Autonomous Robotic Exploration with Simultaneous Environment and Traversability Models Learning

Miloš Prágr*, **Jan Bayer**, **Jan Faigl**

Computational Robotics Laboratory, Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czechia

Correspondence*:
Miloš Prágr
pragrmi1@fel.cvut.cz

Manuscript length (excluding figures, tables, algorithms, and bibliography): 11873 words

10 figures, 5 tables

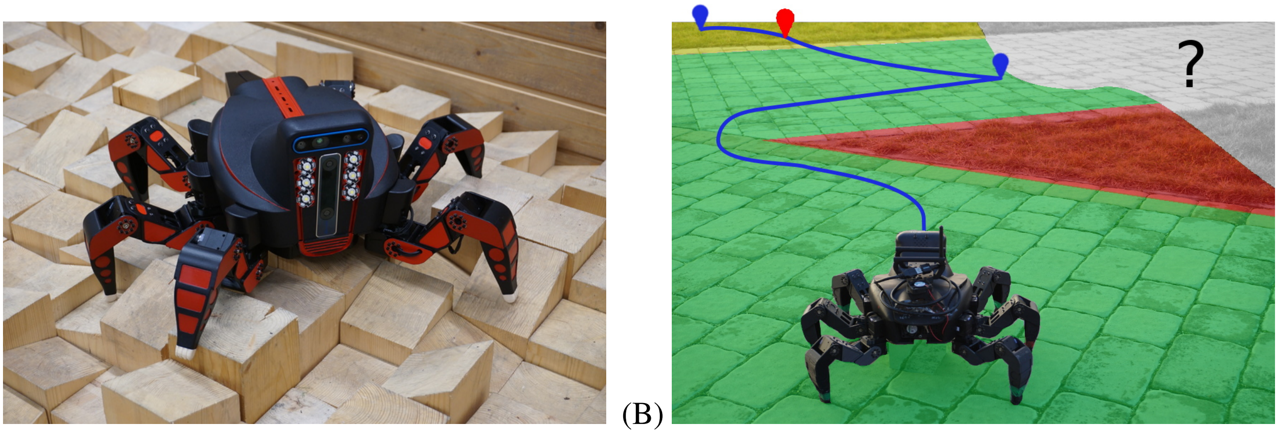
ABSTRACT

In this paper, we address generalized autonomous mobile robot exploration of unknown environments where a robotic agent learns a traversability model and builds a spatial model of the environment. The agent can benefit from the model learned online in distinguishing what terrains are easy to traverse and which should be avoided. The proposed solution enables the learning of multiple traversability models, each associated with a particular locomotion gait, a walking pattern of a multi-legged walking robot. We propose to address the simultaneous learning of the environment and traversability models by a decoupled approach. Thus, navigation waypoints are generated using the current spatial and traversability models to gain the information necessary to improve the particular model during the robot's motion in the environment. From the set of possible waypoints, the decision on where to navigate next is made based on the solution of the generalized traveling salesman problem that allows taking into account a planning horizon longer than a single myopic decision. The proposed approach has been verified in simulated scenarios and experimental deployments with a real hexapod walking robot with two locomotion gaits, suitable for different terrains. Based on the achieved results, the proposed method exploits the online learned traversability models and further supports the selection of the most appropriate locomotion gait for the particular terrain types.

Keywords: mobile robot exploration, active learning, traversability, multi-legged robot, locomotion gait

1 INTRODUCTION

The presented online terrain learning approach is motivated by long-term missions where autonomous robots would improve their operational performance in navigating a priori unknown environments. Some difficult to traverse terrains, such as large rocks, can be identified as obstacles using an observed geometric model of the environment. However, areas that appear flat and thus easy to traverse may, in practice, be



(A) (B)

Figure 1. (A) The hexapod walking robot (courtesy of Forouhar et al. (2021)) (B) and its deployment using the proposed approach. The visualized planned path is to visit determined exploration goals for the spatial (in blue) and traversal cost models (in red). The spatial exploration goals are located close to the boundary of the already explored part of the environment. The traversal cost exploration goals correspond to sites where the terrain traversal cost model can be improved. Since the cost model is already partially learned, the red-tinted turf is known to be hard to traverse, and thus the robot prefers the green-tinted pavement, which is relatively easy to traverse. The yellow-tinted terrain is yet to be experienced by the robot and thus carries the terrain learning goal indicated by the red waypoint. The not-yet-observed area is gray.

hard to traverse due to their terra-mechanical properties, as experienced by NASA's Mars Rover Spirit stuck in soft sand (Brown and Webster, 2010).

In the presented approach, individual terra-mechanical properties are assumed to be partially unknown, and we learn a black-box model to assess the traversability in a particular environment from the terrain appearance (Prágr et al., 2018). Since the scope of the functional relation between the terrain appearance and traversability might be limited to a particular environment, we advocate that on long-term deployments and exploration missions, the terrain models are learned online incrementally (Prágr et al., 2019b) as a part of the mission (Prágr et al., 2019a). Hence, we focus on the exploration of the environment and its terra-mechanical properties represented as the traversal costs that characterize the difficulty of traversing the individual terrains as visualized in Fig. 1. In particular, we consider multi-legged walking robots that can traverse various terrains with different traversal costs (also depending on the particular locomotion gait used), which provide a representative case for demonstrating the benefits of traversability assessment learned online. Compared to the previous work, the presented approach addresses the different locomotion gaits of the robot and distinguishes individual terrain-gait traversal cost models. Besides, the proposed exploration strategy provides a non-myopic (Zlot and Stentz, 2006) solution that takes into account both the spatial exploration and learning of the traversal cost models.

In the proposed approach, the impassable parts of the explored environment are determined by the geometric models using a grid-based elevation map (Bayer and Faigl, 2019). The individual terrain-gait traversal cost models are near-to-far predictors that infer the time to traverse over the traversable areas from their appearance and are learned using the robot's previous experience accrued when traversing similar-appearing terrains using a particular gait. The traversal cost models comprise Gaussian Process (GP) regressors (Rasmussen and Williams, 2006), which predict the traversal costs from the terrain appearance, and Growing Neural Gas (GNG) (Fritzke, 1994) terrain type clustering schemes used to identify similar-appearing terrains. The geometric and traversal cost models are incrementally constructed while exploring the mission environment. The geometric model is continually built from the robot's exteroception, while each traversal cost model accumulates the costs experienced by the robot when moving using the respective locomotion gait. During the deployment, each model continually provides a set of exploration goals to be visited to learn (improve) the model. For several possible goal locations, the exploration strategy is to determine a sequence of the navigational goals to be visited that is addressed as a solution of the *Generalized Traveling Salesman Problem* (GTSP) (Noon, 1988) to provide a non-myopic solution considering the so-called TSP distance cost (Faigl and Kulich, 2013).

The remainder of the paper is organized as follows. In Section 2, we present an overview of the related approaches in mobile robot exploration and traversability assessment. Section 3 formally defines the studied problem of mobile robot exploration with a priori unknown terrain traversal cost assessment. The proposed exploration with online traversal cost learning is presented in Section 4. Section 5 reports on the performed experimental results in simulations and real-world experimental deployments with a multi-legged robot controlled by two motion gaits. In Section 6, we discuss the strong points and limitations of the proposed approach. Section 7 concludes the paper.

2 STATE OF THE ART

This section presents an overview of works related to the proposed approach. First, we focus on the traversability assessment approaches. Then, we survey mobile robot exploration and environment modeling.

2.1 Mobile Robot Traversability

Two questions emerge when reasoning about robot traversability over terrains. First, can the terrain be safely traversed, or should it be avoided? Second, if the terrain is passable, how does it compare to other terrains, i.e., is it easier and safer to traverse? Note that for the sake of clarity, we further denote the binary `true/false` traversability, which determines whether an area is an impassable obstacle or passable terrain, as terrain passability. In contrast, the relative comparison of the traversal difficulty over passable terrains is denoted as assessing the traversal cost. The term traversability is used to describe the notion in general, including both the passability and traversal cost. A review of mobile robot traversability assessment methods can be found in Papadakis (2013) and an overview of learning-based methods for ground robot navigation is in Guastella and Muscato (2021). Hence, we focus on works relevant to how traversability is approached in this paper.

The passability discrimination can be directly incorporated in mapping in the form of occupancy cell grids (Moravec and Elfes, 1985), Gaussian Mixtures (O'Meadhra et al., 2019), GP models (O'Callaghan et al., 2009), or Hilbert maps (Ramos and Ott, 2016). The distinction of terrain passability can be understood as an instance of terrain classification, where terrains are assigned individual classes, and each class carries presumed terra-mechanical properties. For example, some classes can be considered hard-to-traverse vegetation or obstacles (Bradley et al., 2015). Besides terrain classification, terrains can be assigned continuous values describing some observed terrain property such as roughness (Krüsi et al., 2016; Belter et al., 2019) slope (Stelzer et al., 2012), or step height (Homberger et al., 2016; Wermelinger et al., 2016). For continuous measures, passability can be based on thresholding the value as in Stelzer et al. (2012), where the passability is determined by individually thresholding terrain slope, roughness, and step height. Moreover, classes may correspond to a particular robot configuration, such as in Haddeler et al. (2020), where the authors classify terrains into modes of wheeled-legged locomotion.

In instances where the terra-mechanical properties are unknown, and thus terrains' appearance and geometry features are not sufficient to determine their traversability, the traversability can be based on the robot's prior experience with similar terrains. The experience-based measures can be derived from the robot proprioception and described using stability (McGhee and Frank, 1968; Lin and Song, 1993), slippage (Gonzalez and Iagnemma, 2018), vibrations (Bekhti and Kobayashi, 2016), velocity or energy consumption (Kottege et al., 2015). The experience-based approaches describe the traversal cost only over passable terrains since the traversal is needed to acquire the robot experience. An exception worth mentioning is haptic sensing to determine obstacle passability (Baleia et al., 2015), which, however, still relies on the direct interaction of the robot with the terrain.

Since the experience-based approaches use on-location robot experience, they are difficult to employ directly in path planning where it is necessary the evaluate terrain traversability from a distance using only exteroceptive measurements. Near-to-far approaches pair traversability indicators that can be observed only near the robot (such as proprioception or dense short-range measurements) with terrain appearance and geometry that can be observed from farther distances and thus learn to predict traversability from the long-range measurements. Sofman et al. (2006) incrementally learn the relation between dense laser-based features characterizing ground unit traversability and overhead features that can be used to assess traversability from aerial images, while Bekhti and Kobayashi (2016) learn to predict vibration-based traversability from terrain texture. Quann et al. (2020) propose an energy traversal cost regressor considering both terrain position and appearance. Besides, in Mayuku et al. (2021), a self-supervised labeling approach is proposed for a near-to-far scenario, where vibration-based traversal cost is inferred from image data, and the self-supervised data gathering is based on identified terrain classes.

Following the approaches in the literature, we assume that terrain is rigid, and it is possible to distinguish passable terrain and non-traversable obstacles from the terrain geometry using step height similar to Stelzer et al. (2012), or Wermelinger et al. (2016). Hence, this paper focuses on modeling the traversal cost over the determined passable terrains. Moreover, we are motivated by the online cost assessment in mobile robot exploration, where the computational requirements are crucial. Therefore, we avoid high fidelity models, which besides being costly to compute, also rely on plan execution with high precision (such as deterministic foothold placement), which might not be available in practice. The traversal cost is thus learned as a black-box near-to-far model that uses terrain appearance to predict the time to traverse over terrains. Since the scope of the relation between the terrain appearance and traversability might be limited to a particular environment, we incrementally learn the cost predictor by sampling the robot's experience with traversing individual terrains. Similar to the classification in Belter et al. (2019), a color histogram is selected as the terrain appearance descriptor since it is simple to compute and the histograms are sufficiently descriptive to capture multi-colored terrains. Furthermore, we consider locomotion gaits of the employed hexapod walking robot that are suitable for different terrains. Thus, the passable terrain is a terrain traversable by at least one gait, and obstacles are terrain parts that none of the gaits can traverse. We propose a decoupled approach that predicts the traversal cost for each gait independently, and the robot then selects the most cost-efficient gait for each terrain.

Regarding the existing methods, the proposed approach is closest to Haddeler et al. (2020), where modes of the wheeled-legged robot are switched. Besides, the proposed approach is also close to the self-supervised, near-to-far traversability-learning approach proposed by Mayuku et al. (2021). In that regard, the primary contribution of the proposed approach is the integration of active traversability learning in mobile robot exploration, where the robot plans a non-myopic path to improve both the spatial and traversal cost models learned online during the deployment.

2.2 Mobile Robot Exploration and Environment Modeling

Mobile robot exploration is an active perception problem that concerns behaviors where the robot seeks to build a model of a priori unknown environment. The exploration entails the robot seeking areas that are in some capacity unknown to construct a map of the environment. The exploration thus inherently combines localization, navigation, and planning (Schultz et al., 1999) to decide where the robot should go next. Steering the robot navigation to not yet observed areas yields frontier-based exploration (Yamauchi, 1997), where the frontiers represent boundaries between the observed traversable area and the unknown space represented on an occupancy grid (Moravec and Elfes, 1985). Recently, in the octree-based environment model, frontiers are represented as mesh faces with few neighbors (Azpúrna et al., 2021).

Bourgault et al. (2002) and Makarenko et al. (2002) exploit the probabilistic representation on such an occupancy evidence grid, and navigate to maximize the approximated occupancy information gain. Charrow et al. (2015) propose to use Cauchy-Schwarz quadratic mutual information to speed up the information gain computation. Besides, approaches that rely on non-grid-based representation for navigation, such as meshes and topological maps, may retain cell or voxel grids to quantify the information gain (Dang et al., 2020).

In addition to mapping, robots also build models of environment-underlying phenomena that can be temperature models (Luo and Sycara, 2018) or spread of gas (Rhodes et al., 2020). The environment phenomenon can be considered spatial, and the goal is thus to learn the mapping from the position in the environment to the value of the phenomenon. Furthermore, a spatio-temporal model can be considered (Ma et al., 2018) that would require repeatedly visiting particular areas to build the temporal model, which might be needed for changing environments (Krajník et al., 2017).

