A Reliable Traversability Learning Method Based on Human-Demonstrated Risk Cost Mapping for Mobile Robots over Uneven Terrain

Bo Zhang[a,b,c], Guobin Li[a,b], Jiale Zhang[a,b], Xiaoshan Bai[*,a,b]

[a] *The College of Mechatronics and Control Engineering, Shenzhen University, Shenzhen, 518060, China*
[b] *The Shenzhen City Joint Laboratory of Autonomous Unmanned Systems and Intelligent Manipulation, Shenzhen, 518060, China*
[c] *Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen, 518107, China*

Abstract

The paper proposed a traversability learning method based on the human demonstration for generating risk cost maps. These maps aid mobile robots in identifying safe areas for reliable autonomous navigation over uneven terrain. Firstly, a maximum causal entropy-based inverse reinforcement learning method is employed to generate a reward function by considering human-demonstrated trajectories, robot poses, and feature vectors extracted from elevation data. This reward function is intended to accurately capture the behavioral preferences identified in human-demonstrated trajectories, specifically focusing on low-risk areas of the environment. Secondly, the reward function is combined with terrain feature data to generate a cost map and least-cost trajectory. Utilizing a wheeled mobile robot traversing uneven terrain, this paper verifies the adaptability enhancement of the proposed method for autonomous navigation over outdoor uneven terrain. The experimental results show an increase of 4%-10% in the success rate, a decrease of 13.6%-32.1% in the cumulative slope and gradient, and a decrease of 20.8%-27.4% in the Hausdorff distance of the robot's trajectories compared with traditional inverse reinforcement learning-based navigation methods.

Keywords: autonomous navigation, inverse reinforcement learning, feature mapping, uneven terrain

1. INTRODUCTION

Autonomous mobile robots are increasingly being utilized in outdoor scenarios to assist or replace humans in various tasks, such as package delivery, Mars exploration, and mine search [1, 2]. To operate in such environments, mobile robots have to deal with intricate and uneven terrain, including various obstacles difficulty to identify, and uneven and bumpy ground.[3]. However, ensuring a robot's autonomous navigation reliability in such complex environments remains challenging. To overcome such challenges, it is necessary to develop the robot's ability to autonomously identify safe and feasible areas in challenging environments and to safely plan a reliable path to the target point in uneven terrain [4].

Reinforcement learning (RL) is a promising approach to improve the adaptability of mobile robots in complex environments[5–8]. In [5], the authors review the learning-based methods, such as RL, employed to solve the problem of environmental perception and interpretation with the ultimate goal of autonomous navigation of ground vehicles in unstructured environments. By utilising elevation maps of the environment and the robot pose and goal as input, a fully trained deep reinforcement learning (DRL) network is used to generate an environment attention mask that identifies the reduced stability of regions and evaluates the risk level of the terrain in a given area [6]. Q-Learning, combined with the improved A* algorithm, is used for online planning of a safe path on uneven terrain [7]. In [8], a deep reinforcement learning network is used to precisely control the movement of a robot over a rough terrain by using a depth image as an input. It is important to acknowledge that the efficiency of these RL and planning algorithms is significantly affected by the reward function. Numerous existing traditional navigation frameworks rely on the manual setting of environmental reward functions based on personal experience, without taking into account the optimality of these reward values. In challenging and uncertain terrains, such as hilly wooded landscapes, the diverse and extensive presence of surface features will result in the necessity to continuously adjust the reward function settings, rather than completing it in a single attempt. Indeed, in extremely challenging situations, the process of determining an appropriate reward function will encounter struggles, potentially leading to a deteriorated performance for autonomous navigation [9].

Human perceptual abilities and behavior exhibit good adaptability to complex environments. To learn this human environmental adaptability, the

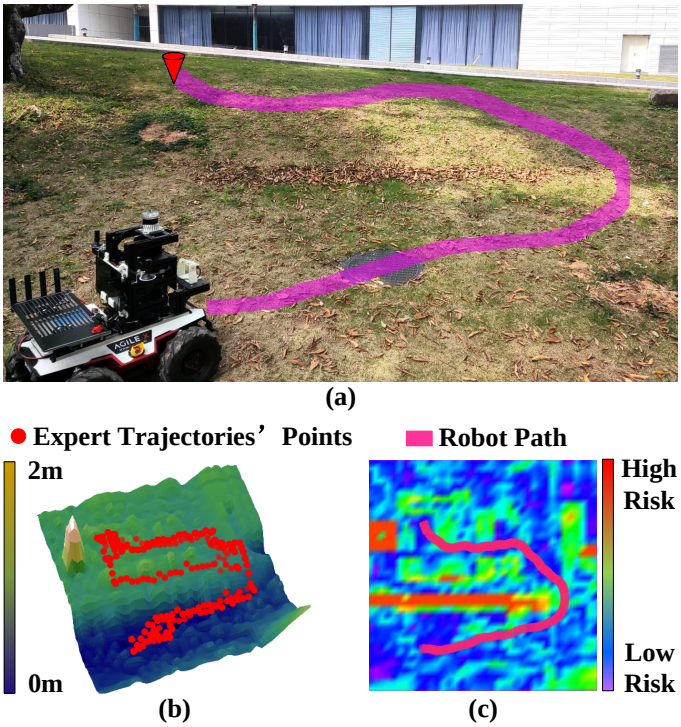

Figure 1: The training process of reward function . Proposed method recovers the cost function that characterizes the environmental risk based on the preference characteristics of the human demonstration for different terrain features and then plans a reliable path to reach the target. Scenario A is shown as figure (a), which includes a slope with a steep front slope and a gentler side slope. Figure (b) shows the DEM corresponding to Scenario A and the 20 human-demonstrated trajectories. These trajectories imply the preference features for low-risk areas of the environment. The environmental cost function and the planned trajectories is shown as figure (c), intercepted by rviz.

human-robot collaborative technology [10] combined with a learning-from-demonstration (LFD) approach is considered to allow robots to learn from humans to traverse uneven terrain. To improve the adaptability of driverless vehicles in urban environments, LFD has been used to train a wide range of reward functions [11] or to learn driving strategies [12]. In [13], dynamic path planning efficiency in social environments is improved by combining LFD with specific environmental feature extraction strategies. For autonomous mobile robots operating in uneven outdoor environments, LFD facilitates the rapid adaptation of robots to new and complicated environments, as well as the acquisition of specific behaviours, such as following road edges

[14]. However, these studies do not account for variations in terrain height or potential risky areas, nor do they involve extracting and mapping terrain features. If the number of trajectories demonstrated by human is insufficient, the learning results of these methods are often suboptimal due to infrequent access to significant features of uneven terrain.

In this paper, a traversability learning method based on human demonstration is proposed for developing risk cost maps to assist mobile robots in identifying safe areas. Initially, the proposed method utilizes the digital elevation model (DEM) to extract the geometric features of the environment. Subsequently, inverse reinforcement learning (IRL) is utilized to acquire the reward function by learning from human demonstrations, simultaneously constructing a risk cost map that depicts the risk information in different areas. Ultimately, considering the environmental risk information, trajectory planning is performed to enable the robot to avoid high-risk areas for reliable autonomous navigation. The advantage of the proposed method lies in its ability to generate a risk cost map of the uneven terrain through LFD. This makes the proposed method particularly suitable for autonomous navigation tasks in challenging terrains where directly specifying environmental reward functions is difficult. The main contributions of our research include the following.

