep slopes. Rejection area B shows an example of avoiding unknown areas. They are some areas that cannot be observed by some sensors, where there may be pits and occlusions. Considering the safety of the car, the algorithm adopts an evasion strategy.

Figure 7 shows the effect of different boundary thresholds on path search results, while Table I shows the search time. From the results, it can be seen that if the threshold of $stddev_{th}$ is too small, the path will be difficult to extend and the search time will be greatly increased at the same time, such as $stddev = 0.05$ in Fig. 7. But when the threshold becomes larger, the path becomes smoother and the search time reduces.

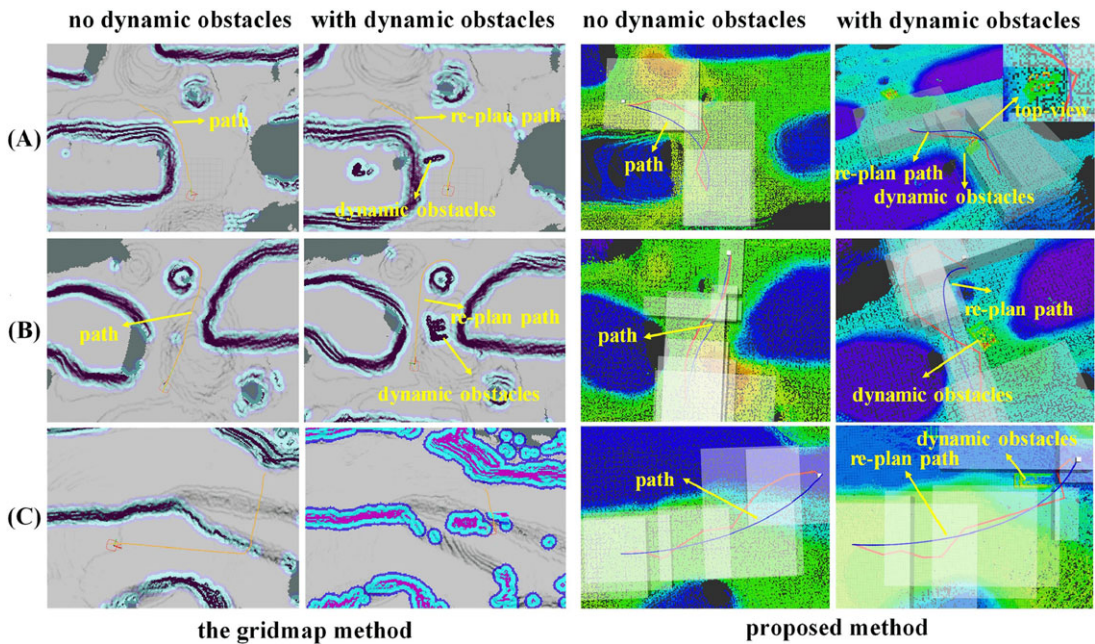


Figure 11. Comparison results with the GridMap method in dynamic obstacle avoidance task.

The parameter $stddev_{th}$ is related to the flatness of the terrain, which needs to be set according to the off-road capability of the unmanned vehicle in the real environment.

Path optimization result: As is introduced in Section 5, the analysis of the traversable corridor and the trajectory optimization of the back-end are performed in this paper based on the path found by the front-end path finder module. Considering the vehicle's controller capability and the safety of the navigation, we set an erosion area of the range of six resolution units to constrain the traversable corridor within the absolute safety area during the analysis of the traversable corridor. As is shown in Fig. 8(a), in order to visualize the results, a semi-transparent cube is used to represent the generated traversable corridor. In generating the traversable corridor, we expanded the size on the Z-axis of the corridor, which is relevant to the height of the vehicle. Figure 8(b) and (c) shows the results of the generated traversable corridor to avoid the pits and steep slopes, showing its ability to navigate safely.