Spatial-based modeling can be considered as information path planning (Singh et al., 2007), where the goal is to find the most information path through the environment (Hollinger and Sukhatme, 2014) subject to a particular constraint such as the robot energy budget (Binney and Sukhatme, 2012). Informative path planning approaches can be broadly divided into myopic and non-myopic methods. The myopic methods are greedy and plan only with regard to the next goal, while non-myopic methods plan with a longer horizon. For example, in the context of frontier-based mobile robot exploration, seeking the closest frontier is myopic, contrary to path planning to visit all the representatives of the frontiers that is non-myopic (Faigl et al., 2012).

Like seeking frontiers in spatial exploration, the explorer learning an underlying model must actively locate sites to sample novel information. Hence, GP regressors (Rasmussen and Williams, 2006) are particularly suited for active learning since it is relatively straightforward to identify uncertain regions where the model should be improved. GP prediction uncertainty is characterized by the differential entropy of the predicted normal distribution, leading to the characterization of information gained by observing individual areas. However, in practice, directly computing the information gained by possible observations is not feasible due to the number of possible actions, especially for a long planning horizon. Hence, various approximations and sampling strategies have been proposed.

Pasolli and Melgani (2011) propose to either directly seek the most uncertain samples signified by the highest prediction variance or to select areas that are the most remote in the feature space given the GP hyper-parameters. In Viseras et al. (2019), the robot selects paths with high average entropy per sampling to tradeoff informativeness and the number of samplings. Martin and Corke (2014) propose to set the mean function of a GP traversal cost regressors to zero, thus motivating a robot to traverse unknown areas where the predictions are close to the zero mean. The *GP Upper Confidence Bound* (GP-UCB) (Srinivas et al., 2010) is an exploration-exploitation method that combines seeking the most uncertain areas with improving the model around the highest value. It can be used when the learner is interested in finding extreme values of the modeled phenomenon, such as temperature (Luo and Sycara, 2018; Shi et al., 2020). Besides, a depth-first variant of the *Monte Carlo Tree Search* (MCTS) to select anytime informative paths can be employed to consider both differential entropy and upper confidence bound to model sampling informativeness (Guerrero et al., 2021).

Karolj et al. (2020) compute a path to the closest spatial frontier that visits all local sampling locations for a magnetism model by solving the *Traveling Salesman Problem* (TSP) over the respective goal locations. In localization in mapping, Ossenkopf et al. (2019) note that occupancy information gained at an unknown location holds little value and thus weight the occupancy gains by a pose uncertainty (Vallvé and Andrade-Cetto, 2015). Hence, the explorer must address how to combine the occupancy and pose uncertainties. In Bourgault et al. (2002) and Stachniss et al. (2005), the total exploration utility is a linear combination of the occupancy uncertainty and the robot localization uncertainty represented using the differential entropy based on its position distribution. In Carrillo et al. (2018), it is argued that combining Shannon's discrete and differential entropies is neither practical nor sound since the differential entropy is neither invariant under a change of variable nor dimensionally correct. Therefore, both quantities may differ significantly in value. Consequently, Carrillo et al. (2018) propose to use the localization uncertainty to weigh the Rényi entropy (Rényi, 1961) of the occupancy grid.

Based on the literature review on exploration approaches, we propose to generalize the previous work (Prágr et al., 2019a) towards a non-myopic approach. The therein proposed method combines active learning of traversal cost over terrains with spatial exploration using a greedy approach. The approximated spatial information gains and cost models are derived from Shannon's discrete and differential entropies, respectively. Considering the reasoning of Carrillo et al. (2018), we avoid a direct combination of these two values in this paper. Besides, we aim to build a modular system that supports the learning of models that range from the spatial map and cost predictors used in this paper to temperature and pollution models. Hence, instead of creating a combined information gain utility function using the Rényi entropy, which is suitable for the combination of a map and robot's localization model used by Carrillo et al. (2018), we elect to use a policy that combines the spatial exploration and cost learning goals (and goals reported by any additional model), similarly to the approach proposed by Karolj et al. (2020).

However, unlike the therein-built magnetism model, a spatial GP, we assume that the terrain traversal cost correlates with the terrain appearance. Therefore, the GP regressor infers the cost from the terrain feature descriptors instead of the terrain location. Consequently, rather than terrains nearby, sampling the cost to traverse an unknown terrain primarily affects the predictions over similarly appearing terrains close in the feature space. The affected terrains are determined using a terrain clustering scheme. *Incremental Growing Neural Gas* (IGNG) (Prudent and Ennaji, 2005) is used to continually construct the terrain class structure, in which each class is assigned traversal cost and sampling reward (information gain) based on the GP's predictions. As a result, we model the computation of the goal visit sequence as an instance of the *Generalized TSP* (GTSP) (Noon, 1988) (also called the Set TSP), which is a variant of the TSP where nodes are grouped into mutually exclusive and exhaustive sets. The problem is then to visit each set instead of visiting each node. In the context of the proposed exploration approach, the individual nodes correspond to possible sampling locations, and the sets are either terrain classes extracted from the cost prediction model or places where the robot can observe areas unknown to the spatial model.

The problem of mobile robot exploration with traversal cost learning is defined in the next section, while the strengths and weak points of the proposed approach are further discussed in Section 6.

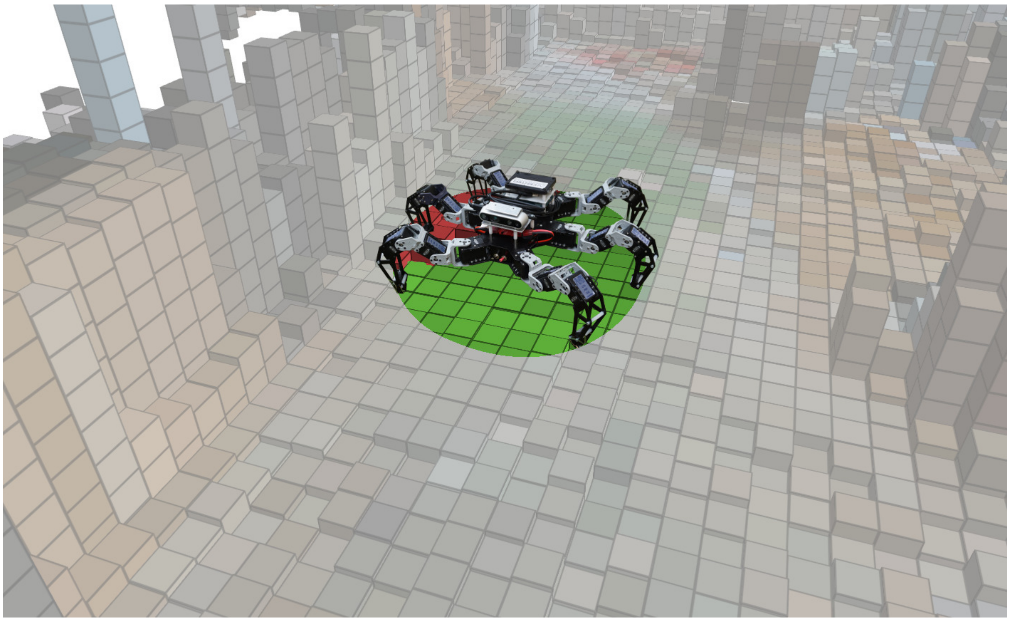


Figure 2. The footprint around the robot position covers the cells with potential multi-legged walking robot footholds.

3 PROBLEM SPECIFICATION

The addressed exploration using an autonomous hexapod walking robot combines spatial exploration with active learning of terrain traversal cost models. The environment is modeled as a 2D grid $\mathbb{W} \subset \mathbb{R}^2$ with cells $\nu \in \mathbb{W}$ with size d_ν corresponding to the size of the robot foothold. The position of the robot p^{robot} is discretized as ν^{robot} within the grid that is at the center of the robot's circular footprint with radius r_{robot} covering all the potential robot's footholds as shown in Fig. 2. Any path ψ can be decomposed to a sequence of neighboring cells as

$$\begin{aligned} &\psi = (\nu_1, \nu_2, \cdots, \nu_n), \\ \text{s.t.}& \\ &\forall i \in 1, \cdots, n: \ \pi(\nu_i) = 1, \\ &\forall i \in 1, \cdots, n-1: \ \nu_{i+1} \in 8\text{nb}(\nu_i), \end{aligned} \tag{1}$$

where n is the number of cells in the respective sequence, the function $8\text{nb}(\nu)$ lists the cells in the 8-neighborhood of ν, and $\pi(\nu) = 1$ indicates that the cell ν is passable. Besides, the robot can use a discrete set of walking gaits $\mathbb{G}$, and it is assumed that the gait changes occur instantaneously at the particular grid cells $\nu \in \mathbb{W}$.

The robot desires to move through the environment as efficiently as possible with respect to (w.r.t.) the cost C. Therefore, it moves along the cheapest path between ν and ν'.

$$\psi^*(\nu, \nu') = \text{argmin}_{\psi \in \Psi(\nu, \nu')} C(\psi), \tag{2}$$

where $\Psi(\nu, \nu')$ is the space of all paths from ν to ν'. The cost $C(\psi)$ of traversing ψ represents a generic path cost such as time to traverse or expected consumed energy, and without the loss of generality, the time to traverse is the cost of choice in this paper. It is assumed that the cost is additive, thus permitting to combine the costs of two consecutive path segments ψ_a and ψ_b into the cost of the combined path $\psi_a \oplus \psi_b$ as

$$C(\psi_a \oplus \psi_b) = C(\psi_a) + C(\psi_b), \tag{3}$$

where $\oplus$ denotes the concatenation of the paths. The cost of a path is decomposed to the sequence of costs to traverse from passable cell ν_a to its neighbor ν_b

$$C(\psi) = \sum_{i=1}^{n-1} \|(\nu_i, \nu_{i+1})\| c(\nu_i, \nu_{i+1}), \tag{4}$$

where $\|(\nu_a, \nu_b)\|$ is the Euclidean distance between the cells (i.e., either d_ν or $\sqrt{2}d_\nu$), and $c(\nu_a, \nu_b)$ is the per-meter cost of traversing from ν_a to ν_b.

In the spatial exploration, the robot builds the geometry model $\mathcal{P}$, which provides the cell passability assessment $\pi(\nu)$. It is assumed that the geometry is sufficient to distinguish the passable areas; hence, the passability model $\mathcal{P}$ is constructed directly from the continually streamed exteroceptive measurements (observed point clouds z^{pcd}).

3.1 Traversal Cost Modeling

The traversal cost is assumed to be too complex to be assessed only from the terrain geometry. In this paper, the task is to learn a traversal cost predictor $\mathcal{C}$ that models the cost as a function of terrain appearance. The cost assessments are used in path planning w.r.t. (4). Besides, the cost model is also responsible for selecting the gaits suitable for the particular terrains traversed by the robot. Since the robot position is abstracted as the center of its circular footprint, the predictor's per-meter-cost predictions are conservative estimates that take into account all the cells on the footprint. The cost predictor is learned online during the exploration from the robot experience that comprises the cost z^c experienced by the robot when traversing terrain described by the terrain appearance descriptor ta using gait g.

The learned model is compared to the uninformed baseline that represents a robot that only explores the spatial map and does not learn the cost models, and thus uses the optimistic flat cost model

$$\hat{c}(\nu_a, \nu_b) = \frac{1}{v_{\text{max}}}, \tag{5}$$

where v_{max} is the maximum robot velocity over all $g \in \mathbb{G}$. Notice that in planning, the particular value of v_{max} is not relevant as long as it is positive since it only scales the total cost, thus not affecting the planning decisions. The baseline selects the gaits reactively, using the fast gait capable of reaching v_{max} by default and switching to slower yet rough-terrain-capable gaits when the robot gets stuck on the traversed terrain.

The proposed approach is evaluated in model scenarios as follows. First, the robot is set to explore the environment $\mathbb{W}$ and thus incrementally learn the model $\mathcal{C}$. Then, the learned and baseline models are used in navigating the robot between a set of benchmark coordinates in $\mathbb{W}$ and the total cost C experienced by the robot (i.e., the time needed to move between the coordinates) using the particular model is considered to be the benchmark value.