- To address the problem of inadequate and poorly generalizable learning results resulting from a restricted amount of human demonstrations, the terrain is mapped as states using geometric features.
- To tackle the difficulty of determining the reward function in complicated terrain, a maximum causal entropy (MCE) IRL is employed to learn the reward function. This function captures the human preference characteristics of low-risk areas of the environment.
- To improve the reliability of path planning, the methodology for generating the cost map and cost function considers both the robot kinematics and risk information.

2. RELATED WORK

This section provides a review of existing research in the following three areas: learning from demonstration, extracting terrain features and navigation in uneven terrain.

2.1. Learning from demonstration

LFD is a methodology allowing robots to learn knowledge by analyzing data obtained through human demonstrations. This data involves the human's assessment of possible risks and their preferences about environmental features. The learning results are usually manifested in the reward function. The LFD methodology eliminates the requirement for engineers to manually specify reward functions, which is a necessary step in traditional methods. The reward functions acquired by LFD exhibit a higher degree of alignment with the optimal criteria defined by human experts [14]. LFD research in autonomous navigation for mobile robots is inadequate in spite of various LFD-based approaches. These methods involve acquiring the reward function through imitation learning, as described in [15]. The demonstrated data in the method come from either trajectories categorized by humans on satellite maps [16] or trajectories gathered through remote operator control of mobile robots [17]. Furthermore, the integration of IRL with various path planning algorithms, such as Rapidly Exploring Random Trees, has been employed to facilitate the acquisition of navigation behavior for robots [18]. Regarding unstructured environments, this research in [19] compares several IRL approaches to determine the most desirable trajectory. By learning the numerous demonstrations of human driving behavior, deep neural networks (DNNs) have motivated the utilization of deep IRL with convolutional neural networks [20] and highly parallel model predictive control [21] to obtain reward functions. However, these IRL methods using DNNs require considerable demonstration data to determine reward functions. The utilization of an approach that develops the reward function through a limited quantity of demonstration data exhibits significant potential. Recent research has demonstrated the effectiveness of maximum entropy IRL in mitigating the matching ambiguity, illustrated in [22]. An MCE framework expands on the maximum entropy framework in statistical modeling by incorporating Markov decision processes (MDPs) with stochastic transitions [23]. This research utilizes the MCE IRL technique for obtaining the reward function from human demonstrations.

2.2. Extracting terrain features

Accurately identifying terrain features is critical to improve learning efficiency and generalizability. In the initial stages of the investigation, scholars relied on topographic mapping methods [24] and commonly utilized LiDAR or cameras to reconstruct the three-dimensional environment [25]. These

techniques facilitated the acquisition of point clouds or image data, which were subsequently employed for terrain classification [26]. The utilization of DNNs has gained prominence in the field of terrain categorization, particularly in the incorporation of visual input through neural networks [27]. A method for semantic terrain segmentation was presented in a recent work by [28] that utilized both RGB pictures and 3D point clouds. Furthermore, several methods utilized data on robot-ground interaction in real-time for DNNs training, as mentioned by [29]. Nevertheless, most of the visual features extracted by DNNs from robot-environment interaction data can considerably distinguish between environments. Moreover, the enormous amount of 3D point cloud data means that extracting environmental features directly from the data is insufficient to meet real-time navigation requirements. The critical role of DEM for efficient outdoor navigation tasks by robots has been highlighted in recent studies [5]. In our paper, the environmental DEM obtained from the 3D point cloud data [30] is merged with the developed feature mapping approach, presented in Section 4.1, to enhance the efficiency of environmental feature extraction. To overcome the challenges of DEM in modeling overlapping structures, the proposed algorithm filters out the point clouds that exceed the robot's current maximum height, and maintains the point clouds that impact the robot's navigation process. This excludes the necessity for investigating intricate overlapping structures. By employing DEM, the proposed method is now limited to solely taking into account the elevation data of the terrain. The trade-off for this approach is the inevitable restriction in the recognizability of the ground materials.

2.3. Autonomous Navigation on uneven terrain

The safety of path planning algorithms is a crucial consideration in the framework of autonomous navigation for mobile robots performing on uneven terrain. Early research on autonomous navigation in uneven terrain involved robot-terrain interactions [31], occupancy maps [32], continuous obstacle spaces [33], and uneven terrain fields [34]. High-dimensional data can now be handled in the motion planner thanks to the advancement of DRL. For example, in [8], the sensor data onboard was applied to train DNNs to directly output the robot motion control command. In other approaches, perceptual inputs such as robot poses and images introduced additional reward constructs and self-supervised modules are applied to train the neural network [35], or a navigation method that learns from experience is used to allow the robot to pass through areas that were originally judged to be im-

passable [36]. However, most of these DRL methods are based on artificially awarded environmental reward functions. Therefore, in the environment of a robot traversing challenging and uneven terrain with various features, it becomes necessary to adjust the environmental reward function in order to ensure the continuing effectiveness of the algorithm. In the proposed method, the environmental reward function is acquired through IRL using human tutorials. In order to ensure that the learned autonomous navigation can be applied to a variety of difficult terrain scenarios, this method is combined with a path planning algorithm.

3. NOTATIONS AND PROBLEM FORMULATION

3.1. Notations

Table 1 lists the notations and their meanings that are used in our formulation. The robot's location is set as the origin of the world coordinate system at program startup, and the position of the grid in the map relative to the world coordinate origin is represented by the coordinates (i, j). The symbol t denotes the temporal information of the variable or matrix. The DEM $\mathbf{E^t}$ and the cost map $\mathbf{C^t}$ are both represented as two-dimensional matrices in this paper. The values in the matrices represent elevation values or risk cost values at corresponding positions. The size of all matrix data structures in our method are 200×200, representing a realistic environment of $20 \times 20m$ with a resolution of $0.1m$. Additionally, the abbreviations used in this paper and their corresponding full names are listed in Table 2 below.

3.2. Problem Formulation

In this paper, to address the challenge in traditional autonomous navigation frameworks of manually specifying environment cost maps in complex and uneven terrains, an LFD-based method for constructing risk cost maps is adopted, which can be divided into two stages: learning the reward function from human-demonstrated trajectories and calculating the cost map based on this reward function for least-cost trajectory planning. Additionally, a basic assumption is that the policy implied in the human-demonstrated trajectory ζ_E is the optimal policy π^E. To exploit this potentially optimal policy function, an IRL module is trained. This module requires in human-demonstrated trajectories and information about the current environment as

Table 1: Symbols and their meanings used in the formulas

Symbol	Definitions
$\mathbf{E^t}$	DEM data acquired at time t
γ	Slope characteristic quantity of a cell
ω	Uneven characteristic quantity of a cell
s_t	Robot's state at time t
a_t	Robot's action at time t
π	Policy function
ζ	Trajectory containing state-action pairs
ϕ	Feature vectors mapped by trajectory or state
R	Reward function learned from MCE IRL
θ	Reward function parameters
$\mathbf{C^t}$	Cost map for path planning at time t

Table 2: Abbreviations used in this paper

RL	Reinforcement learning
DRL	Deep reinforcement learning
IRL	Inverse reinforcement learning
MCE	Maximum causal entropy
DNNs	Deep neural networks
LFD	Learning from demonstration
DEM	Digital elevation model
MDPs	Markov decision processes
MPL	motion primitive library
IOC	Inverse optimal control
TL	Trajectory length
CSR	Cumulative slope and roughness
SR	Success rate
PT	Planning time
MHD	Modified hausdorff distance

input, and determines a reward function for the current environment:

$$R_\theta(s_t) = \theta \cdot \phi(s_t), \tag{1}$$

where the subscript θ indicates the dependence of the reward function R on the parameter θ, $\phi(s_t)$ is the feature vector corresponding to state s_t, and s_t contains topographic information about the current environment.