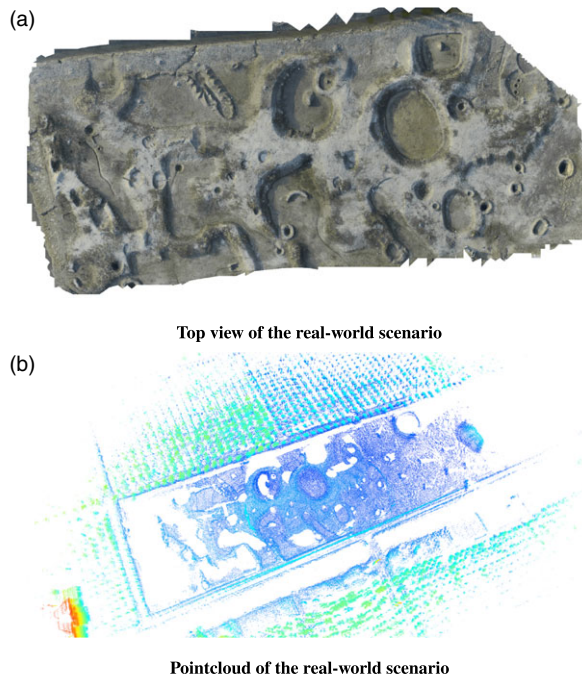
According to Section 5.2, in the process of back-end optimization, constraining the optimized path within the generated passable corridor can not only ensure the safety of the trajectory but also smooth the trajectory and increase the feasibility of it. As is shown in Fig. 9, there are the trajectory generation results for three different sets of starting points, where transparent gray cube is the visualized traversable corridor, the red line is the path searched by the front-end module, and the blue smooth line is the path optimized by the mini-snap method. As is shown in the picture, a smooth trajectory can be generated effectively through the method proposed in this paper, and the trajectory can be constrained within a passable and safe corridor through a single calculation. The comparison of the running time of each part in generating three sets of trajectories is listed in Table II. As is shown in the table, the front-end path finder module and the passable corridor analysis module take approximately the same time, while the back-end trajectory optimization module spends far less time than the first two modules.

6.2. Comparisons and analysis

In this part, our method is compared with a popular 2.5D navigation map representation method proposed by P. Fankhauser using GridMap library [46]. In order to meet the need for real-time computing,

Table IV. Comparison of re-planning time when encounter dynamic obstacles

Item	Methods	TTA time (ms)	Traj time (ms)	Total time (ms)	Start point	Goal point
A	GridMap method	1369	24.15	1393.15	(0,0,0)	(13.799,8.528,0.275)
	Proposed method	305.73	8.58	314.31		
B	GridMap method	1120	31.27	1151.27	(17,12,-0.4)	(33.812,7.228,-0.025)
	Proposed method	266.01	21.66	287.67		
C	GridMap method	1027	/	/	(17,-9,-0.4)	(25.825,-34.094,-1.351)
	Proposed method	245.49	19.61	265.1		

**Figure 12.** The real-world off-road experimental area (point cloud map is built by laser-slam).

only the normal vectors, smoothening, and variance of each cell are considered when performing the TTA in GridMap. Moreover, only the body-centered $20\text{ m} \times 20\text{ m}$ size local map is processed and updated to reduce the time of TTA and improve the real-time performance. However, the re-planning process works only when dynamic obstacles or scene changes are located within 10 m of the vehicle. In the navigation, the sample-based path planning in the 2D map and the dynamic window approach are used for effective control and autonomous navigation.

6.2.1 Mapless navigation

In this part, we compared the proposed method with the GridMap method in a mapless navigation task, in which we do not provide a pre-generated map model. The unmanned vehicle needs to build a map while exploring toward the target. Twenty trials are conducted using each of the two methods, with a target point on a rugged terrain about 40 m from the starting point, and the maximum speed of the