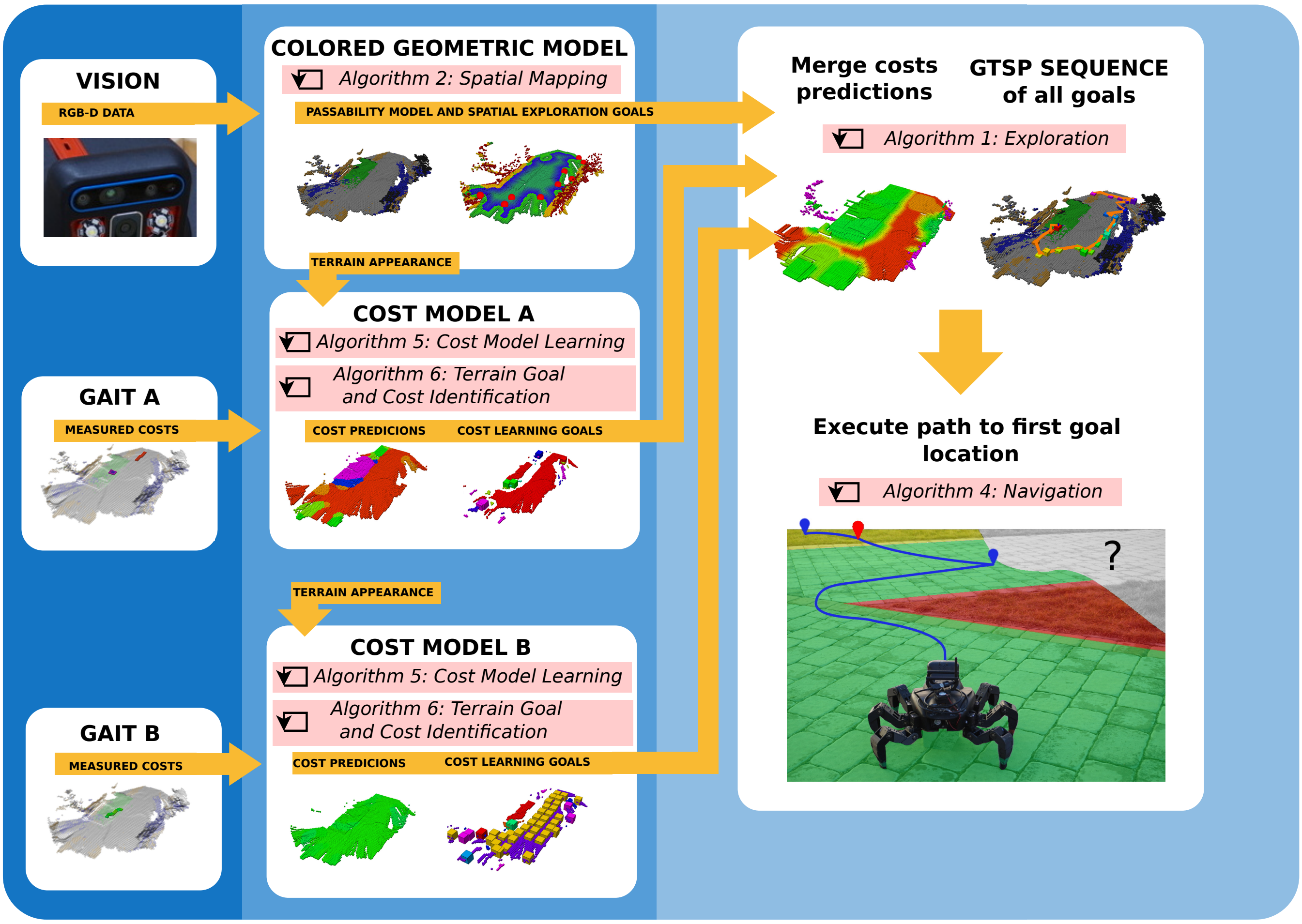


Figure 3. An overview of the proposed exploration system. The robot uses the RGB-D data to build the color elevation model of the environment, in which it identifies the passable areas (Alg. 2). The terrain appearance stored in the model is paired with the costs experienced by the robot to learn the traversal cost models for the individual locomotion gaits (Algs. 5 and 6). The cost predictions for the individual gaits and the terrain passability are used to plan the exploration path in a TSP sequence (Alg. 1) over every goal reported by the geometric and cost models. The robot navigates to the first goal in the sequence (Alg. 4).

4 PROPOSED SYSTEM FOR ACTIVE TERRAIN LEARNING IN EXPLORATION

In this section, we describe the proposed system for active terrain learning and exploration, which is overviewed in Fig. 3. During the exploration, which yields the spatial geometric passability model $\mathcal{P}$, the goal of the robot is also to learn the traversal cost model $\mathcal{C}$. The geometric passability model $\mathcal{P}$ describes the shape of the environment and thus areas passable by the robot. The traversal cost model is decomposed into the set of models $\mathcal{C} = \mathcal{C}^{\mathbb{G}} = \{\mathcal{C}^g\}_{g\in\mathbb{G}}$, where each traversal cost model $\mathcal{C}^g$ predicts the costs associated with traversing the passable terrain using the gait $g \in \mathbb{G}$. The respective cost predictors are *Gaussian Process* (GP) regressors (Rasmussen and Williams, 2006) that use terrain appearance to infer the robot-experienced traversal cost accrued during the deployment. Each GP is coupled with the *Incremental Growing Neural Gas* (IGNG) (Prudent and Ennaji, 2005) that clusters similarly appearing terrains and hence identifies terrain types not yet visited by the robot. The exploration problem is modeled as an open-ended instance of the *Generalized Traveling Salesman Problem* (GTSP) (Noon, 1988), a variant of the TSP where the vertices are organized in disjoint sets, and each set is visited once. In this paper, each set corresponds to an exploration or learning goal (a set of sampling sites) yielded by the spatial or cost model.

Table 1. Used Symbols

Description	Symbol	Description	Symbol
World gridmap model	$\mathbb{W}$	Gridmap cell	ν
Gridmap cellsize	d_ν	Current robot position	ν^{robot}
Robot footprint radius	r_{robot}	Cell ν passability	$\pi(\nu)$
Path	ψ	Optimal path	ψ^*
Walking robot gait	g	Robot gait set	$\mathbb{G}$
Cost (time to traverse)	C	Per-meter cost	c
Geometric passability model	$\mathcal{P}$	Cost model	$\mathcal{C}$
Measured cost	z^c	Maximum robot velocity	v_{max}
Colored elevation gridmap	$\mathcal{M}_{2.5D}$	Robot sensor range	r_{sensor}
Terrain appearance desciptor	ta	Descriptor radius	r_{hist}
Spatial clustering radius	c_{radius}	Cluster min cells	$c_{\text{min cells}}$
Cost model, all gaits	$\mathcal{C}^{\mathbb{G}}$	Cost model, particular gait	$\mathcal{C}^g$
Cost prediction, all gaits	$\hat{c}$	Cost prediction, particular gait	$\hat{c}_{\mathcal{C}^g}$
Distance transform per-meter loss	c_{loss}	Cost measurement variance	σ^2_{sense}
Cost measurement filter initial variance	σ_0^2		
GP regressor	$\mathcal{R}$	GP learning set	$\mathcal{L}$
GP prediction mean	$\hat{\mu}_c$	GP prediction variance	$\hat{\sigma}_c^2$
Prediction uncertainty / GP entropy	H	High cost in cost transform	c_{high}
Min learning set size	$n_{\mathcal{L}}^{\text{min}}$	GP model noise variance	σ_ϵ^2
Exponential kernel lenghtscale	l	Exponential kernel output variance	σ_s
Maximum allowed cost	c_{max}		
Terrain class model	$\mathcal{T}$	Terrain class	T
Approximated cost information gain	$I_{\mathcal{C}}$	Terrain class uncertainty threshold	$H_{\mathcal{C}}^{\text{GT}}$
Min GT terrain type size	m_T	Sampling lattice	S
Sampling lattice point	p_S	Sampling lattice size	d_S
Goal set	Γ	Goal	γ
Passability goal set	$\Gamma_{\mathcal{P}}$	Cost goal set, all gaits	$\Gamma_{\mathcal{C}}^{\mathbb{G}}$
Cost goal set, particular gait	$\Gamma_{\mathcal{C}}^g$	TSP distance matrix	D
Current exploration goal	ν_E^*	Current exploration path	ψ_E
Enforced sampling gait	g^{enforced}	Gait sampling duration	Δt_{sample}
IGNG structure	Ω	IGNG measurement	x
IGNG neuron set	Ω_{neurons}	IGNG connection set	$\Omega_{\text{connections}}$
IGNG neuron	ω	IGNG adaptation threshold	σ^{IGNG}
IGNG winner warp rate	ϵ_1^{IGNG}	IGNG neighbor warp rate	$\epsilon_{\text{nb}}^{\text{IGNG}}$
IGNG neuron mature age	$a_{\text{mature}}^{\text{IGNG}}$	IGNG connection maximum age	$a_{\text{max}}^{\text{IGNG}}$
Terrain type erosion steps	$n_{\text{erode}}^{\text{steps}}$	Terrain type dilation steps	$n_{\text{dilate}}^{\text{steps}}$
Terrain type dilation size	$n_{\text{dilate}}^{\text{size}}$		

In the rest of the section, we describe the exploration process. The symbols used in the description are listed in Tbl. 1. First, we show how the GTSP is used to find the exploration path. Then, we show the geometric environment model in detail and the related passability model $\mathcal{P}$, the traversal cost models $\mathcal{C}^g$, and their use to find the exploration goals.

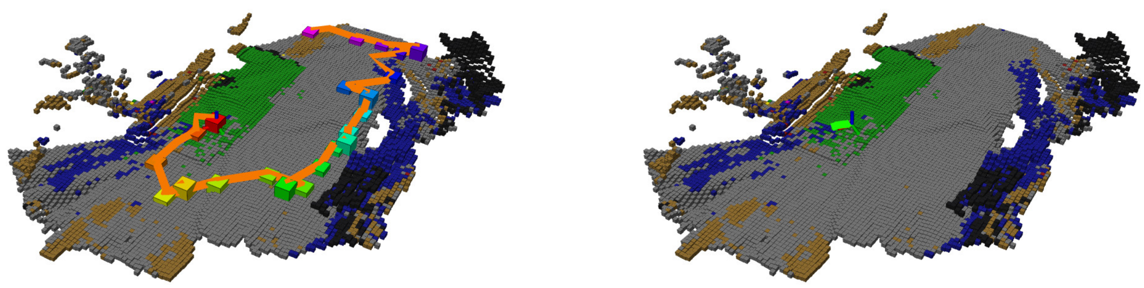
4.1 Exploration

The robot explores the passability model $\mathcal{P}$ and learns the traversal cost models $\mathcal{C}^{\mathbb{G}}$ by visiting the exploration $\Gamma_{\mathcal{P}}$ and cost learning $\Gamma_{\mathcal{C}}^{\mathbb{G}}$ goals, which are continually yielded by the respective models. Each goal $\gamma \in \Gamma_{\mathcal{P}} \cup \Gamma_{\mathcal{C}}^{\mathbb{G}}$ is associated with a set of sites (cells) $\gamma = \{\nu_i\}_{i=0}^{|\gamma|}$ where the robot can improve its models by sampling the respective goal. The robot needs to visit one of the corresponding locations to sample the goal. Geometric model goals $\gamma \in \Gamma_{\mathcal{P}}$ are located at singular sites $\gamma = \{\nu\}$, where the robot can improve the spatial model by observing new areas. Each traversal cost model goal $\gamma \in \Gamma_{\mathcal{C}}^{\mathbb{G}}$, where $\Gamma_{\mathcal{C}}^{\mathbb{G}} = \cup_{g\in\mathbb{G}} \Gamma_{\mathcal{C}}^{g}$, is associated with a set of sites $\gamma = \{\nu_i\}_{i=0}^{|\gamma|}$ at which the robot can improve the model by experiencing novel gait-terrain costs. The areas covered by the individual goals in a given cost model are designed to be disjoint. Thus, sampling the traversal cost model at a site corresponding to the goal ${}^{1}\gamma_{\mathcal{C}}^{g} \in \Gamma_{\mathcal{C}}^{g}$ provides no, or severely limited, information regarding the traversal cost model at a site corresponding to a different goal ${}^{2}\gamma_{\mathcal{C}}^{g} \neq {}^{1}\gamma_{\mathcal{C}}^{g}$. On the other hand, the passability and traversal cost models are considered independent. Sampling at one particular site might improve both models since the robot can observe previously unseen areas while experiencing untraversed terrain. However, two cost models cannot be improved at once since the robot can only experience the cost for the currently used gait.

Given the current robot position ν_t^{robot} and models $\mathcal{P}_t$ and $\mathcal{C}_t^{\mathbb{G}}$ at any time t during the exploration, the robot selects a shortest exploration path $\psi_E(p_t^{\text{robot}}, \mathcal{P}_t, \mathcal{C}_t^{\mathbb{G}})$ that visits at least one site corresponding to each goal. The path planning is modeled as an instance of the GTSP, where vertices (sites) are organized in disjoint sets (goals), and each set is visited exactly once. The distance matrix D describes the costs of paths between the individual sites, including the distances between the current robot position and the goal sites

$$D(\nu, \nu') = \hat{C}(\psi^*(\nu, \nu')). \tag{6}$$

Two transforms are applied to the distance matrix D to create an open instance of the GTSP. First, the robot does not need to return to its current position after exploring the environment. Hence, the problem is transformed by setting the cost to reach the current robot position from any goal as zero $\forall \gamma \in \Gamma_{\mathcal{P}} \cup \Gamma_{\mathcal{C}}^{\mathbb{G}}, \forall \nu \in \gamma : D(\nu_\gamma, \nu^{\text{robot}}) = 0$. Second, we apply the Noon-Bean transformation (Noon and Bean, 1993) to transform an instance of the GTSP into an instance of the TSP.



(A) (B)

Figure 4. An example of a planned exploration path; (A) the global path over the sequence of goals determined by the TSP solver; (B) the local path to the first goal.

The open instances of the transformed TSP are solved by the LKH solver (Helsgaun, 2000), a heuristic solver with asymptotic time complexity bounded by $\mathcal{O}(m^{2.2})$, where m is the number of vertices, which has been found sufficient for updates with tens of goal sites. The solver returns the sequence of sites $(\nu^{\text{robot}}, \nu_0, \nu_1, \cdots, \nu_n)$ to be visited through the environment, see Fig. 4A, where n is the number of goals and each site ν_i corresponds to a different goal. The robot navigates towards the first site of the sequence and its current exploration goal ν_E^* becomes $\nu_E^* = \nu_0$, see an example of the path in Fig. 4B.

The plan is recomputed on-demand either when there is a change in the goal set or as a result of reaching the current goal. Moreover, upon reaching a cost model goal, the robot switches to the model's respective gait g^{enforced} and is forced to move forward for Δt_{sample} (or until an obstacle is reached) to sample the traversal cost over the terrain. The exploration ends when every model reports zero goals. The exploration process is summarized in Alg. 1.

Algorithm 1: Exploration

Input: $\nu^{\text{robot}}_{1,\dots,n}$ – Robot positions; $z^{\text{pcd}}_{1,\dots,n}$ – RGB-D measurements; $z^{c}_{1,\dots,n}$ – Cost measurements.
Output: $\mathcal{P}$ – Passability model; $\mathcal{C}$ – Cost model.