Next, a cost map $\mathbf{C^t}$ is constructed based on the reward function recovered from the demonstrated trajectories. The value of each cell index (i, j) in the cost map $\mathbf{C^t}(i, j)$ can directly characterize the risk level of the corresponding terrain. The cost map $\mathbf{C^t}$ guides the navigation component to plan a least-cost trajectory to reach the goal point (see Section 4.3). The least-cost trajectory satisfies the following criteria:

$$\min \sum_{(i,j)\in\zeta} \mathbf{C^t}(i, j), \tag{2}$$

where (i, j) represents the terrain cells traversed by the trajectory ζ.

3.3. Inverse Reinforcement Learning

IRL is based on the MDPs, which comprise four components (S, A, T, R): S is the set of states, A is the set of actions, T is the state-action transition model, and R is the reward function. In MDPs, the interaction between the robot and the environment involves the robot's observation of the state s of the environment, selecting an action a based on the current policy π, executing the action, receiving the current reward r as feedback, and using this reward r to guide the improvement of the policy π so that the robot can achieve higher rewards. It is worth noting that in MDPs of IRL, the reward function R is learned from human demonstrations D (See Fig.2). In this paper, IRL is employed to learn the environmental reward function R from human demonstration trajectories $D = \{\zeta_i\}_{i=1}^{n}$ on uneven outdoor terrain, where each trajectory ζ encompasses a sequence of states and actions. By integrating methods for constructing cost maps and planning paths, the robot is guided to perform tasks designed to imitate human behaviors.

4. LFD BASED TRAVERSABILITY LEARNING METHOD

This section introduces the three main components of the proposed method: extracting and mapping environmental features using terrain geometry, building reward functions from human demonstrations using MCE IRL, and generating a least-cost trajectory using cost maps (refer to Fig.3).

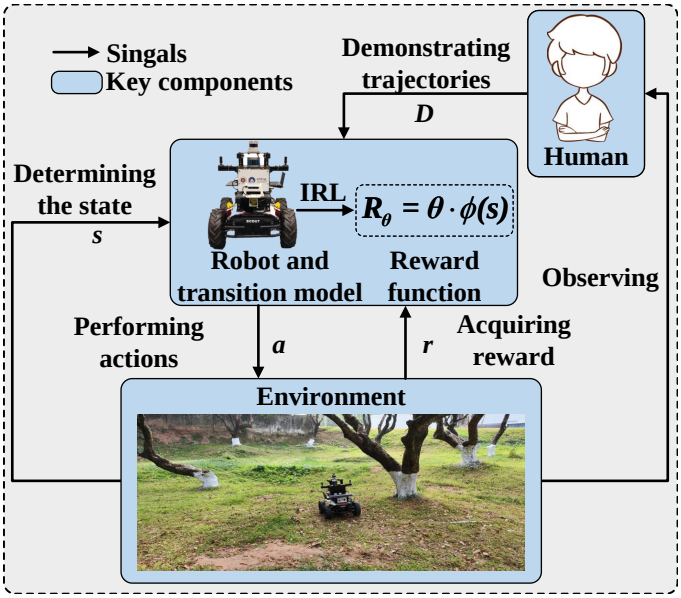

Figure 2: MDPs in IRL.

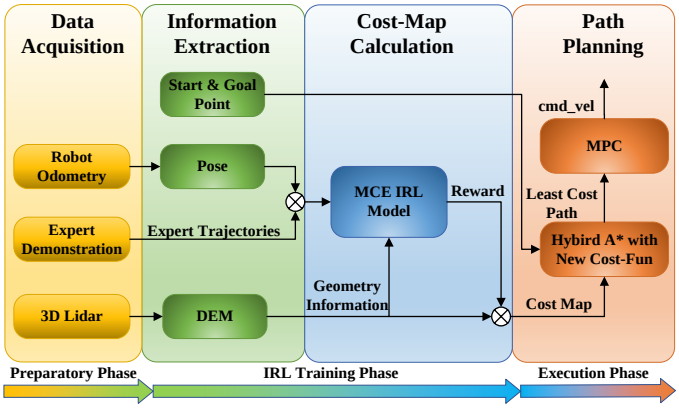

Figure 3: Proposed algorithmic architecture. The DEM of the current environment and the robot's pose obtained from sensors are used along with the human-demonstrated trajectories as input to the MCE IRL module. The output is a reward function characterizing the trajectory preference for environmental features. This reward function is combined with the DEM to calculate a cost map, which serves as the input to the Hybrid A* algorithm for trajectory planning. The actual execution of the planned trajectory is performed using model predictive control (MPC).

4.1. Environmental geometric feature extraction and mapping

To improve the computational efficiency of IRL optimization and obtain a generalizable reward function with topographic information, DEM data are used to compute the geometric features of each cell for state mapping. Unlike the traditional method, which maps only feature vectors to coordinates (x, y), slope and roughness are computed to characterize the cell state.

The slope γ of a cell is calculated by plane fitting. Let U_i be a square region of size $N \times N$ sampled from the DEM, which contains N^2 points, denoted as $(x_{i,j}, y_{i,j}, z_{i,j}), i, j \in U_i$. Plane fitting is applied to N^2 points within region U_i, resulting in the 3D plane $Ax_{i,j} + By_{i,j} + C = z_{i,j}$, where A, B, C are the coefficients of the 3D plane. The center point γ of the region U_i is then calculated using the following equation:

$$\gamma = \frac{\pi}{2} - \sin^{-1} \frac{1}{\sqrt{A^2 + B^2 + 1}}. \tag{3}$$

The abrupt changes in the ground elevation values are denoted by ω and calculated as follows:

$$\omega = d^{-1} \sqrt{\frac{1}{N^2} \sum_{i,j \in U_i} (z_{i,j} - \bar{z})^2}, \tag{4}$$

where d is the wheel diameter of the mobile robot, and $\bar{z}$ is the average height of N^2 cells within U_i.

Each cell state is represented as $s = [\gamma, \omega]^T$ instead of the coordinates (x, y). To enable the states to participate in the computation, each state s must be mapped into a feature vector $\phi(s)$. This is achieved by dividing each geometric feature into k discrete intervals ranging from $0, \frac{1}{k}$ to $\frac{k-1}{k}$, resulting in a total of k^2 possible cell state combinations. Each state corresponds to an independent unit vector $\phi(s) \in \mathbb{R}^{k^2}$. The proposed feature extraction method decreases the number of computations and demonstration trajectories required for the subsequent IRL optimization process. In addition, this feature mapping method can be adapted to different environments by adjusting the k value, which is more efficient for computing.

4.2. Learning reward functions from demonstrations

The function of the IRL module is to explain human behaviors and build the underlying reward function. The feature extraction and mapping approach mentioned in the previous Section 4.1 greatly reduces the number of

required demonstration trajectories and maps uneven terrain into an MDP with stochastic transition dynamics. Therefore, an IRL method based on the MCE principle [23] is chosen, which has advantages over other IRL methods because the ambiguity of the decision to find the reward function is eliminated by the maximum entropy principle. MCE IRL is suitable for MDPs with few trajectories and random state transitions.