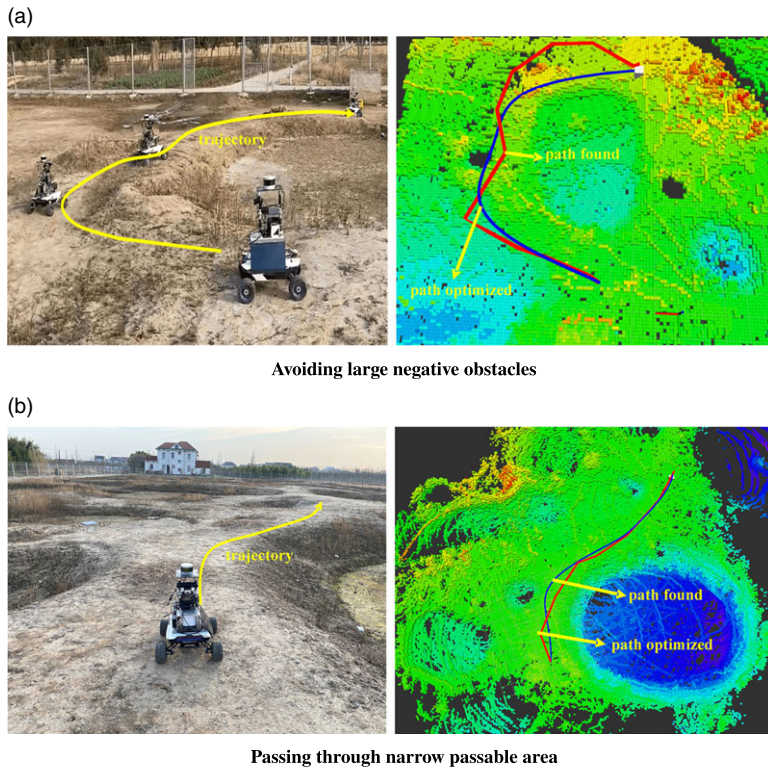


Figure 13. Real navigation in off-road experiment.

vehicle is set as 1m/s. We compared the average navigation time, trajectory length, re-planning times, and success rate.

Table III shows that our proposed method has better performance in most metrics, including the average navigation time, re-planning times, and success rate. It is worth noting that the average trajectory length of our method is longer than that of the GridMap method. This is because the navigation approach of GridMap is more aggressive, which marking unknown regions as free. However, such aggressiveness also leads to a higher probability of failure in the mapless navigation tasks. In contrast, our approach can significantly improve navigation by selecting appropriate local target points and re-planning. As the instance shown in Fig. 10, in most of the failure-prone regions of the GridMap method shown in Fig. 10(b), our approach can effectively re-plan to avoid the risk, as shown in Fig. 10(a).

6.2.2. Dynamic obstacle avoidance

Three sets of experiments are used to compare the dynamic performance of the two methods in specific scenarios including avoiding large negative obstacles (experiment A), passing through narrow passable area (experiment B), and navigating to the bottom of a depression (experiment C). The results are shown in Fig. 11.

As is shown in Table IV, kinodynamic collision-free trajectories are both generated in scenarios with and without dynamic obstacles in experiments A and B. However, our method is faster in the re-planning process and tends to generate a smoother trajectory. In the experiment C, the GridMap method fails to perform the re-planning process when dynamic obstacles are encountered while navigating to the depression. This is because the sudden detection of dynamic objects makes it difficult to avoid when the vehicle encounters negative obstacles due to the limited view of the sensor detection.

6.3. Real-world experiments

We validate our proposed approach in a real-world, unstructured environment using the hardware platform mentioned in Section 3 to show the capability of its real-time navigation. The environment we test is a field site about 500 m in length and 200 m in width, with the maximum undulation of 5 m. And there are pits, soil slopes, ponds, and other obstacles that are difficult for unmanned vehicles to cross, as shown in Fig. 12.

Figure 13 shows some instances of the proposed method. When the target point is on the other side of a huge depression as shown in Fig. 13(a), our method can reasonably plan a path such that the vehicle can avoid the huge negative obstacle along the periphery. In Fig. 13(b), when reaching a target point requires passing through a narrow area, our approach can drive the vehicle to pass in a reasonable posture, avoiding being stuck or detouring.