$\mathcal{M}_{2.5D}, \Gamma_{\mathcal{P}} \leftarrow$ start process: `spatialExploration` $(z^{pcd}_{1,\dots,n})$ // Init. spatial modeling (Alg. 2).
for $g \in \mathbb{G}$ **do** // For each gait.
  $\mathcal{R}^g, \mathcal{L}^g \leftarrow$ start process: `learning` $(\mathcal{M}_{2.5D}, z^{c}_{1,\dots,n})$ // Init. cost model learning (Alg. 5).
  $\Gamma^g_{\mathcal{C}}, \mathcal{T}^g \leftarrow$ start process: `terrainTypeClustering` $(\mathcal{M}_{2.5D}, \mathcal{R}^g, \mathcal{L}^g)$ // Start terrain clustering and goal identification (Alg. 6).
$\nu^*_E \leftarrow \varnothing$ // Set the current exploration goal.
$g^{\text{enforced}} \leftarrow \varnothing$ // Set the sampling-enforced gait.
$\psi_E \leftarrow \varnothing$ // Set the exploration path.
start process: `navigate` $(\mathcal{M}_{2.5D}, \psi_E, \nu^{robot}_{1,\dots,n}, g^{enforced})$ // Init. navigation (Alg. 4).
$finished \leftarrow$ `false`
while not $finished$ **do**
  `getLatest` $(\mathcal{M}_{2.5D}, \Gamma_{\mathcal{P}}, \forall g \in \mathbb{G} : \mathcal{R}^g, \Gamma^g_{\mathcal{C}}, \mathcal{T}^g)$ // Get the current models and goals.
  if ν^*_E has been reached and $\exists g \in \mathbb{G} : \nu^*_E \in \gamma^g_{\mathcal{C}}$ **then** // If the robot reached a cost model goal.
    $\nu^*_E \leftarrow$ `forwardMotionGoal()` // Sample the reached goal.
    $\psi_E \leftarrow$ `planToStraight` (ν^*_E) // Plan straight sampling path.
    $g^{\text{enforced}} \leftarrow g$ // Force the robot to use the particular gait.
  else if $\Gamma_{\mathcal{P}} \cup \Gamma^{\mathbb{G}}_{\mathcal{C}}$ has changed or ν^*_E has been reached **then** // Else if the goal has changed or current goal is reached.
    $(\nu^{\text{robot}}_i, \nu_0, \nu_1, \cdots, \nu_{|\Gamma_{\mathcal{P}} \cup \Gamma^{\mathbb{G}}_{\mathcal{C}}|}) \leftarrow$ `solveGTSP` $(\Gamma_{\mathcal{P}} \cup \Gamma^{\mathbb{G}}_{\mathcal{C}}, \mathcal{M}_{2.5D}, \nu^{robot}_i)$ // Solve the GTPS.
    $\nu^*_E \leftarrow \nu_0$ // Update the current exploration goal.
    $\psi_E \leftarrow$ `planToOptimal` (ν^*_E, r_{robot}) // Plan cheapest path to the goal.
    $g^{\text{enforced}} \leftarrow \varnothing$ // Allow the robot to use any gait.
  else // Otherwise, check whether the exploration is finished.
    $finished \leftarrow \mathcal{M}_{2.5D} \neq \varnothing \ \wedge \ \Gamma_{\mathcal{P}} = \{\}$ // Continue exploring if spatial model is not intialized or reports goals.
    for $g \in \mathbb{G}$ **do** // For each gait.
      $finished \leftarrow finished \ \wedge \ \mathcal{R}^g \neq \varnothing \ \wedge \ \Gamma^g_{\mathcal{C}} = \{\}$
      // Continue exploring if the gait-terrain cost model is not initialized or reports goals.
  $\mathcal{P} \leftarrow \mathcal{M}_{2.5D}$ // Report the grid map as the passability model.
  $\mathcal{C} \leftarrow \{\mathcal{R}^g, \mathcal{T}^g\}_{g \in \mathbb{G}}$ // Report the regressors and class sets as the cost model.
  return $\mathcal{P}, \mathcal{C}$

4.2 Environment Geometry & Passability Model

The grid environment $\mathbb{W}$ is represented by the colored elevation grid map $\mathcal{M}_{2.5D}$ with the cell size d_ν. The grid map is built online during the exploration according to Alg. 2 using the robot's range measurements and RGB camera images. The elevation at each cell $\nu \in \mathcal{M}_{2.5D}$ is obtained by fusing the localized range measurements z^{pcd}_i into the grid map using one dimensional Kalman filter as in Fankhauser et al. (2014) or Bayer and Faigl (2020). The localization of the robot, and thus also the localization of the range measurements, is considered to be solved by the Intel RealSense T265 tracking camera, which estimates the robot's full 6 DOF pose based on visual

Simultaneous Localization and Mapping supported by an inbuilt Inertial Measurement Unit[1]. The grid map is used as a model of the terrain geometry to identify passable places. It also captures the color of the terrain texture that is processed to compute the terrain appearance descriptors.

Algorithm 2: Spatial Exploration

Input: $z^{\text{pcd}}_{1,\ldots,n}$ – RGB-D measurements.
Output: $\mathcal{M}_{2.5D}$ – Elevation grid map; $\Gamma_{\mathcal{P}}$ – Passability goals.

```
while exploration is running do
    M_2.5D ← updateMapByRangeMeasurements(M_2.5D, z_i^pcd)   // Fuse range and color measurements.
    M_2.5D ← recomputePassability(M_2.5D)                    // Update cell passability.
    M_2.5D ← recomputeEntropy(M_2.5D)                        // Update cell entropy.
    Γ_P ← clusterEntropyRepresentatives(M_2.5D)              // Cluster entropy representatives (Alg. 3).
    reportLatest(M_2.5D, Γ_P)
```

We define the passability of the cell $\nu \in \mathcal{M}_{2.5D}$ as the probability $\pi(\nu)$ that the cell ν can be traversed by the robot. The probability itself is based on the observed roughness of the terrain computed based on Bayer and Faigl (2021) as

$$\rho(\nu) = \max_{\nu' \in 8\text{nb}(\nu)} \Delta(\nu, \nu'), \tag{7}$$

where $8\text{nb}(\nu)$ is the 8-neighborhood of the cell ν, and the step height $\Delta(\nu_a, \nu_b)$ is

$$\Delta(\nu_a, \nu_b) = |\,\text{elevation}(\nu_a) - \text{elevation}(\nu_b)| \tag{8}$$

with $\text{elevation}(\nu)$ denoting the estimated height of the terrain at ν. The probability that the robot can pass a cell ν is

$$\pi(\nu) \begin{cases} 0 & \text{if } \rho(\nu) > \rho_{\text{obstacle}} \\ 1 & \text{otherwise} \end{cases}, \tag{9}$$

where the threshold ρ_{obstacle} represents the lowest obstacle to be detected. An example of the grid map is shown in Fig. 5A.

In active perception scenarios, the information about the terrain model $\mathcal{M}_{2.5D}$ gained by observing the cell ν' is evaluated by entropy based on the known passability. Since the distribution of the passability is binary and depends on the 8-neighborhood of the cell, information gained by observing ν' with unknown height is approximated as

$$I^{\text{cell}}_{\mathcal{P}}(\nu') \approx \frac{k(\nu') + 1}{9}, \tag{10}$$

where $k(\nu)$ is the number of the unknown cells in the neighborhood of ν. Thus, the expected information gained by perceiving the terrain from the position of the cell ν can be expressed as

$$I^{\text{model}}_{\mathcal{P}}(\nu) = \sum_{\nu' \in \delta(r_{\text{sensor}}, \nu)} \begin{cases} I^{\text{cell}}_{\mathcal{P}}(\nu') & \text{if observable}(\nu, \nu') \\ 0 & \text{otherwise} \end{cases}, \tag{11}$$

where $\delta(r_{\text{sensor}}, \nu)$ is the sensor range r_{sensor}-large neighborhood of ν, the function $\text{observable}(\nu, \nu')$ returns `true` if and only if the cell ν' is observable from ν, which is determined by casting

[1] In the simulated experiments, the localization is provided by the simulator.

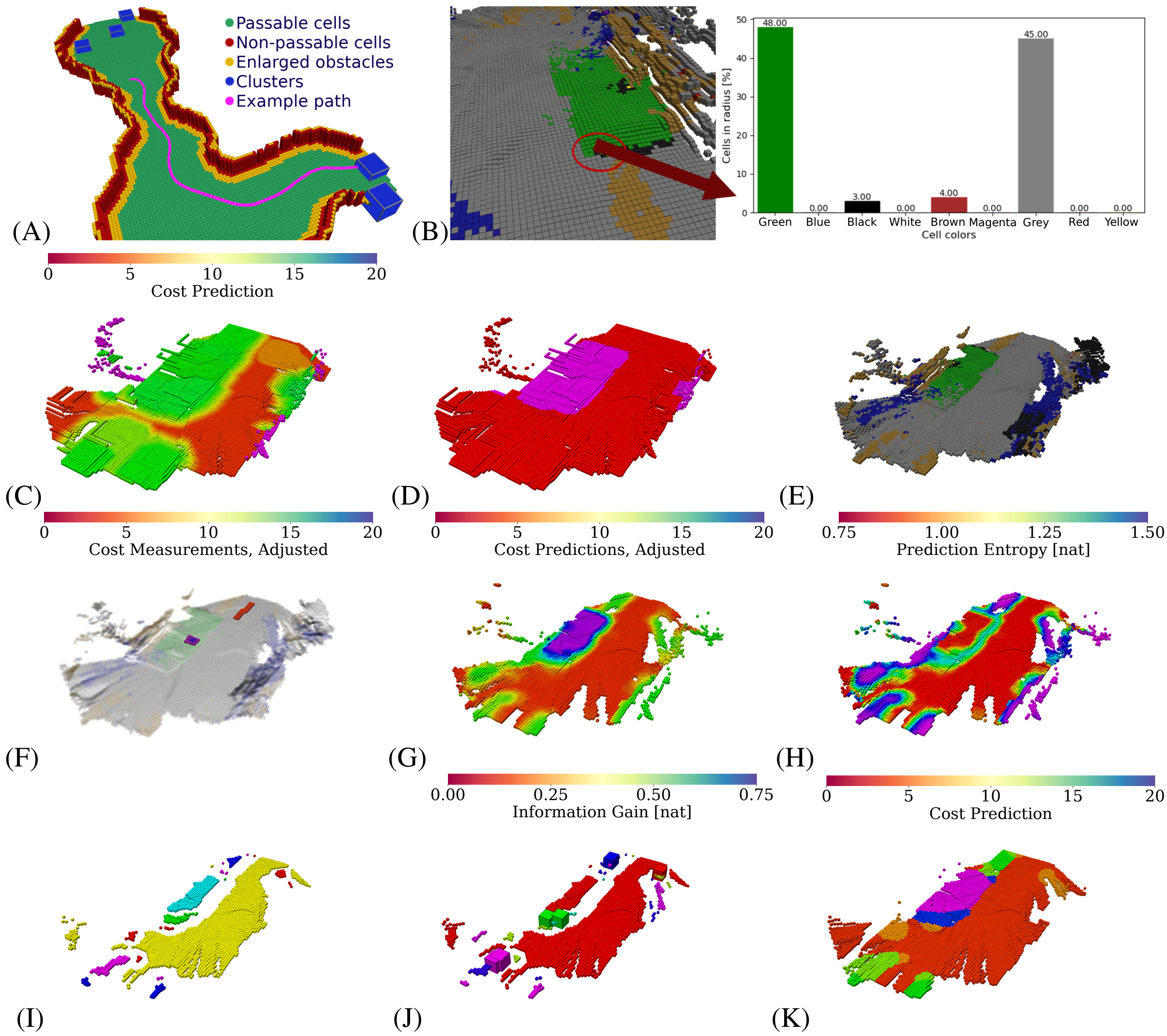


Figure 5. Illustration of the color-geometric and cost models. (A) A visualization of the online built geometrical model with marked passability and clusters based on the cells with non-zero information according to the shown color legend; (B) terrain appearance descriptor calculated as a histogram of cell colors. The costs used in path planning; (C) the minimal cost over gaits after the distance transform; (D) the respective cheapest gait (gaits in red and purple). (E) The colors used to build the color histogram terrain appearance descriptor; (F) the measured costs used for learning the GP (adjusted by hyperbolic tangent), visualized over the terrain appearance; (G) the raw GP cost prediction; (H) the GP prediction uncertainty. (I) The terrain clusters (arbitrary colors used to distinguish clusters); (J) the information gained with terrain learning goals (goal colors corresponding to clusters); (K) the cluster costs used in planning.

a ray from ν to ν' in the current elevation map $\mathcal{M}_{2.5D}$. Using all the cells with non-zero entropy in the TSP formulation is computationally intensive. Thus, we propose to spatially cluster the entropy to generate a limited number of spatial entropy representants by Alg. 3.

Algorithm 3: Cluster Entropy Representatives

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map.
Output: $\Gamma_{\mathcal{P}}$ – Passability goal set.

Procedure `cluster`($\mathcal{M}_{2.5D}$)
  $A \leftarrow \varnothing$ // Init. set of clusters.
  for $\nu \in \mathcal{M}_{2.5D} : I_{\mathcal{P}}^{model}(\nu) > 0$ **do** // For each map cell with non-zero entropy.
    if $A = \varnothing$ **then** // If no clusters in set.
      $A \leftarrow A \cup \{\{\nu\}\}$ // Create a new cluster.
    else
      $d \leftarrow$ `distanceToClosestCluster`(ν, A)
      if $d < c_{radius}$ **then**
        `addToClosestCluster`(ν, A) // Add point to existing cluster.
      else
        $A \leftarrow A \cup \{\{\nu\}\}$ // Create new cluster.
  $\Gamma_{\mathcal{P}} \leftarrow \varnothing$ // Init. cluster representants.
  for $A_i \in A$ **do** // For each clusters.
    if $|A_i| > c_{min\ cells}$ **then**
      $\Gamma_{\mathcal{P}} = \Gamma_{\mathcal{P}} \cup \{$`averageCoordinateCell`$(A_i)\}$ // Create new representatives.
  return $\Gamma_{\mathcal{P}}$

Besides the terrain geometry, the grid map $\mathcal{M}_{2.5D}$ also carries the terrain texture calculated by the following approach. Each cell is provided a 10-bit color by projecting the camera image to the map $\mathcal{M}_{2.5D}$. Then, the color space is shrunk to 9 different colors, defined by color prototypes listed in Fig. 5B. The relative amount of the cell colors within the radius r_{hist} matched to the selected color prototypes are used to build a 9-dimensional terrain appearance descriptor $\mathrm{ta}(\nu)$ for each cell $\nu \in \mathcal{M}_{2.5D}$, which is visualized as a color histogram in Fig. 5B.