As input to the MCE IRL component is a set of human demonstrations $D = \{\zeta_i\}_{i=1}^{n}$ comprising n human-demonstrated trajectories. Each trajectory $\zeta = [(s_1, a_1), (s_2, a_2), ..., (s_{T-1}, a_{T-1}), (s_T, a_T)]$ contains sequences of action variables $\mathbf{a}_{1:T}$ and sequences of state variables $\mathbf{s}_{1:T}$. In the causal entropy method, the action a_t selected by the robot at time t depends not only on the current state s_t but also on the past state $\mathbf{s}_{1:t-1}$ and the past action $\mathbf{a}_{1:t-1}$. Therefore, the probability of performing action a_t can be expressed as $P(a_t|\mathbf{s}_{1:t}, \mathbf{a}_{1:t-1})$. Similarly, the state s_t is related to the past state $\mathbf{s}_{1:t-1}$ and the past action $\mathbf{a}_{1:t-1}$. The probability of reaching the state s_t can be expressed as $P(s_t|\mathbf{s}_{1:t-1}, \mathbf{a}_{1:t-1})$. On the basis of this, the causal entropy is defined as follows.

$$H\left(\mathbf{a}_{1:T} \| \mathbf{s}_{1:T}\right) \triangleq \mathbb{E}_{P(\mathbf{a}_{1:T}, \mathbf{s}_{1:T})}\left[-\log P\left(\mathbf{a}_{1:T} \| \mathbf{s}_{1:T}\right)\right], \tag{5}$$

where $\mathbb{E}$ is the expectation and:

$$\begin{cases} P\left(\mathbf{a}_{1:T}, \mathbf{s}_{1:T}\right) = & \prod_{t=1}^{T} P\left(s_t \mid \mathbf{s}_{1:t-1}, \mathbf{a}_{1:t-1}\right) \times \\ & P\left(\mathbf{a}_{1:T} \| \mathbf{s}_{1:T}\right) \\ P\left(\mathbf{a}_{1:T} \| \mathbf{s}_{1:T}\right) \triangleq & \prod_{t=1}^{T} P\left(a_t \mid \mathbf{a}_{1:t-1}, \mathbf{s}_{1:t}\right) \end{cases}. \tag{6}$$

Remark 1: The entropy H is utilized to evaluate the distribution P derived by optimization. If there are multiple distributions P that satisfy the requirements, the one with the highest entropy H should be selected, which means determining a solution that conforms with the information provided to the solver with minimal deviation [23].

To allow the states into the computation, the feature mapping method described in Section 4.1 is used to transfer each state $\mathbf{s}$ to a feature vector $\phi(s)$. The feature vector representing the trajectory ζ is denoted as $\phi(\zeta)$, and the relationship between two feature vectors is shown as follows:

$$\phi(\zeta) = \sum_{(s_i, a_i) \in \zeta} \phi(s_i). \tag{7}$$

MCE IRL is based on feature-expectation matching, which requires the expected frequency of visiting features in the human demonstrations to be equal to the expected frequency of features visited by the robot based on the recovered reward function. This is expressed as follows:

$$\mathbb{E}_{\pi^L}[\phi(\zeta)] = \mathbb{E}_{\pi^E}[\phi(\zeta)], \tag{8}$$

where π^L is the strategy held by the robot, which is obtained from the restructured reward, and π^E represents the potential strategy followed by the human demonstrations and encodes the notion of environmental rewards and optimality.

Combining (5) and (8), an optimization problem can be formulated as follows:

$$\begin{aligned} &\underset{\{P(a_t|\mathbf{s}_{1:t},\mathbf{a}_{1:t-1})\}}{\operatorname{argmax}} \quad H\left(\mathbf{a}_{1:T}\|\mathbf{s}_{1:T}\right), \\ &s.t. \begin{cases} \mathbb{E}_{\pi^L}[\phi(\zeta)] = \mathbb{E}_{\pi^E}[\phi(\zeta)] \\ \sum_{\forall \mathbf{s}_{1:t},\mathbf{a}_{1:t-1}} P(a_t|\mathbf{s}_{1:t},\mathbf{a}_{1:t-1}) = 1 \end{cases} . \end{aligned} \tag{9}$$

By solving this optimization problem, an action probability distribution can be obtained. This distribution, denoted $P(a_t|\mathbf{s}_{1:t},\mathbf{a}_{1:t-1})$, maximizes entropy and ensures that the expected feature visitation frequency of the robot's trajectory, $\mathbb{E}_{\pi^L}[\phi(\zeta)]$, matches that of the human-demonstrated trajectory $\mathbb{E}_{\pi^E}[\phi(\zeta)]$.

To learn the reward function, it must be parameterized. Combining (1) and (7), the reward of trajectory ζ can be defined as a linear combination of the feature vectors that represent the states:

$$R_\theta(\zeta) = \theta^T\phi(\zeta) = \sum_{i=0}^{|\zeta|}\theta_i\phi(s_i). \tag{10}$$

where $|\zeta|$ is the length of the trajectory ζ. Thus, given the trajectory feature vector $\phi(\zeta)$, the objective of solving (9) is to optimize weight parameter vector θ to optimize reward $R_\theta(\zeta)$.

To solve parameter θ from (9), the Lagrange multipliers method and the gradient ascent method are used. The Lagrange multiplier is parameter θ of the reward function, and the gradient formula of parameter θ is obtained as:

$$\nabla_\theta L(P,\theta) = \mathbb{E}_{\pi^E}[\phi(\zeta)] - \sum_{s_t\in S} D_{s_t}\phi(s_t), \tag{11}$$

where S is the set containing all states s in the trajectory ζ, and D_{s_t} is the expected state visitation frequency and can be calculated iteratively as follows:

$$D_{s_{t+1}} = \sum_{s_t \in S} \sum_{a_t \in A} D_{s_t} \pi \left(a_t \mid s_t, \theta\right) P \left(s_{t+1} \mid s_t, a_t\right). \tag{12}$$

where A is the set containing all actions a in the trajectory ζ, and the policy function $\pi(a_t|s_t, \theta)$ can be calculated by the current optimized parameter θ combined with the known environmental state transfer probability $P(s_{t+1}|s_t, a_t)$:

$$\pi(a_t|s_t, \theta) = \exp(\log Z_{s_t,a_t} - \log Z_{s_t}), \tag{13}$$

where:

$$\begin{cases} \log Z_{s_t,a_t} = \theta^T \phi(s_t) + \lambda \sum\limits_{s_{t+1} \in S} P(s_{t+1}|s_t, a_t) \log Z_{s_{t+1}} \\ \log Z_{s_t} = \underset{a_t}{\text{softmax}} \log Z_{s_t,a_t} \end{cases}, \tag{14}$$

where $\lambda \in (0, 1)$ is the discount factor, and $\underset{x}{\text{softmax}} f(x) \triangleq \log \sum\limits_{x} f(x)$.

Remark 2: The parameter θ is determined through a rolling optimization process by equations (11)-(14). In (14), θ represents the result of the previous optimization step, which is not optimal. The optimization process is accomplished once the $\nabla_\theta L(P, \theta)$ in (11) reaches a sufficiently minimal number. By substituting the optimal value of θ into (10), the ultimate reward function R_θ is obtained.