7. Conclusions and future work

This paper proposes an autonomous navigation method for unmanned vehicles based on terrain analysis in complex unstructured terrain. Particularly, it is specially optimized for complex unstructured terrain with dynamic obstacles and negative obstacles such as pits and steep slopes. This method uses laser SLAM to build the environment map and an octree to represent and store the navigation map. We optimized the TTA method in front-end sampling, the target autonomous exploration process, and back-end trajectory constraint in the 3D octomap to make navigation more efficient. And we showed the improvement of the proposed method in dynamic obstacle avoidance task and mapless navigation task compared to traditional 2D or 2.5D methods.

The navigation method proposed in this paper relies on data from LiDAR sensors, which lack a high-level semantic representation of terrain information. This makes it slightly insufficient in facing complex off-road environment with vegetation obstacles such as tall grass. In the future, we may consider combining visual information to optimize terrain analysis.

Acknowledgements. This paper was supported by the Major Program of National Natural Science Foundation of China under Grant No. 61690214, Shanghai Science and Technology Action Plan under Grant No. 18DZ1204000, 18510745500, 18510750100, 18510730600, Shanghai Aerospace Science and Technology Innovation Fund (SAST) under Grant No. 2019-080, 2019-116 and Shanghai Sailing Program under Grant No. 20YF1417300, and the Natural Science Fund of China (NSFC) under Grant No. 51575186.

Conflicts of interest. The authors declare that they have no conflict of interest.

Notes on contributor(s). Bo Zhou is currently pursuing the Ph.D. degree in mechanical engineering with East China University of Science and Technologies, Shanghai, China. His research interests include Aerial Robotics, motion planning, control and intelligent recognition, artificial intelligence, and computer vision.

Jianjun Yi is a professor with the School of Mechanical and Power Engineering and Computer Science, East China University of Science and Technology, Shanghai, China. His main research interests include intelligent control system, mechatronics, smart sensing and intelligent recognition, artificial intelligence, embedded control system, and so on.

Xinke Zhang is currently pursuing the Ph.D. degree in mechanical engineering with East China University of Science and Technologies, Shanghai, China. His research interests include UGV and UAV slam technology, deep learning model construction, and semantic map construction.

Liwei Chen is currently pursuing the Ph.D. degree in mechanical engineering with East China University of Science and Technologies, Shanghai, China. His research interests include multi-agent system, slam technology, and artificial intelligence application in robotics.

Ding Yang is currently pursuing the Master's degree in School of Mechanical and Power Engineering, East China University of Science and Technology, Shanghai, China. His research interests include embedded control system and computer vision.

Fei Han is currently working at Shanghai Key Laboratory of Aerospace Intelligent Control Technology, Shanghai, China. His research interests include image processing, motion planning, and multi-agent system.

Hanno Zhang is currently working at Shanghai Key Laboratory of Aerospace Intelligent Control Technology, Shanghai, China. She received the Ph.D. degree from China Academy of Science, Beijing, China. Her research interests include image processing and point cloud processing.