4.3 Traversal Cost Model

The cost model $\mathcal{C}$ predicts the per-meter traversal cost c over observed areas deemed passable by the geometric passability model $\mathcal{P}$. The traversal cost model predicts the traversal cost from terrain appearance. Since the robot position is abstracted as the center of its circular footprint, the $\mathcal{C}$'s per-meter-cost predictions are conservative estimates that take into account all the cells on the footprint

$$\hat{c}(\nu_a, \nu_b) = \max_{\nu' \in \delta(r_{\text{robot}}, \nu_a)} \hat{c}(\nu'), \tag{12}$$

where $\delta(r, \nu)$ lists all cells within the r-radius of cell ν, and $\hat{c}(\nu)$ is the $\mathcal{C}$ cost estimate over cell ν. An example of the traversal cost assessment is depicted in Fig. 5C.

The cost $\hat{c}$ is reported for the whole model set $\mathcal{C} = \mathcal{C}^{\mathbb{G}} = \{\mathcal{C}^g\}_{g \in \mathbb{G}}$, since it is the best gait-terrain cost

$$\hat{c}(\nu) = \min_{g \in \mathbb{G}} \hat{c}_{\mathcal{C}^g}(\nu), \tag{13}$$

where each gait-terrain cost $\hat{c}_{\mathcal{C}^g}$ is the prediction of the particular model $\mathcal{C}^g$. Besides, when navigating through the environment, the robot selects its gait w.r.t. the minimization in (13) as depicted in Alg. 4. An example of gait selection is visualized in Fig. 5D. A distance transform with c_{loss} per-meter-loss is used over the cell grid with the best-gait costs $\hat{c}(\nu)$ to dissuade the robot from navigating areas near terrain boundaries where frequent gait changes are likely.

Algorithm 4: Navigate

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; ψ_E – Exploration path; $\nu^{\text{robot}}_{1,\dots,n}$ – Robot positions; g^{enforced} – Enforced sampling gait.

while exploration is running **do**
- `getLatest(`$\mathcal{M}_{2.5D}, \psi_E, g^{\textit{enforced}}$`)`
- **if** $g^{\textit{enforced}} \neq \varnothing$ **then** // If the robot is forced to sample a gait-terrain model.
  - `setGait(`$g^{\textit{enforced}}$`)` // Use the particular gait.
- **else**
  - $g^{\text{best}} \leftarrow \operatorname{argmin}_{g \in \mathbb{G}} \max_{\nu' \in \delta(r_{\text{robot}}, \nu_i^{\text{robot}})} \hat{c}_{\mathcal{C}^g}(\nu')$ // Find the best gait for the robot position.
  - `setGait(`$g^{\textit{best}}$`)` // Use the particular gait.
- `walkAlong(`ψ_E`)` // Continue along the exploration path.

Each gait-terrain model $\mathcal{C}^g$ comprises the cost regressor $\mathcal{R}$ and the terrain type clustering $\mathcal{T}$. In $\mathcal{R}$, we use GP regression to predict the traversal costs because it provides the predicted values and models the prediction uncertainty. Each traversal cost regressor $\mathcal{R}$ is learned from the learning set $\mathcal{L}$ of the paired terrain descriptors and the respective traversal costs observed when using the particular gait g that are depicted in Fig. 5E and Fig.5F, respectively. The particular learned cost regressor $\mathcal{R}$ is used to predict the normal distribution of the traversal cost at queried terrain descriptor ta as

$$\mathcal{N}(\hat{\mu}_c, \hat{\sigma}_c^2)(\mathrm{ta}, \mathcal{R}) = \mathrm{predict}(\mathrm{ta}, \mathcal{R}). \tag{14}$$

The cost prediction (visualized in Fig. 5G) is the expected value

$$\hat{c}(\mathrm{ta}, \mathcal{R}) = E(\mathcal{N}(\hat{\mu}_c, \hat{\sigma}_c^2)(\mathrm{ta}, \mathcal{R})) = \hat{\mu}_c(\mathrm{ta}, \mathcal{R}), \tag{15}$$

and the uncertainty of the prediction (shown in Fig. 5H) is characterized by the differential entropy

$$H(\mathcal{N}(\hat{\mu}_c, \hat{\sigma}_c^2)(\mathrm{ta}, \mathcal{R})) = \frac{1}{2} \log(2\pi e \hat{\sigma}_c^2(\mathrm{ta}, \mathcal{R})). \tag{16}$$

The prediction uncertainty is used to approximate the information gain $I_{\mathcal{C}}$ associated with sampling the individual observed terrains, thus identifying areas the robot needs to visit to improve the traversal cost model.

The terrain type clustering $\mathcal{T}$ identifies the distinct terrain types (terrain descriptor clusters) in the environment. The terrain class set $\mathcal{T}$ is designed to be disjoint regarding the prediction model. Thus, sampling the traversal cost model at a cell corresponding to one terrain class provides no, or severely limited, information regarding the traversal cost model at a location corresponding to a different class. In particular, following Pasolli and Melgani (2011), the classes are selected to be mutually distant in the terrain descriptor space. Each observed cell is assigned the closest terrain class as the closest class in the descriptor space

$$T^*(\nu) = \mathrm{argmin}_{T \in \mathcal{T}} \| \mathrm{ta}(\nu), \mathrm{ta}(T) \|, \tag{17}$$

where $\mathrm{ta}(T)$ is the appearance assigned to the terrain class $T \in \mathcal{T}$. Since, on small terrain classes, it might not be possible to acquire enough samples to learn the traversal cost with sufficient certainty, we apply class erosion as described in Appendix 1. The erosion output is the learning class assignment T and the planning class assignment $\hat{T}$. We avoid computing the cost prediction for every cell independently[2] , and report the $\mathcal{C}^g$ prediction over a particular area as the cost to traverse over its respective terrain type

$$\hat{c}_{\mathcal{C}^g}(\nu) = \begin{cases} \hat{c}(\mathrm{ta}(\hat{T}(\nu)), \mathcal{R}) & \text{if } \hat{T}(\nu) \neq \varnothing, \\ c_{\max} & \text{otherwise}, \end{cases} \tag{18}$$

where the maximum cost $c_{\max}$ is reported for cells with no class (i.e., eroded) $\varnothing$.

The rest of this section describes how the traversal cost experience used to learn the models is measured, how the GP regressor is learned, and how the terrain type clustering is used to identify the locations where to improve the cost model.

4.3.1 Traversal Cost Measurement

The measured traversal cost describes the time needed to traverse between cells as $z^c(\nu, \nu')$. Since the distance between two cells is significantly lower than the robot stride length, the cost is smoothed over path segments (cell sequences) with a fixed duration. In particular, the per-meter cost c is continually measured as the inverted robot velocity v^{-1} over the path segment traversed by the robot in the last Δt s

$$v^{-1}(\psi_s) = \frac{T(\psi_s)}{\|\psi_s\|}, \tag{19}$$

where $\|\psi_s\|$ is the length of the segment in meters and $T(\psi)$ is the measurement duration that is fixed to Δt. If the robot had not changed its gait on the segment, the cost is reported to the particular model $\mathcal{C}^g$ as the cost to traverse the midpoint of the segment as $z^c(\nu_{\lfloor|\psi_s|/2\rfloor}, \nu_{\lfloor|\psi_s|/2\rfloor+1})$. Besides, to remove potential cost spikes, the cost is further smoothed using a moving average window of the same (Δt) duration. Since the inverse velocity is unbounded and has both high values and high variance for a stuck robot, the cost to be used by the predictor is transformed as

$$c = c_{\mathrm{high}} \tanh\left(\frac{1}{c_{\mathrm{high}}} \frac{v^{-1}}{v_{\max}^{-1}}\right), \tag{20}$$

where the maximum robot velocity $v_{\max}$ (maximum from all $g \in \mathbb{G}$) scales the cost of the robot moving over an ideal terrain to 1, and the high cost c_{high}, which should only be experienced by a stuck robot, is used in the transform to bound the cost values.

4.3.2 Gaussian Process Traversal Cost Regressor

The employed GP regressor predicts both the prediction mean and variance making it suitable to model the prediction distribution as in (14). Its description is dedicated to Appendix 2 to make the paper self-contained. GP regressor is learned only if there are at least $n_{\mathcal{L}}^{\min}$ learning pairs in $\mathcal{L}$ to avoid learning overconfident predictors at the beginning of the exploration. The learning is summarized in Alg. 5.

[2] In practice, for small environments, it is feasible to compute the prediction for every cell, and we do so for visualization as depicted in Fig. 5G and Fig. 5H.

Algorithm 5: Traversal Cost Model Learning

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; $z^c_{1,\ldots,n}$ – Cost measurements.
Output: $\mathcal{L}$ – Learning set; $\mathcal{R}$ – Regressor; $\mathcal{M}_{2.5D}$ – Elevation grid map with measured cost assignments.

while exploration is running **do**
  getLatest($\mathcal{M}_{2.5D}$)
  $\mathcal{M}_{2.5D} \leftarrow$ insertIfNovel($\mathcal{M}_{2.5D}, z^c_i$) // Save novel cost measurements to grid map.
  $\mathcal{L} \leftarrow \varnothing$ // Initialize learning set.
  for $\nu \in \mathcal{M}_{2.5D} : \exists c(\nu), \exists\, \mathrm{ta}(\nu)$ **do** // For each described grid map cell with measured cost.
    $\mathcal{L} \leftarrow \mathcal{L} \cup (\mathrm{ta}(\nu), c(\nu))$ // Add the cell to the learning set.
  if $|\mathcal{L}| \geq n^{min}_{\mathcal{L}}$ **then** // If the learning set is large enough.
    $\mathcal{R} \leftarrow \mathrm{learn}(\mathcal{L})$ // Learn the GP regressor.
    reportLatest($\mathcal{L}, \mathcal{R}, \mathcal{M}_{2.5D}$)

The covariance function used in this work is the squared exponential kernel

$$K(x, x') = \sigma_s^2 \exp\left(-\frac{1}{2}\frac{(x - x')^2}{l^2}\right), \tag{21}$$

where σ_s^2 is the output variance, and l is the lengthscale. We consider that the robot's cost and feature models have known ranges based on (20) and the histogram descriptor, respectively. Therefore, similarly to Karolj et al. (2020), the kernel hyperparameters l and σ_s^2, and GP's σ_ϵ^2 have fixed values that we consider to be dependent on the system parameters.

The GP is continually relearned when new observations using the particular gait g are experienced. The learning complexity can be bounded by $\mathcal{O}(n^4)$, where n is the number of training points. The size of the learning set $\mathcal{L}$ is limited by using at most one training point corresponding to each cell in $\mathcal{M}_{2.5D}$, and by storing measurements only when they are novel and thus likely to improve the model. Hence, the relative traversal cost $c(\nu)$ experienced at cell ν is paired with the appearance descriptor $\mathrm{ta}(\nu)$ of the respective traversed terrain, and when building the learning set $\mathcal{L}$, the model reports the pair $(\mathrm{ta}(\nu), c(\nu))$ for each cell where both values are available.

Since the robot keeps only one measurement for each cell, each novel cost measurement $z^c(\nu, \nu')$ experienced when using the gait g is allocated to the grid map cell ν and its neighbors in $8\mathrm{nb}(\nu)$, and the traversal cost $c(\nu)$ at the cell ν is modeled using Kalman filter with the estimated value and covariance as

$$c_k = \frac{\sigma^2_{\text{sense}} c_{k-1} + \sigma^2_{k-1} z^c_k}{\sigma^2_{\text{sense}} + \sigma^2_{k-1}}, \quad \sigma^2_k = \frac{\sigma^2_{\text{sense}}\sigma^2_{k-1}}{\sigma^2_{\text{sense}} + \sigma^2_{k-1}}, \tag{22}$$

where z^c_k is the k-th cost measurement at ν and σ^2_{sense} is its variance. The filter is initialized by the first cost observation z^c_0 at the respective cell, and the initial filter variance is σ^2_0.

Two cases are considered as situations when the cost is novel, and thus the model should be improved by storing the cost w.r.t. (22): (i) when the prediction is erroneous; and (ii) when the prediction is uncertain. For the former, the cost experienced at the cell ν is accumulated if the measured cost z^c is out of the approximate $95\,\%$ confidence interval $|\hat{\mu}_c(\mathrm{ta}(\nu)) - z^c| > 2\hat{\sigma}_c(\mathrm{ta}(\nu))$ of the prediction at ν. For the latter, the approximated information gain of the prediction is considered, and the robot accrues measurements when there is a potential of information gain $I_{\mathcal{C}}(T(\nu)) > 0$, which computation is described in the following paragraphs.

4.3.3 Terrain Type Clustering and Goal Identification

The traversal cost exploration goals $\Gamma^g_{\mathcal{C}}$ are selected by the robot as areas where the model can be improved and thus are the areas where the traversal cost model is uncertain. Each goal represents a terrain class where the robot can sample novel information about the cost model. The overall approach to goal identification is summarized in Alg. 6.

Algorithm 6: Terrain Type Clustering, Goal Identification, and Cost Identification

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; $\mathcal{R}$ – Regressor; $\mathcal{L}$ – Learning set.
Output: $\mathcal{M}_{2.5D}$ – Elevation grid map with cost assignments $\Gamma_{\mathcal{C}}$ – Cost goals; $\mathcal{T}$ – Terrain classes.