4.3. Least-cost trajectory generation

Before utilizing the planning algorithm to generate a least-cost navigation trajectory, it is necessary to construct a cost map containing information about the risk of the current uneven terrain. By combining the reward function recovered from the IRL with the terrain feature vector, the corresponding cost value is calculated as follows:

$$\mathbf{C^t}(i, j) = (\gamma(i, j) + \omega(i, j)) \times R_\theta^{-1}(s_t). \tag{15}$$

Moreover, to improve the security of robot navigation, the following operation is performed on the cost map:

$$\mathbf{C^t}(i, j) = \infty \quad \text{if} \quad \mathbf{C^t}(i, j) > C_{\max}, \tag{16}$$

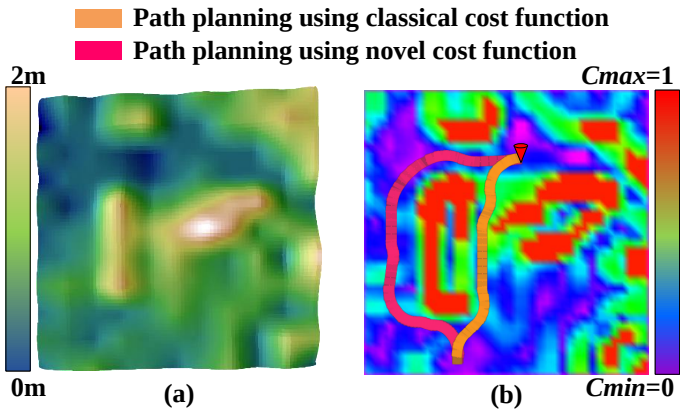

Figure 4: Cost map generation and least-cost path planning. (a) A testing environment with numerous steep slopes and elevation values between 0 and 2 m. (b) The generated cost map in Rviz, where regions closer to red indicate higher risk. The least-cost path is planned using our cost function (pink) and compared to the standard cost function (orange) using the Hybrid A* algorithm. Compared with the orange path, the pink path stays away from high-risk (red) areas and does not pass through medium-risk (green) areas, resulting in a lower CSR. Therefore, the pink path can be considered to have higher reliability.

where C_{max} is determined based on the generation value corresponding to very high-risk terrain and allows the robot to avoid these high-risk areas. In this paper, $C_{max} = 1$ (See Fig.4).

The purpose of the proposed path planning method is to validate the effectiveness of cost maps. Therefore, in this paper, path planning method does not take into account complex situations such as dynamic obstacles. The Hybrid A* algorithm is used to generate the least-cost navigation trajectory. The advantage of the Hybrid A* algorithm over the A* algorithm lies in its ability to generate trajectories that satisfy the robot's dynamic constraints, reducing the burden of trajectory optimization [37]. However, the tipping risk of the wheeled robot driving on uneven terrain is not considered in the cumulative cost function and the heuristic cost function of the classical Hybrid A* algorithm. Moreover, the classical cost function penalizes the safety detours executed by the robot, rendering path planning in uneven terrain unreliable. Therefore, the cost function for the Hybrid A* algorithm is redesigned to match the cost map obtained in Section 4.2.

The Hybrid A* algorithm chooses the node with the lowest total cost value c from the priority queue to appear each time, where the total cost

value corresponding to node n is defined as $c(n) = g_s(n) + h_g(n)$. The cumulative cost g designed according to (15) considers the recovery cost and is defined as:

$$g_s(n) = \sum_{k=1}^{n-1}(w_1 \cdot l_{k,k+1} + w_2 \cdot \mathbf{C^t}(i_{k+1}, j_{k+1})), \tag{17}$$

where the subscript s denotes the accumulation from the starting state, $l_{k,k+1}$ is the length of the path elapsed from time k to time $k+1$, and w_1, w_2 are weighting factors. For the heuristic cost h, we account for not only the kinetic constraints and obstacle constraints but also the total cost on the prediction path. Therefore, the heuristic cost h is defined as:

$$h_g(n) = max(h_{rs}(n), h_{2d}(n)), \tag{18}$$

where $h_{rs}(n)$ denotes the cumulative cost of the Reeds-Shepp curve [38] considering wheeled robot dynamics constraints but not obstacles from the current state s_n to the end state s_g. The cumulative cost of the path refers to the sum of the corresponding generated trajectory values containing nodes in $\mathbf{C^t}$. $h_{2d}(n)$ denotes the cumulative cost of the path from s_n to s_g without considering vehicle dynamics constraints but considering obstacles. This path is given by the two-dimensional A* algorithm [7] using (17) as the cumulative cost function and the Euclidean distance from s_n to s_g as the heuristic function. (18) is applied to ensure that $h_g(n) = h_{2d}(n)$ is satisfied if the robot is far from s_g for emphasizing obstacle avoidance to approach the goal quickly. Inversely, if the robot is closer to s_g, $h_g(n) = h_{rs}(n)$ is satisfied, ensuring that the robot can approach the target point with the appropriate pose. The Hybrid A* algorithm is configured with both standard and proposed cost functions respectively to compare which cost function can result in more reliable trajectory planning. Additionally, two evaluation metrics are set:

- Trajectory Length (TL) - The length of the robot's trajectory (m).
- Cumulative Slope and Roughness (CSR) - The sum of the gradient and unevenness values of the cell passed by the robot along the trajectory divided by the TL, where a lower value indicates a safer trajectory.

Table 3 shows the results of using the Hybrid A* algorithm with the standard and unique cost functions to plan 30 trajectories independently.

The higher CSR observed in the standard cost function can be attributed to its exclusive reliance on distance information, neglecting to account for environmental risk information. As a result, the planning algorithm incurs a penalty if detour decisions are predicated upon security concerns. In contrast, the proposed cost function associated with lower CSR values indicates the trajectories' effectiveness for avoiding medium to high-risk areas and the reliability of the Hybrid A* algorithm. Moreover, the trajectories planned based on the proposed cost function have a higher TL compared with the standard cost function. This is attributed to the detour processing made by the proposed cost function to enhance safety by avoiding certain medium to high-risk areas (see Fig.4(b)).

5. EXPERIMENTAL RESULT

In this section, the implementation of proposed method in real outdoor scenarios is presented. Then, the evaluation metrics are explained and the experimental results are discussed.

5.1. Implementation

The robot used in the experiments is equipped with a four-wheel differential chassis as a mobile carrier, an Oster OS0-128 LiDAR, an SBG-Ellipse-N INS, a dual real-time kinematics (RTK) antenna for receiving GPS satellite signals, a microcomputer (NUC) that houses an Nvidia RTX 2060 GPU and an Intel i7 CPU, and two auxiliary debugging peripherals, a screen and Wi-Fi. The NUC is installed with Ubuntu 18.04 and ROS Melodic (See Fig.5).

All our programs run in the ROS environment. DEM data are obtained using the elevation mapping ROS package [30] with point clouds generated by LiDAR. INS and GPS are used to obtain the vehicle position. The demonstration trajectories are obtained by human remote control operations. The

Table 3: Comparison of experimental results of two cost functions

Metrics	Cost Function	Mean	Best
TL	Standard	**28.64**	**21.93**
	Proposed	37.83	24.51
CSR	Standard	0.39	0.28
	Proposed	**0.21**	**0.17**

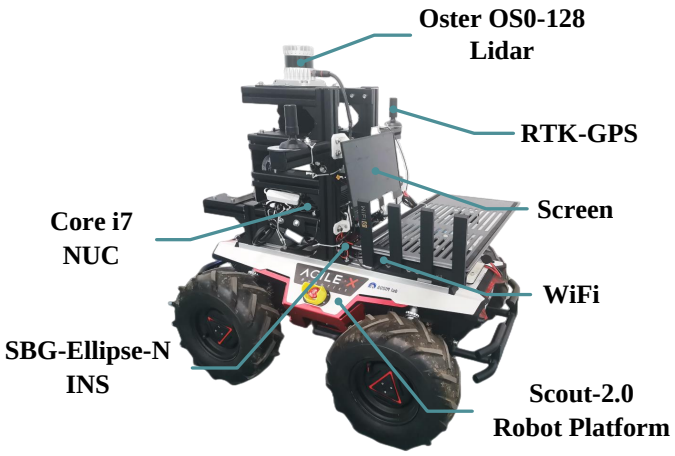

Figure 5: Robot with associated sensors and auxiliary peripherals

code for our method is lightweight enough that all the ROS packages can be run together in real time on the NUC.