References

- [1] J. An and J. Lee, "Robust positioning and navigation of a mobile robot in an urban environment using a motion estimator," *Robotica* **37**(8), 1320–1331 (2019).
- [2] W. Guo, Y. Fukano, K. Noshita and S. Ninomiya, Field-based individual plant phenotyping of herbaceous species by unmanned aerial vehicle," *Ecol. Evol.* **10**(21), 12318–12326 (2020).
- [3] L. Xiao, J. Wang, X. Qiu, Z. Rong and X. Zou, Dynamic-slam: Semantic monocular visual localization and mapping based on deep learning in dynamic environment," *Rob. Auton. Syst.* **117**, 1–16 (2019).
- [4] H. Almqvist, M. Magnusson and A. J. Lilienthal, "Improving point cloud accuracy obtained from a moving platform for consistent pile attack pose estimation," *J. Intell. Rob. Syst.* **75**(1), 101–128 (2014).
- [5] K. Ji, H. Chen, H. Di, J. Gong, G. Xiong, J. Qi and T. Yi, "CPFG-SLAM: A Robust Simultaneous Localization and Mapping Based on Lidar in Off-Road Environment," *2018 IEEE Intelligent Vehicles Symposium (IV)* (2018).
- [6] Y. Yang, D. Tang, D. Wang, W. Song, J. Wang and M. Fu, "Multi-camera visual slam for off-road navigation," *Rob. Auton. Syst.* **128**, 103505 (2020).
- [7] S. Naudet-Collette, K. Melbouci, V. Gay-Bellile, O. Ait-Aider and M. Dhome, "Constrained rGBD-SLAM," *Robotica* **39**(2), 277–290 (2021).
- [8] A. Pfrunder, P. Borges, A. R. Romero, G. Catt and A. Elfes, "Real-Time Autonomous Ground Vehicle Navigation in Heterogeneous Environments Using a 3D Lidar," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2017).
- [9] S. Oishi, Y. Inoue, J. Miura and S. Tanaka, "Seqslam++: View-based robot localization and navigation," *Rob. Auton. Syst.* **112**, 13–21 (2019).
- [10] S. Chhaniyara, C. Brunskill, B. Yeomans, M. Matthews, C. Saaj, S. Ransom and L. Richter, "Terrain trafficability analysis and soil mechanical property identification for planetary rovers: A survey," *J. Terramech.* **49**(2), 115–128 (2012).
- [11] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Eng. Appl. Artif. Intell.* **26**(4), 1373–1385 (2013).
- [12] Z. Zhu, N. Li, R. Sun, H. zhao and D. xu, *Off-road Autonomous Vehicles Traversability Analysis and Trajectory Planning Based on Deep Inverse Reinforcement Learning*. arXiv 2019, arXiv:1909.06953 (2019).
- [13] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots* **34**(3), 189–206 (2013).
- [14] D. C. Guastella and G. Muscato, "Learning-based methods of perception and navigation for ground vehicles in unstructured environments: A review," *Sensors* **21**(1), 73 (2020).
- [15] M. Luong and C. Pham, "Incremental learning for autonomous navigation of mobile robots based on deep reinforcement learning," *J. Intell. Rob. Syst.* **101**(1), 1–11 (2021).
- [16] Proceedings of Machine Learning Research, Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization, vol. 48 (2016).
- [17] M. Wulfmeier, D. Rao, D. Z. Wang, P. Ondruska and I. Posner, "Large-scale cost function learning for path planning using deep inverse reinforcement learning," *Int. J. Rob. Res.* **36**(10), 1073–1087 (2017).
- [18] F. G. Oliveira, A. A. Neto, D. Howard, P. Borges, M. F. Campos and D. G. Macharet, "Three-dimensional mapping with augmented navigation cost through deep learning," *J. Intell. Rob. Syst.* **101**(3), 1–21 (2021).
- [19] M. Quann, L. Ojeda, W. Smith, D. Rizzo, M. Castanier and K. Barton, "Offroad ground robot path energy cost prediction through probabilistic spatial mapping," *J. Field Rob.* **37**(3), 421–439 (2020).
- [20] M. Bellone, G. Reina, L. Caltagirone and M. Wahde, "Learning traversability from point clouds in challenging scenarios," *IEEE Trans. Intell. Transp. Syst.* **19**(1), 296–305 (2018).
- [21] N. Kingry, M. Jung, E. Derse and R. Dai, "Vision-based Terrain Classification and Solar Irradiance Mapping for Solar-Powered Robotics," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018) pp. 5834–5840.
- [22] J. L. Martínez, M. Morán, J. Morales, A. Robles, and M. Sánchez, "Supervised learning of natural-terrain traversability with synthetic 3D laser scans," *Appl. Sci.* **10**(3), 1140 (2020).
- [23] S. Chiodini, L. Torresin, M. Pertile and S. Debei, "Evaluation of 3D CNN semantic mapping for rover navigation," arXiv 2020, arXiv:2006.09761 (2020).
- [24] D. Maturana, P. W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," *In: Field and Service Robotics* (Springer, Cham, 2018) pp. 335–350.
- [25] N. Pérez-Higueras, A. Jardón, N. Rodríguez and C. Balaguer, "3D exploration and navigation with optimal-RRT planners for ground robots in indoor incidents," *Sensors* **20**(1), 220 (2019).
- [26] F. Gao, W. Wu, W. Gao and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *J. Field Rob.* **36**(4), 710–733 (2018).
- [27] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart and J. Nieto, "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2017).
- [28] H. Wang and Y. Liu, "A low-cost autonomous navigation system for a quadrotor in complex outdoor environments," *Int. J. Adv. Rob. Syst.* **17**(1), 172988142090515 (2020).
- [29] M. Likhachev, G. J. Gordon and S. Thrun, "ARA*: Anytime A* with Provable Bounds on Sub-Optimality," *In: Advances in Neural Information Processing Systems* (2003) pp. 767–774.
- [30] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," (1998).
- [31] Z. Ajanovic, B. Lacevic, B. Shyrokau, M. Stolz and M. Horn, "Search-based Optimal Motion Planning for Automated Driving," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018).