- $\mathcal{T} \leftarrow \varnothing$ // Init. terrain class set.
- $\Gamma_{\mathcal{C}} \leftarrow \varnothing$ // Init. goal set.
- **while** exploration is running **do**
  - `getLatest` $(\mathcal{M}_{2.5D}, \mathcal{R}, \mathcal{L})$
  - **if** $\mathcal{R} \neq \varnothing$ **then**
    - $\mathcal{M}_{2.5D}, \mathcal{T} \leftarrow$ `cluster` $(\mathcal{M}_{2.5D}, \mathcal{T})$ // Update terrain clusters (Alg. 7).
    - $\mathcal{T} \leftarrow$ `computeInformationGain` $(\mathcal{R}, \mathcal{L}, \mathcal{M}_{2.5D}, \mathcal{T})$ // Compute information gain (Alg. 8).
    - $\Gamma_{\mathcal{C}} \leftarrow$ `identifyGoals` $(\mathcal{M}_{2.5D}, \mathcal{T})$ // Identify goals (Alg. 9).
    - $\mathcal{M}_{2.5D} \leftarrow$ `setPlanningCost` $(\mathcal{M}_{2.5D}, \mathcal{T})$ // Identify costs (Alg. 10).
    - `reportLatest` $(\mathcal{M}_{2.5D}, \Gamma_{\mathcal{C}}, \mathcal{T})$ // Report costs assigned to $\mathcal{M}_{2.5D}$, goals, and class set.

Algorithm 7: Cluster

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; $\mathcal{T}$ – Terrain classes.
Output: $\mathcal{M}_{2.5D}$ – Elevation grid map with class assignments; $\mathcal{T}$ – Terrain classes.

- **Procedure** `cluster` $(\mathcal{M}_{2.5D}, \mathcal{T})$
  - $A \leftarrow \varnothing$ // Init. the adaptation dataset.
  - **for** $\nu \in \mathcal{M}_{2.5D} : \exists \mathrm{ta}(\nu)$ **do** // For each described cell on the grid map.
    - $A \leftarrow A \cup \mathrm{ta}(\nu)$ // Add the descriptor to the adaptation set.
  - **for** $\mathrm{ta} \in \mathrm{draw}(A, n^{IGNG})$ **do** // For a randomly drawn subset of the adaptation set.
    - $\mathcal{T} \leftarrow$ `adaptIGNG` $(\mathcal{T}, \mathrm{ta})$ // Adapt the IGNG (Alg. 11).
  - **for** $\nu \in \mathcal{M}_{2.5D} : \exists \mathrm{ta}(\nu)$ **do** // For each described cell on the grid map.
    - $T^*(\nu) \leftarrow \mathrm{argmin}_{T \in \mathcal{T}} \| \mathrm{ta}(\nu), \mathrm{ta}(T) \|$ // Assign its terrain type.
  - $\mathcal{M}_{2.5D} \leftarrow \mathrm{erode}(\mathcal{M}_{2.5D})$ // Erode the classes over the grid map.
  - **return** $\mathcal{M}_{2.5D}, \mathcal{T}$

The clustering scheme presented in Alg. 7 is based on the IGNG, described in Appendix 3 to make the paper self-contained. In the neural gas, each neuron is a terrain prototype $\mathrm{ta}(T)$ in the descriptor space that represents a terrain class T. When separating the classes, the intuition is that for exponential kernels, the length scale describes the range from the data where the model can reliably extrapolate, as used, e.g., in (Karolj et al., 2020). Hence, new classes are inserted into the neural gas when the distance from all prototypes exceeds $\sigma^{\mathrm{IGNG}} = 2l$. The neural gas is constructed incrementally by repeated adaptation using the appearance descriptors in the environment, where the size of each adaptation batch is limited to n^{IGNG}

descriptors that are randomly sampled from all the descriptors, and the yielded terrain classes can be seen in Fig. 5I.

Algorithm 8: Compute Information Gain

Input: $\mathcal{R}$ – Regressor; $\mathcal{L}$ – Learning set; $\mathcal{M}_{2.5D}$ – Elevation grid map; $\mathcal{T}$ – Terrain classes.
Output: $\mathcal{T}$ – Terrain classes with information gain assignments.

Procedure `computeInformationGain(`$\mathcal{R}$, $\mathcal{L}$, $\mathcal{M}_{2.5D}$, $\mathcal{T}$`)`
- $H_{\mathcal{C}}^{\mathrm{GT}} \leftarrow -\infty$ // Initialize the experienced-terrain uncertainty threshold.
- **for** $T' \in \mathcal{T} : \exists \nu \in \mathcal{M}_{2.5D}, T(\nu) = T'$ **do** // For each terrain class represented on the eroded grid.
  - **if** $|T' \cap \mathcal{L}| > m_T$ **then** // If the class has enough ground truth measurements.
    - $H_{\mathcal{C}}^{\mathrm{GT}} \leftarrow \max(H_{\mathcal{C}}^{\mathrm{GT}}, H(\hat{\sigma}_c^2(\mathrm{ta}(T'))))$ // Adjust the experienced-terrain uncertainty threshold.
- **for** $T' \in \mathcal{T} : \exists \nu \in \mathcal{M}_{2.5D}, T(\nu) = T'$ **do** // For each terrain class represented on the eroded grid.
  - $I_{\mathcal{C}}(T') \leftarrow \max(H(\hat{\sigma}_c^2(\mathrm{ta}(T'))) - H_{\mathcal{C}}^{\mathrm{GT}}, 0)$ // Compute the information gain.
- **return** $\mathcal{T}$ // Return the terrain classes with assigned information gains.

The terrain classes for which the cost model can be improved are identified using the cost regressor $\mathcal{R}$-predicted traversal cost distribution $\mathcal{N}(\hat{\mu}_c, \hat{\sigma}_c^2)(\mathrm{ta}(T))$ at the class prototypes $\mathrm{ta}(T)$. The traversal cost exploration goals are selected according to Alg. 8 as the classes where there is potential for information gain; see the visualization in Fig. 5J. The gain is approximated from the prediction entropy

$$I_{\mathcal{C}}(T) \approx \max(H(\hat{\sigma}_c^2(\mathrm{ta}(T))) - H_{\mathcal{C}}^{\mathrm{GT}}(\mathcal{L}), 0), \tag{23}$$

where $H_{\mathcal{C}}^{\mathrm{GT}}$ is a threshold value associated with the uncertainty of the experienced traversal costs. The robot learns when there is potential of information gain $I_{\mathcal{C}} > 0$, and no information can be gained at eroded cells $I_{\mathcal{C}}(\varnothing) = 0$. We set the threshold value based on the highest prediction uncertainty for terrains that are considered certain since they cover cells that are already in the learning set as

$$H_{\mathcal{C}}^{\mathrm{GT}}(\mathcal{L}) = \max_{T \in \mathcal{T} : |\{\nu \in \mathcal{M}_{2.5D} : T(\nu) = T\} \cap \mathcal{L}| > m_T} H(\hat{\sigma}_c^2(\mathrm{ta}(T))), \tag{24}$$

where we avoid overconfident GP-predictions for barely sampled terrains by allowing only terrain classes with at least m_T observed ground truth cost values. The threshold equals the maximum value over such ground truth terrain classes.

Algorithm 9: Identify Goals

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; $\mathcal{T}$ – Terrain classes.
Output: $\Gamma_{\mathcal{C}}$ – Cost model goals.

Procedure `identifyGoals` ($\mathcal{M}_{2.5D}$, $\mathcal{T}$)
  for $T \in \mathcal{T} : I_{\mathcal{C}}(T) > 0$ **do** // For each terrain class where information can be gained.
    $\gamma_{\mathcal{C}}(T) \leftarrow \varnothing$ // Initialize the sampling site set.
  for $\nu \in S$ **do** // For each cell on the sampling lattice.
    if $\exists \nu' \in \mathcal{M}_{2.5D} : \|\nu, \nu'\| < \frac{\sqrt{2}}{2} d_S, I_{\mathcal{C}}(T(\nu')) > 0, c(\nu') = \varnothing$ **then** // If there is a close enough cell that has non-zero inforation gain and no measured cost.
      $\nu'' \leftarrow \operatorname{argmin}_{\nu' \in \mathcal{M}_{2.5D} : \|\nu,\nu'\| < \frac{\sqrt{2}}{2} d_S, I_{\mathcal{C}}(T(\nu')) > 0, c(\nu') = \varnothing} \|\nu, \nu'\|$ // Find the closest such cell.
      $\gamma_{\mathcal{C}}(T(\nu'')) \leftarrow \gamma_{\mathcal{C}}(T(\nu'')) \cup \nu''$ // And add it to the respective goal as a sampling site.
  for $T \in \mathcal{T} : I_{\mathcal{C}}(T) > 0, |\gamma_{\mathcal{C}}(T)| = 0$ **do** // For each terrain class with information gain but no sampling cell.
    $\mathcal{T} \leftarrow \mathcal{T} / T$ // Prune the terrain class.
  return $\cup_{T \in \mathcal{T} : I_{\mathcal{C}}(T) > 0} \gamma_{\mathcal{C}}(T)$ // Return the goal set.

The sampling locations (visualized, for example, in Fig. 5J) corresponding to the terrain class are sampled along a lattice S with the cellsize $d_S >> d_\nu$ as depicted in Alg. 9. For each lattice point p_S, the closest cell ν in $\delta(\frac{\sqrt{2} d_S}{2}, p_S)$ radius that is not associated with a traverability measurement and that is informative with $I_{\mathcal{C}}(T(\nu)) > 0$ is reported as a sampling site; if no such cell exists, no site is reported for the lattice point. Since only cells without measurements are considered, it is possible for small terrain classes to run out of cells before reaching m_T measurements. In such a case, the class is considered too small to learn and is no longer reported as a goal, and it is pruned from the class set. Beside the goals, the traversal cost $\hat{c}_{\mathcal{C}^g}(\nu)$ (visualized in Fig. 5K) is also reported for the ν's prototype $\mathrm{ta}(\hat{T}(\nu))$ w.r.t. (13) according to Alg. 10.

Algorithm 10: Set Planning Cost

Input: $\mathcal{M}_{2.5D}$ – Elevation grid map; $\mathcal{T}$ – Terrain classes.
Output: $\mathcal{M}_{2.5D}$ – Elevation grid map with cost assignments.

Procedure `setPlanningCost` ($\mathcal{M}_{2.5D}$, $\mathcal{T}$)
  $\mathcal{M}_{2.5D} \leftarrow \mathrm{dilate}(\mathcal{M}_{2.5D})$ // Dilate the classes over the grid map.
  for $\nu \in \mathcal{M}_{2.5D}$ **do** // For each cell.
    if $\hat{T}(\nu) \neq \varnothing$ **then** // If the cell has a dilated class.
      $\hat{c}_{\mathcal{C}^g}(\nu) \leftarrow \hat{c}(\hat{T}(\nu))$ // Report the class cost.
    else // Otherwise.
      $\hat{c}_{\mathcal{C}^g}(\nu) \leftarrow c_{\max}$ // Report the maximum cost.
  return $\mathcal{M}_{2.5D}$ // Return the map with planning cost assignment.

5 EXPERIMENTAL EVALUATION

The proposed exploration with active terrain learning has been examined in simulated trials and real experimental deployments using a hexapod walking robot. The simulated and real scenarios have been set up so that the robot first explores the environment and learns the cost models using the proposed method and, in some tests, a selected baseline method. Then, the performance has been evaluated and compared with the baseline approach by navigating the robot over a sequence of benchmark waypoints using the respective traversal cost models of the environment learned during the exploration.

Table 2. Gait Parametrization

Gait Parameter / Gait	*Fast* Gait	*Tall* Gait
Gait Cycle Duration [s]	1.10	2.90
Step Height [m]	0.04	0.07
Max Forward Speed [ms^{-1}]	0.05	1.25×10^{-2}

The hexapod walking robot, which can be seen in Fig. 1, is used in the real deployment, and the simulation is parameterized to mimic the robot's motion and sensory capabilities. The robot has six legs, each comprising three Dynamixel XM430-W350 servomotors. The robot is equipped with the Intel RealSense D435 camera used to construct the colored environment model and the Intel RealSense T265 localization camera. The onboard computation is provided by the Intel NUC 10i7FNK with Intel Core i7 10710U accompanied with $64\,\mathrm{GB}$ memory, running Ubuntu 18.04 with ROS Melodic (Quigley et al., 2009). The robot locomotion is facilitated by a blind adaptive motion gait (Faigl and Čížek, 2019). The robot uses two particular gait configurations, see Tbl. 2: the *fast* gait suitable for flat, even surfaces, and the *tall* gait that performs better than the *fast* gait over rough terrain but otherwise is slower. The robot is equipped with a reflex that detects that the robot is stuck with costs exceeding c_{max} and switches over to the *tall* for $\Delta t_{\mathrm{fallback}}$ seconds to avoid the robot getting stuck when using the baseline model or at the beginning of the learning process. The parameterization of the proposed method can be found in Tbl. 3, and the operating frequencies of the proposed method's processes are depicted in Tbl. 4.

5.1 Simulated Scenarios

The simulated scenarios are based on a courtyard environment captured by four 3D scans obtained using Leica BLK 360 3D scanner and visualized in Fig. 6A. The scanner has standard deviation of $4\,\mathrm{mm}$ at $10\,\mathrm{m}$, and $7\,\mathrm{mm}$ at $20\,\mathrm{m}$. The scans total approx. 1.4×10^{8} points.

Two virtual environments are created using the scan: *small* and *large*. The *small* environment represents a small section of the courtyard, where the simulated robot mimics the real robot's speed and sensory equipment. It is used to test the benefit of the individual components of the proposed approach by comparing them to baseline methods where the particular component is removed or simplified. The *large* environment comprises terrain segments observed in the scan that are rearranged to create a larger, artificial environment with obstacles where different exploration algorithms are compared using a faster robot with an extended sensor range.

5.1.1 Small Environment

The *small* environment is concerned with a section of the environment that is detailed in Fig. 6B. We have created a simulation model of the environment containing several types of pavement (gray, red) and turf (green, brown) that are shown in Fig. 6C. The turf is modeled as hard to traverse and can get the robot stuck for the *fast* gait, while the pavement does not impede the robot, see Fig. 6D.