5.2. Evaluation Metrics

In addition to the TL and CSR metrics mentioned in Section 4.3, the subsequent metrics are utilized to compare the navigation performance of proposed method with the motion primitive library (MPL) [16] and the inverse optimal control (IOC) [14] in outdoor scenarios. The MPL method is based on imitation learning and uses maximum margin planning to learn costs given desirable trajectory demonstrations. The IOC method is based on maximum entropy IRL, which does not map features to the environment.

- Success Rate (SR) - The number of times the robot successfully reaches the target goal while avoiding obstacles and high-risk areas divided by the total number of attempts. A value closer to 1 indicates a higher practical feasibility of the method.

- Planning Time (PT) - The algorithm's computational time for trajectory planning (ms).

- Modified Hausdorff Distance (MHD) - The Hausdorff distance between the method planning trajectory and the human-demonstrated trajec-

tory, defined as:

$$h(\zeta_L, \zeta_E) = \frac{1}{|\zeta_L|} \sum_{s_i \in \zeta_L} \left(\min_{s_j \in \zeta_E} \hat{d}\left(s_i, s_j\right) \right), \tag{19}$$

where ζ_L is the trajectory to be compared, ζ_E is the human-demonstrated trajectory, and the function $\hat{d}$ indicates the distance between two states along these trajectories. A lower value indicates that the trajectory planned by the method is more similar to the human demonstrations.

The CSR, TL and SR metrics represent our attention to plan trajectories with lower accumulated risk, shorter trajectory lengths, and successful arrival at the destination. In the training scenarios, the MHD metric additionally signifies our desire for the planned trajectories to closely resemble the demonstrations. Furthermore, this paper focuses primarily on the risk characteristics of the environments, while ignoring the additional energy consumption caused by risky regions. Considering that the controller intends to maintain a relatively constant robot speed, the trajectory length can be seen as a reflection of the energy consumption.

5.3. Testing Scenarios

Our autonomous navigation framework is trained and tested in the following three outdoor scenarios and compared with the previously mentioned methods:

- Scenario A - a training scenario consisting of an uneven outdoor terrain environment with elevation values ranging from 0 to $2m$. The robot acquires the reward function R by learning from human demonstrations in scenario A. Twenty human-demonstrated trajectories are acquired by human remote control operation of the robot, which contains a preference for areas with low environmental risk (see Fig.1).

- Scenario B - a test scenario consisting of an uneven outdoor terrain environment with elevation values ranging from 0 to $1.5m$. Scenario B serves as a generalization test scenario, where experiments do not involve IRL process. The reward function R in Scenario B is learned from the human demonstrations in Scenario A (see Fig.8(a)).

- Scenario C - a test scenario consisting of an urban environment with a high curb and small slope, where the curb is well above the ground

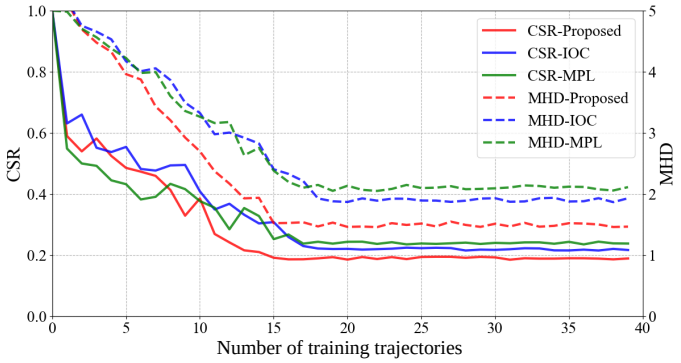

Figure 6: The correlation between the number of demonstration trajectories and the quality of generating cost maps. A smaller value of CSR and MHD indicates a higher quality of the cost map corresponding to the trajectory.

clearance of the robot and must be safely avoided. Similar to Scenario B, Scenario C also serves as a generalization test scenario (see Fig.8(c)).

5.4. Comparison and Analysis

To validate the correlation between different numbers of demonstration trajectories and the quality of the generated cost maps, a comparison is performed between proposed method and other methods in generating the cost maps using various numbers of demonstration trajectories in Scenario A. To mitigate potential impacts from planning algorithm parameters and the robot's intrinsic state, a fixed-parameter A* algorithm is employed to generate a single trajectory. Since the actual optimal cost map is unknown, the validation of the cost map quality was performed indirectly by assessing the performance of the trajectories planned on the map. CSR and MHD metrics are used to evaluate the quality of the generated cost maps.

The experimental results (see Fig. 6) show that if the number of demonstration trajectories is less than 15, increasing the number of demonstration trajectories for all three methods consistently leads to a significant reduction in both the CSR and MHD values of the generated trajectory. This observation indicates an improvement in the quality of the generated cost maps. The effect above occurs due to the increase in the number of demonstration trajectories, resulting in a higher frequency of visits to significant states D_{st} in both the proposed approach and IOC methods. As indicated by (11), the frequency of visiting these significant states directly influences the iterative gradient during the training process, thereby affecting the cost

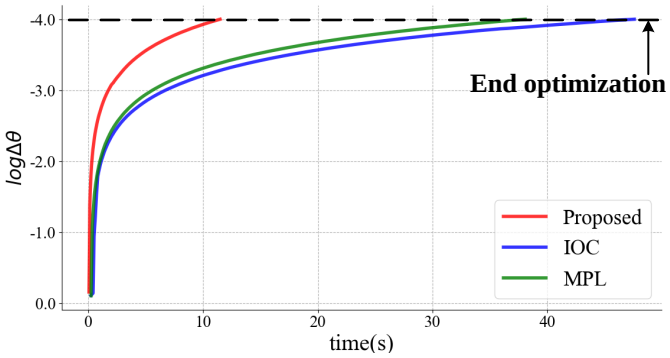

Figure 7: Optimization process of parameter θ. The convergence criterion is defined as the absolute change $\Delta\theta < 0.0001$ ($log\Delta\theta < -4$). This indicates that the parameter θ has reached a stable solution and the optimization is finished.

values corresponding to these states in the generated cost map. In the MPL method, increasing the number of demonstration trajectories further restricts the space of possible cost functions, thereby encouraging the learned cost function to get closer to the underlying optimal cost function. Once the number of demonstration trajectories exceeds twenty, increasing the number of demonstration trajectories has a marginal effect on the quality of the cost maps generated. Therefore, a consistent set of twenty demonstration trajectories is employed in subsequent experiments to train the proposed method, IOC and MPL methods.