- [32] H. Huang, A. V. Savkin and W. Ni, “Energy-efficient 3D navigation of a solar-powered uav for secure communication in the presence of eavesdroppers and no-fly zones,” *Energies* **13**(6), 1445 (2020).
- [33] J. Kuffner and S. LaValle, “RRT-connect: An Efficient Approach to Single-Query Path Planning,” *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2 (2000) pp. 995–1001.
- [34] L. Chen, Y. Shan, W. Tian, B. Li and D. Cao, “A fast and efficient double-tree RRT-like sampling-based planner applying on mobile robotic systems,” *IEEE/ASME Trans. Mechatron.* **23**(6), 2568–2578 (2018).
- [35] M. Werling, J. Ziegler, S. Kammel and S. Thrun, “Optimal Trajectory Generation for Dynamic Street Scenarios in a Frenét Frame,” *2010 IEEE International Conference on Robotics and Automation* (2010).
- [36] W. Ding, W. Gao, K. Wang and S. Shen, “An efficient B-spline-based kinodynamic replanning framework for quadrotors,” *IEEE Trans. Rob.* **35**(6), 1287–1306 (2019).
- [37] W. Ding, L. Zhang, J. Chen and S. Shen, “Safe trajectory generation for complex urban environments using spatio-temporal semantic corridor,” *IEEE Rob. Autom. Lett.* **4**(3), 2997–3004 (2019).
- [38] N. Ratliff, M. Zucker, J. A. Bagnell and S. Srinivasa, “Chomp: Gradient Optimization Techniques for Efficient Motion Planning,” *2009 IEEE International Conference on Robotics and Automation* (2009) pp. 489–494.
- [39] D. Mellinger and V. Kumar, “Minimum Snap Trajectory Generation and Control for Quadrotors,” *2011 IEEE International Conference on Robotics and Automation* (2011) pp. 2520–2525.
- [40] J. Zhang and S. Singh, “Low-drift and real-time lidar odometry and mapping,” *Auton. Robots* **41**(2), 401–416 (2016).
- [41] T. Shan and B. Englot, “Lego-Loam: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain,” *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018).
- [42] G. Kim and A. Kim, “Scan Context: Egocentric Spatial Descriptor for Place Recognition within 3D Point Cloud Map,” *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018).
- [43] F. Gao, W. Wu, Y. Lin and S. Shen, “Online Safe Trajectory Generation for Quadrotors Using Fast Marching Method and Bernstein Basis Polynomial,” *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2018) pp. 344–351.
- [44] J. Chen, T. Liu and S. Shen, “Online Generation of Collision-Free Trajectories for Quadrotor Flight in Unknown Cluttered Environments,” *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016) pp. 1476–1483.
- [45] M. Quigley, K. Conley, B. Gerkey and J. Faust, “ROS: An Open-Source Robot Operating System,” *ICRA Workshop on Open Source Software*, vol. **3** (2009) p. 5.
- [46] P. Fankhauser and M. Hutter, “A universal grid map library: Implementation and use case for rough terrain navigation,” *Stud. Comput. Intell.* **1**(5), 99–120 (2016).