First, to demonstrate the benefits of using a cost model learned from prior experience, the robot is tasked to execute two tours in the environment using the learned cost model and a flat-cost baseline model. Second, the utility of exploring along the proposed GTSP-derived path is demonstrated by comparing its

Table 3. System Parametrization

Symbol	Parameter	Unit	Value, Split by Environment	
			Real/Small Sim.	Large Sim.
d_ν	Gridmap cellsize	m	0.05	0.10
r_{sensor}	Sensor range	m	2.5	10.00
c_{radius}	Spatial clustering radius	m	0.50	2.00
c_{min_cells}	Spatial clustering, min cells per cluster	-	10	10
r_{robot}	Robot footprint radius	m	0.25	0.40
ρ_{obstacle}	Roughness passability threshold	m	0.25	0.25
r_{hist}	Histogram descriptor radius	m	0.25	0.30
Δt	Cost measurement window duration	s	5.00	1.00
$v_{\max}$	Maximum robot velocity	$\mathrm{m\,s^{-1}}$	0.05	0.25
c_{loss}	Cost distance-transform per-meter loss	–	10.00 / 15.00*	7.5
c_{high}	High cost for cost transform	–	20.00	20.00
$c_{\max}$	Maximum cost for path planning	–	20.00	20.00
σ_{sense}	Kalman filter cost measurement variance	–	0.10	0.10
σ_0^2	Kalman filter initial variance	–	1.00	1.00
σ_s	GP output variance	–	1.00	1.00
σ_ϵ	GP observation noise	–	0.50	0.50
l	GP lengthscale	–	0.40	0.40
$n_{\mathcal{L}}^{\text{min}}$	Minimum learning set size	–	25.00	25.00
$n_{\text{erode}}^{\text{steps}}$	Cluster erosion steps	–	2.00	2.00
m_T	Minimum size of a ground truth cluster	–	10.00	10.00
d_S	Cost-model sampling lattice cell size	m	0.44	0.44
$n_{\text{dilate}}^{\text{steps}}$	Cluster dilation steps	–	3.00	3.00
$n_{\text{dilate}}^{\text{size}}$	Cluster dilation size	–	2.00	2.00
ϵ_1^{IGNG}	GNG warp scale winner	–	1.00×10^{-3}	1.00×10^{-3}
$\epsilon_{\text{nb}}^{\text{IGNG}}$	GNG warp scale neighbor	–	1.00×10^{-5}	1.00×10^{-5}
$a_{\text{mature}}^{\text{IGNG}}$	GNG age mature	–	1.00×10^{2}	1.00×10^{2}
$a_{\text{max}}^{\text{IGNG}}$	GNG max edge age	–	50.00	50.00
n^{IGNG}	GNG learning batch size	–	5.00×10^{3}	5.00×10^{3}
Δt_{sample}	Cost sampling duration	s	30.00	12.00
$\Delta t_{\text{fallback}}$	Stuck fallback duration	s	30.00	3.00

* Different value used in *small* simulation/real deployment.

time to explore the environment with a greedy, myopic baseline, which drives the robot to the cheapest goal to reach w.r.t. to the so far learned costs.

The first tour comprises four waypoints. The robot starts at the bottom-left point and executes the tour counter-clockwise until reaching the start location again. Two particular areas are designed to demonstrate the utility of the learned model: (i) the segment between the bottom-right and top-right waypoints where the robot can choose either a direct route over the turf or a longer path over the pavement; (ii) and the area around the top-left waypoint where the turf cannot be avoided and thus the robot needs to switch to the *tall* gait. The second tour comprises 20 points randomly sampled in the environment, and it serves to demonstrate the performance of the learned model over a tour that was not handcrafted.

Table 4. System Operation Frequencies

Module	Frequency	Condition
Elevation mapping	5.00 Hz	
Spatial goal identification	0.33 Hz	
Cost measurement	20.00 Hz	Only if using the respective gait.
Cost learning	0.10 Hz	Only if not already running.
Goal identification	0.10 Hz	
Goal Sequence Planning	1.00 Hz	Only after goal set change or reaching a goal.
Path Planning	1.00 Hz	Only after goal set change or reaching a goal.

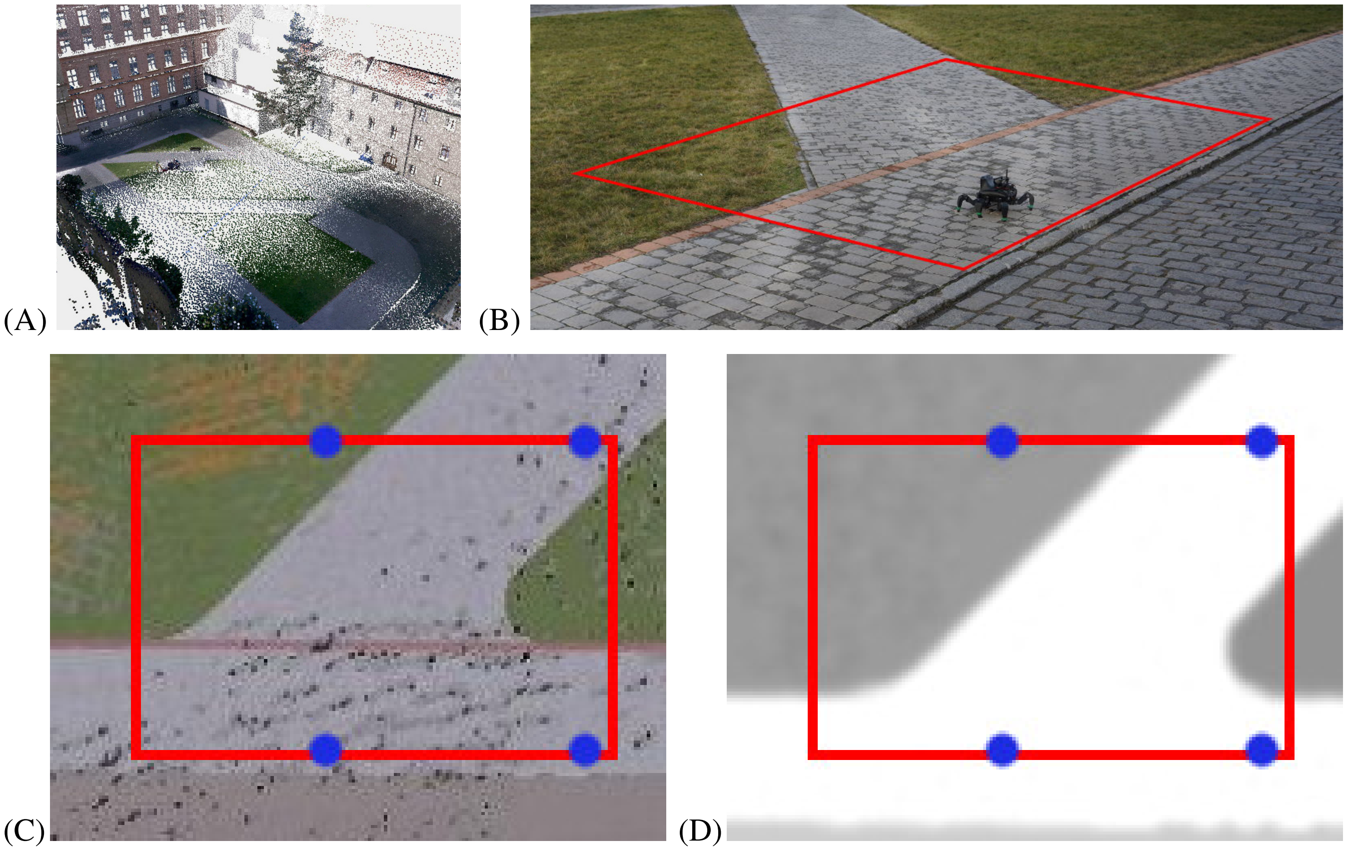


Figure 6. (A) The 3D scan of the university campus at Charles Square in Prague, (B) and the section of the courtyard and the respective simulated environment (C) color and (D) relative traversability (light areas easier to traverse). The red bounding box represents the area where the robot should explore. The blue points are the points to be visited by the robot in the first test tour.

Besides the proposed approach and the baseline, in the simulated tests, we also deploy a hybrid gait selection approach that chooses its gait using the proposed model but does not plan its path w.r.t. the predicted costs and walks directly to the next waypoint. Unlike the baseline approach, which switches to the *tall* gait when stuck and repeatedly tries to switch back to the *fast* gait, the hybrid gait selection approach switches gaits only when approaching or leaving the terrain identified as hard to traverse by the model. Hence, it should outperform the baseline over longer sections on difficult terrains, where the baseline is slowed down by trying to switch back to the *fast* gait.

The simulation environment consists of the Intel i7-9700 $3.00\,\mathrm{GHz}$ with $32\,\mathrm{GB}$ memory running Ubuntu 18.04 with ROS Melodic. Since the captured environment comprises terrains that might slow down the robot because they are somewhat non-rigid, instead of using a geometry-based simulator such as Gazebo, which cannot model such terrains, we elect to build a virtual environment over a simple simulator using real-world data. The simulation is performed using the Simple Two Dimensional Robot Simulator (STDR) [3] within the ROS ecosystem. On top of the simulator, we have implemented an interface that simulates the robot's RGB-D camera, which assigns each point in the robot's simulated exteroceptive measurements color based on the point's position in the environment color map shown in Fig. 6C, and filters the measurements to contain only points within the $87\,\mathrm{deg}$ wide field of view of the simulated RGB-D camera. The terra-mechanical properties are simulated by slowing down the robot over the individual traversed terrains w.r.t. the performance observed over such terrain in a real-world deployment, as shown in Fig. 6D.

In the evaluation, the robot first explores and learns the models shown in Fig. 7A to Fig. 7I. An example exploration path can be seen in Fig. 7J. The robot learns that the turf, which appears either green or brown, cannot be traversed by the *fast* gait and thus selects the *tall* gait over that terrain type. On the other hand, the pavement does not hinder the *fast* gait, which is considerably faster and thus preferred.

Although the two gait models create the terrain clusters independently, the clusters in Fig. 7E and Fig. 7H differ only in cluster indices used in the internal representation (each index is associated with a different color in the visualization). It can be observed that the robot does not use any clusters associated with the red line on the pavement, either removing the thin cluster outright in the erosion or pruning the small erosion remains after the robot finds out that it cannot get enough samples to learn such a small terrain.

In the particular exploration run shown in Fig. 7J, the robot first walks along the left side of the exploration bounds, learning the *fast* gait costs for both the pavement and turf and the *tall* gait cost over the turf. Then, the robot learns the *tall* gait cost over the pavement while clearing the spatial exploration goals. During the exploration, it can be seen that the robot avoids walking over the remaining turf, only approaching it at the very end of the exploration. Thus, the robot needs only to enter and not leave the turf (minimizing the time on the costly terrain) to reach the goal that lies on the turf.

The test runs using the baseline, and the learned model over the first tour are shown in Fig. 7K and Fig. 7L, respectively. Besides, the development of the tours that would be used at different points during the exploration can be seen in Fig. 7M through 7O. In the baseline test, the robot walks directly between the waypoints and only switches to the *tall* gait after getting stuck. On the other hand, when using the learned model, the robot avoids the turf if possible and switches to the *tall* gait before entering the turf while pursuing the top-left goal.

The performance over 25 simulated trials (5 exploration runs, each with 5 tour tests for the tour tests; 25 runs for the simulated exploration tests) can be observed in Tbl. 5. On the first tour, the hybrid gait selection approach is slower than the reactive baseline. In the authors' opinion, it is caused by the conservative (large) value of r_{robot}, which compels the robot to use the slow *tall* gait on the border between the rough terrain and pavement, while the reactive approach only tries to switch back to the *fast* gait (which is its main disadvantage when compared to the hybrid approach) a few times on the short rough terrain segment. Nonetheless, the proposed learned model knows to avoid such areas and performs better or the same as the other approaches over every tour segment. Hence, the results suggest that robot benefits from using the learned costs in path planning. Over the second tour, the robot performs similarly. The learned model outperforms the baseline when moving around or over the turf. Both approaches exhibit similar travel times