Twenty human-demonstrated trajectories (see Fig. 1(b)) are uniformly used by different methods for training, and each method performs 30 experiments, resulting in 30 obtained trajectories. The optimization curve of parameter θ during training is shown in Fig. 7. The optimization of parameter θ using the MPL and IOC methods takes 38.083 and 47.579 seconds, respectively, while the convergence process of parameter θ in the proposed method takes only 11.843 seconds, which is significantly faster than that of the MPL and IOC methods. This is due to the feature mapping method mentioned in Section 4.1, which reduces the computational effort of the optimization process by reducing the number of states to $100(k = 10)$ in the 200×200 DEM. In contrast, the MPL and IOC methods perform feature mapping by cell coordinates, which results in a DEM containing $40,000$ feature vectors and a longer optimization process.

The experimental results are listed in Table 4. The SR statistical results show that all methods have a higher success rate in the training environment,

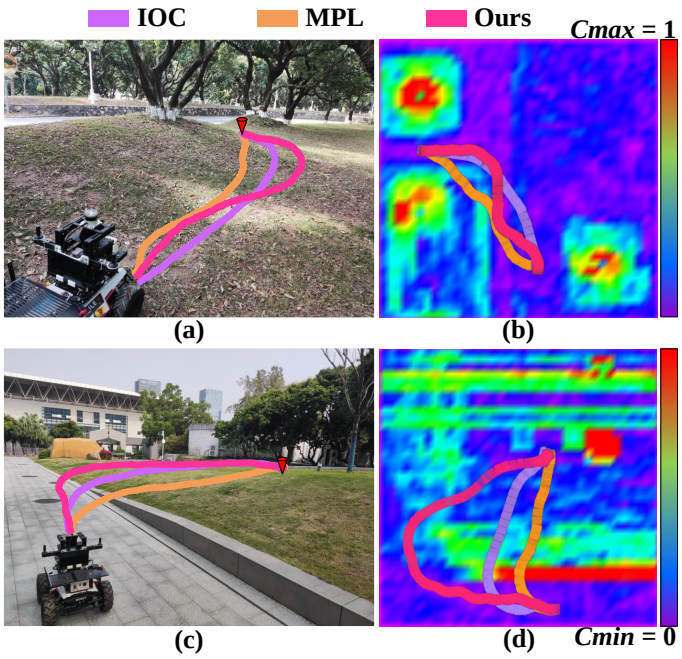

Figure 8: Robot trajectories navigating in different testing scenarios using proposed method (pink), IOC (purple) and MPL (orange). (a) Testing Scenario B. (b) The corresponding cost map generated by proposed method for Scenario B. (c) Testing Scenario C. (d) The corresponding cost map generated by proposed method for Scenario C.

and the proposed method's success rate remains high in the test environment. In contrast, the MPL and IOC methods have decreased SRs. This is because proposed method selects the lowest risk area for navigation through terrain feature mapping. However, since the MPL and IOC methods do not feature maps of the environment, the absence of some environmental features in the new scenario results in a reduction in the SR. Furthermore, MPL, based on imitation learning, has a more severe feature loss in the new environment Scenario C, which is different from the training environment Scenario A, leading to a lower navigation SR.

The TL and CSR values in Table 4 show that proposed method navigates through the least costly state points, resulting in a lower average trajectory risk. Thus, these TL and CSR values validate the reliability of proposed method. However, this also results in an increase in TL compared to other methods. The PT metrics in Table 4 show that the proposed method takes more time in trajectory planning compared with IOC and MPL. The reason

Table 4: Experimental results

Metrics	Method	Scenario A		Scenario B		Scenario C	
		Mean	Best	Mean	Best	Mean	Best
SR	MPL	0.93	-	0.80	-	0.17	-
	IOC	0.90	-	0.87	-	0.57	-
	Ours	0.93	-	**0.90**	-	**0.93**	-
PT	MPL	317.99	298.23	232.19	**185.32**	252.77	207.62
	IOC	**284.23**	**265.19**	**210.73**	189.47	**224.60**	**193.57**
	Ours	393.92	344.17	249.41	198.56	326.89	274.33
TL	MPL	**58.33**	52.62	**25.74**	**21.87**	**28.02**	**23.49**
	IOC	62.25	**51.57**	30.29	25.64	35.33	29.69
	Ours	61.94	59.72	29.14	27.91	43.12	40.51
CSR	MPL	0.26	0.22	0.28	0.23	0.67	0.31
	IOC	0.27	**0.19**	0.22	0.17	0.37	0.29
	Ours	**0.22**	0.20	**0.19**	**0.15**	**0.21**	**0.18**
MHD	MPL	2.15	1.41	-	-	-	-
	IOC	1.97	**1.31**	-	-	-	-
	Ours	**1.56**	1.35	-	-	-	-

is that the proposed method necessitates extra computation of Reeds-Shepp curves when calculating the heuristic function (18). Additionally, the longer trajectories generated by the proposed method result in a larger search space during the planning process, leading to an increase in PT. In this paper, the robot's maximum speed is 2 m/s, and PT values below 500 ms are sufficient for the robot to perform navigation tasks. Therefore, the PT values for the proposed method in Table 4 are considered acceptable. In addition, the MHD values in Table 4 confirm that the trajectories generated by the proposed method are closer to the optimal navigation behavior defined by the human demonstrations, indicating that the proposed method learns human strategy π^E more effectively. The MHD metric is not applicable in Scenarios B and C because the MHD metric is used to evaluate the similarity between trajectories generated by the proposed method and human-demonstrated trajectories, and Scenarios B and C serve as generalization test scenarios, not involving the IRL process. Human demonstrations are only conducted in Scenario A.

In urban environment Scenario C, the curbs above the robot's ground clearance pose a challenge for all methods, and the lower elevation gain in the curb regions is likely to lead to erroneous trajectories. However, proposed method can focus on the low-risk feature regions near the curb, thus achieving a higher SR than other methods. It is observed that the paths from proposed method have low cumulative slope and roughness and can handle challenging curb scenarios (See Fig. 8(d)).

6. CONCLUSIONS

This study introduced a traversability learning method based on human demonstration for generating risk cost maps. Proposed method combines geometry-based terrain feature extraction with MCE IRL to generate a cost map, and employs the hybrid A* algorithm with a proposed cost function to generate the least-cost trajectory. Proposed method is compared with other IRL-based navigation methods on various metrics on a real wheeled robot and uneven outdoor environments to demonstrate its effectiveness.

Despite the promising result, our method still has some limitations that require future investigation. For example, the limited perceptual capability of LiDAR prevents the robot from discriminating between different soil materials. This limitation can be addressed in the future by incorporating visual information from a camera. In addition, our method may not perform well in new environments that differ significantly from the training environment. For instance, in the challenging newly-appeared surfaces stricken by disasters, the generalization of our method might be compromised without sufficient demonstrated data for these new features. This issue will be addressed by adding online demonstration correction and integrating LFD results from multiple scenarios. Finally, it can be anticipated that the combination of large models with big data can significantly enhance the representational capability of IRL methods. In other words, large models will enable IRL to improve algorithm performance in different scenarios by learning from massive human demonstration data. This is an important direction for future improvements to our method.

CRediT authorship contribution statement

Bo Zhang: Conceptualization, Supervision, Formal analysis, Writing - Review & Editing, Funding acquisition. **Guobin Li**: Investigation, Method-

ology, Software, Experiment, Writing - Original Draft. **Xiaoshan Bai**: Validation, Visualization, Resources, Funding acquisition. **Jiale Zhang**: Software, Experiment, Writing - Review & Editing.

Declaration of competing interest

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

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) (No.62173234, 62003217 and 62373255), the Shenzhen Science and Technology Program (No.20220818095816035), and the Shenzhen Natural Science Fund (No.20220809175803001).

References

[1] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, P. J. Stuckey, Integrated task assignment and path planning for capacitated multi-agent pickup and delivery, IEEE Robotics and Automation Letters 6 (3) (2021) 5816–5823.

[2] A. L. Rankin, M. W. Maimone, J. J. Biesiadecki, N. Patel, D. Levine, O. Toupet, Mars curiosity rover mobility trends during the first 7 years, Journal of Field Robotics 38 (5) (2021) 759–800.

[3] D. Shah, S. Levine, Viking: Vision-based kilometer-scale navigation with geographic hints, in: Robotics: Science and Systems (RSS), 2022.

[4] S. Siva, M. B. Wigness, J. G. Rogers, Robot adaptation to unstructured terrains by joint representation and apprenticeship learning, in: Robotics: Science and Systems (RSS)), 2019.

[5] D. C. Guastella, G. Muscato, Learning-based methods of perception and navigation for ground vehicles in unstructured environments: A review, Sensors 21 (1) (2021) 73.

[6] K. M. K. Weerakoon, A. J. Sathyamoorthy, U. Patel, D. Manocha, Terp: Reliable planning in uneven outdoor environments using deep reinforcement learning, in: 2022 International Conference on Robotics and Automation (ICRA), 2022, pp. 9447–9453.

[7] B. Zhang, G. Li, Q. Zheng, X. Bai, Y. Ding, A. Khan, Path planning for wheeled mobile robot in partially known uneven terrain, Sensors 22 (14) (2022) 5217.

[8] K. Zhang, F. Niroui, M. Ficocelli, G. Nejat, Robot navigation of environments with unknown rough terrain using deep reinforcement learning, in: 2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), 2018, pp. 1–7.

[9] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, P. Abbeel, Overcoming exploration in reinforcement learning with demonstrations, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 6292–6299.

[10] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, S. S. Srinivasa, Trust-aware decision making for human-robot collaboration, ACM Transactions on Human-Robot Interaction 9 (2) (2020) 1 – 23.

[11] M. Wulfmeier, D. Rao, D. Z. Wang, P. Ondruska, I. Posner, Large-scale cost function learning for path planning using deep inverse reinforcement learning, The International Journal of Robotics Research 36 (10) (2017) 1073–1087.

[12] C. You, J. Lu, D. Filev, P. Tsiotras, Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning, Robotics and Autonomous Systems 114 (2019) 1–18.

[13] B. Kim, J. Pineau, Socially adaptive path planning in human environments using inverse reinforcement learning, International Journal of Social Robotics 8 (1) (2016) 51–66.

[14] M. B. Wigness, J. G. Rogers, L. E. Navarro-Serment, Robot navigation from human demonstration: Learning control behaviors, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 1150–1157.

[15] B. D. Argall, S. Chernova, M. Veloso, B. Browning, A survey of robot learning from demonstration, Robotics and autonomous systems 57 (5) (2009) 469–483.

[16] D. Silver, J. A. Bagnell, A. Stentz, High performance outdoor navigation from overhead data using imitation learning, in: Robotics: Science and Systems (RSS), 2008.

[17] B. Suger, B. Steder, W. Burgard, Traversability analysis for mobile robots in outdoor environments: A semi-supervised learning approach based on 3d-lidar data, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 3941–3946.

[18] N. Pérez-Higueras, F. Caballero, L. Merino, Teaching robot navigation behaviors to optimal rrt planners, International Journal of Social Robotics 10 (2) (2018) 235–249.

[19] S. Siva, M. B. Wigness, J. G. Rogers, L. Quang, H. Zhang, Nauts: Negotiation for adaptation to unstructured terrain surfaces, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 1733–1740.

[20] M. Wulfmeier, D. Z. Wang, I. Posner, Watch this: Scalable cost-function learning for path planning in urban environments, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016, pp. 2089–2095.

[21] S. Triest, M. G. Castro, P. Maheshwari, M. Sivaprakasam, W. Wang, S. Scherer, Learning risk-aware costmaps via inverse reinforcement learning for off-road navigation, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 924–930.

[22] Z. Xie, Q. Zhang, Z. Jiang, H. Liu, Robot learning from demonstration for path planning: A review, Science China Technological Sciences 63 (8) (2020) 1325–1334.

[23] B. D. Ziebart, J. A. Bagnell, A. K. Dey, The principle of maximum causal entropy for estimating interacting processes, IEEE Transactions on Information Theory 59 (4) (2013) 1966–1980.

[24] K. Konolige, M. Agrawal, M. R. Blas, R. C. Bolles, B. Gerkey, J. Sola, A. Sundaresan, Mapping, navigation, and learning for off-road traversal, Journal of Field Robotics 26 (1) (2009) 88–113.

[25] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, D. Rus, Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 5135–5142.

[26] K. M. Wurm, R. Kümmerle, C. Stachniss, W. Burgard, Improving robot navigation in structured outdoor environments by identifying vegetation from laser data, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2009, pp. 1217–1222.

[27] A. J. Sathyamoorthy, K. M. K. Weerakoon, T. Guan, J. Liang, D. Manocha, Terrapn: Unstructured terrain navigation using online self-supervised learning, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 7197–7204.

[28] T. Guan, Z. He, D. Manocha, L. Zhang, Tnes: terrain traversability mapping, navigation and excavation system for autonomous excavators on worksite, Autonomous Robots 47 (6) (2023) 695–714.

[29] S. Siva, M. B. Wigness, J. G. Rogers, L. Quang, H. Zhang, Self-reflective terrain-aware robot adaptation for consistent off-road ground navigation, ArXiv abs/2111.06742 (2021).

[30] P. Fankhauser, M. Bloesch, M. Hutter, Probabilistic terrain mapping for mobile robots with uncertain localization, IEEE Robotics and Automation Letters 3 (4) (2018) 3019–3026.

[31] T. M. Howard, A. Kelly, Optimal rough terrain trajectory generation for wheeled mobile robots, The International Journal of Robotics Research 26 (2) (2007) 141–166.

[32] B. A. Merhy, P. Payeur, E. M. Petriu, Application of segmented 2-d probabilistic occupancy maps for robot sensing and navigation, IEEE Transactions on Instrumentation and Measurement 57 (12) (2008) 2827–2837.

[33] M. Pivtoraiko, R. A. Knepper, A. Kelly, Differentially constrained mobile robot motion planning in state lattices, Journal of Field Robotics 26 (3) (2009) 308–333.

[34] S. Shimoda, Y. Kuroda, K. Iagnemma, Potential field navigation of high speed unmanned ground vehicles on uneven terrain, in: 2005 IEEE International Conference on Robotics and Automation (ICRA), 2005, pp. 2828–2833.

[35] S. Josef, A. Degani, Deep reinforcement learning for safe local planning of a ground vehicle in unknown rough terrain, IEEE Robotics and Automation Letters 5 (4) (2020) 6748–6755.

[36] G. Kahn, P. Abbeel, S. Levine, Badgr: An autonomous self-supervised learning-based navigation system, IEEE Robotics and Automation Letters 6 (2) (2021) 1312–1319.

[37] D. Dolgov, S. Thrun, M. Montemerlo, J. Diebel, Practical search techniques in path planning for autonomous driving, in: International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR), 2008, pp. 29–34.

[38] J. A. Reeds, L. A. Shepp, Optimal paths for a car that goes both forwards and backwards, Pacific Journal of Mathematics 145 (2) (1990) 367–393.