[3] http://stdr-simulator-ros-pkg.github.io

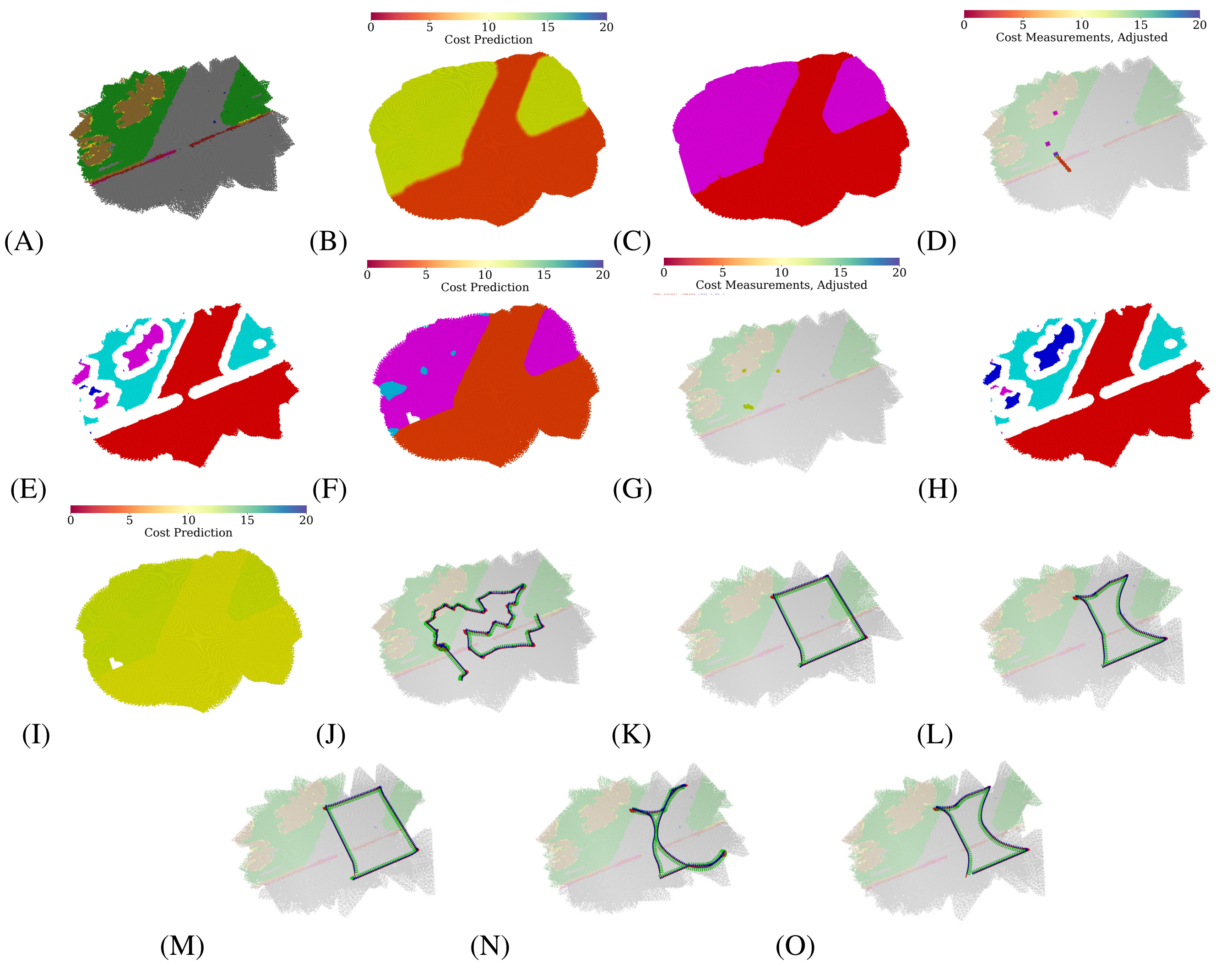


Figure 7. The environment assessment after the simulated scenario run with regards to both gaits; (A) dominant color in the histogram feature; (B) merged cost used for planning; (C) selected gait (*fast* in red, *tall* in purple); (D) costs used for learning the *fast* gait model (adjusted by hyperbolic tangent in (20)), visualized over the terrain appearance; (E) clusters used in the *fast* gait model (arbitrary colors used to distinguish clusters); (F) *fast* gait cost predictions assigned by the dilated clusters. (G) costs used for learning the *tall* gait model (adjusted by hyperbolic tangent using (20)), visualized over the terrain appearance; (H) clusters used in the *tall* gait model (arbitrary colors used to distinguish clusters); (I) *tall* gait cost predictions assigned by the dilated clusters. (J) exploration run; (K) test-tour run using the baseline model without the learned traversal costs; (L) test-tour run using the learned traversal costs. The development of the path through the fully discovered simulated environment during the exploration; (M) at the beginning of the exploration, the robot uses flat costs and thus does not avoid difficult terrains; (N) after learning the costs for the *fast* gait, the robot is too cautious and avoids going near the costly turf; (O) after learning the *tall* gait costs, the robot is less cautious and is willing to walk near difficult terrain.

when the direct path between the waypoint leads only over the pavement. Unlike over the first tour, the hybrid gait selection performs better than the baseline approach, presumably due to longer sections over hard-to-traverse terrains on the second tour. The proposed approach consistently outperforms the baseline and hybrid gait selection approaches; we conclude that the robot benefits from using the learned model.

Besides the tour tests, the results suggest that the robot benefits from using the non-myopic GTSP planner compared to the myopic greedy approach. Even though the performance of the two approaches

Table 5. Performance as the time (total cost) in seconds to traverse

Small Virtual Environment, Tour 1 (mean $\pm$ std of 25 runs)					
Method/Time [s]	Segment 1	Segment 2	Segment 3	Segment 4	Full Tour
Baseline	79.99 ± 0.00	239.59 ± 6.62	133.20 ± 6.76	177.59 ± 13.04	630.39 ± 21.06
Gait selection	80.00 ± 0.00	275.00 ± 8.06	125.49 ± 7.39	164.00 ± 7.39	644.50 ± 7.34
Proposed	80.00 ± 0.00	119.99 ± 0.00	112.40 ± 4.27	142.40 ± 4.27	454.80 ± 4.27

Small Virtual Environment, Tour 2 (25 runs)	
Method/Time [s]	Full Tour
Baseline	2748.00 ± 30.59
Gait selection	2523.12 ± 39.48
Proposed	2271.99 ± 33.38

Small Virtual Env., Exploration (5 runs)	
Environment	**Time** [s]
GTSP	1382.68 ± 241.47
Greedy	1547.16 ± 203.71

Large Virtual Environment (mean $\pm$ std of 5 runs)		
Method	**Full Tour Time** [s]	**Exploration Time** [s]
Proposed	554.00 ± 13.56	1167.15 ± 163.69
Spatial-only	859.99 ± 156.02	545.40 ± 137.43

Real Deployment	
Test	**Time** [s]
Test Segment, Baseline	454.00
Test Segment, Proposed	143.00
Exploration, Proposed*	1364.00

* The similarity between the real and simulated times to explore is coincidental.

appears relatively close, the *Mann-Whitney U Test* (Mann and Whitney, 1947) rejects the null hypothesis of the same exploration time distribution at $99.5\,\%$ confidence against both the two-sided and the relevant one-sided alternative. In the authors' opinion, the high variance in the observed exploration times can be attributed to the effect of random chance in exploration since neither myopic nor non-myopic approaches are informed about the terrains in unexplored areas. However, the myopic explorer is more likely to make a bad decision, such as not clearing some of the goals in a particular area that needs to be visited later. Therefore, the proposed non-myopic approach performs better overall.

5.1.2 Large Environment

The *large* environment is an artificial $20 \times 25\,\mathrm{m}$ outdoor/indoor scenario. The map comprises patches from the courtyard scan rearranged as shown in Fig. 8. Given the size of the environment, the robot is sped up 5 times. The cell size is increased to $0.1\,\mathrm{m}$, and other parameters are adjusted accordingly, see Tbl. 3. Besides, the robot uses an omnidirectional sensor with the increased range of $10\,\mathrm{m}$, which expands the range of terrains that can be observed without the respective terrain's traversal. To accommodate the simulation of the increased sensor range, the virtual environment is run on AMD Ryzen Threadripper 3960X $3.8\,\mathrm{GHz}$ with $48\,\mathrm{GB}$ memory running Ubuntu 18.04 and ROS Melodic, using STDR in the same manner as for the *small* environment.

Similar to the *small* environment, the robot is first set to explore the environment and then is tasked to visit the set of waypoints shown in Fig. 8C. The proposed algorithm is compared to a *spatial-only* baseline approach, which learns the cost models only as a result of experiencing cost while pursuing spatial exploration goals. The *spatial-only* changes the gaits in a reactive fashion when stuck and hence only learns the model for the *tall* gait if it enters the difficult green or brown turf during the exploration.

The quantitative results for the *large* environment are shown in Tbl. 5. Since the proposed approach actively tries to sample every terrain type, it is slower to explore the whole environment. However, the proposed approach performs better in the tour evaluation. Closer examination suggests that while the tour times of the proposed approach remain similar in all trials, the *spatial-only* times vary wildly since

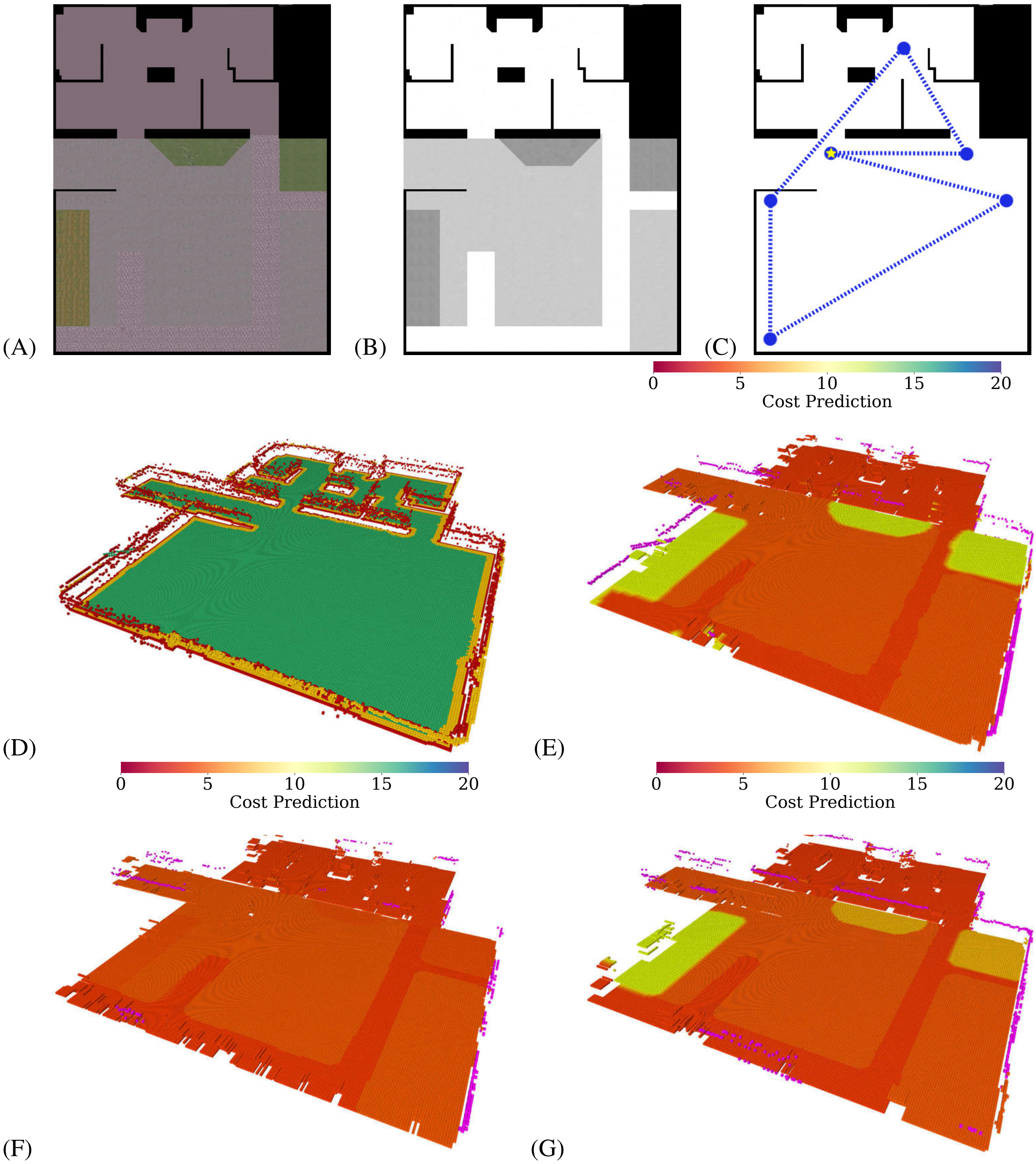


Figure 8. The large simulated environment (A) color and (B) relative traversability, (C) and the test tour through the environment, which starts at the starred node and is counter-clockwise. The built maps of the large simulated environment: (D) geometric map and (E) merged costs used for planning after exploration using the proposed approach; merged costs ofter exploration using the *spatial-only* model while (F) avoiding and (G) traversing rough terrain, respectively.

the learned models differ based on which terrains the robot has traversed during the exploration. This randomness can be attributed to differences in simulation and plan execution. Besides, Fig. 8D-G shows the learned maps for the proposed model, and for the *spatial-only* model in both the cases when the rough terrain was and was not traversed. For the case when a rough terrain was traversed by the *spatial-only* model, the costs differ between the individual rough areas. However, the ground truth costs shown in Fig. 8B suggest that they should be the same, as is the case for the proposed model. Likely, this is caused by the robot traversing only the brown-green rough terrain located on the left of the environment. The green terrain, located in the center and right of the environment, appears somewhat similar to the brown-green terrain. Hence, the robot considers it to be difficult to traverse to a certain degree. However, since the *spatial-only* model does not deliberately sample the terrains, the model's guess differs somewhat from the exact cost to traverse the particular terrain, decreasing the fidelity of the predictions.

Overall, the presented results suggest that the proposed approach presents a tradeoff in terms of exploration and execution time: the longer time spent exploring the environment and learning the cost models provides the robot with better cost maps, which shorten the time to navigate the environment after it is explored. It should be noted that since the behavior of the *spatial-only* model is affected by random chance (differences in simulation and plan execution), it can provide models as good as the proposed approach. However, there is no guarantee that this would happen regularly, while the proposed approach has returned high fidelity maps in every test case.

5.2 Real Robot Experimental Deployment

The viability of the proposed approach is demonstrated in the real experimental deployment, where the robot explores an indoor $2 \times 6\,\mathrm{m}$ area visualized in Fig. 9. The office-like environment comprises flat synthetic terrain that is easy to traverse but appears to the robot differently colored at different locations since it is glossy and carries the color of nearby objects located next to the arena. When building the colored elevation map $\mathcal{M}_{2.5D}$, we use the first color observed at each location to account for the issue. Besides the flat terrain, green artificial turf is placed in a part of the test area to provide a relatively hard terrain to traverse. The robot interacts with the real terrains similarly to the simulation: the *fast* gait may get stuck on the turf but is faster than the *tall* gait over the flat parts of the arena. During the experiment, the robot is set to explore the area; even though it can leave the bounds of the $2\,\mathrm{m} \times 6\,\mathrm{m}$ large area, it does not pursue goals located outside of the bounds.

Fig. 10 shows the maps learned in the experimental run, which is also presented in the accompanying video. A colored map of the environment is depicted in Fig. 10A. The overall costs and selected gaits through the environment are shown in Fig. 10B and Fig. 10C, respectively.

During the experimental deployment, the robot first learns the largest gray appearing flat terrain using the *fast* gait. Then, it learns on the turf for both gaits and returns to the gray area to learn for the *tall* gait. Afterward, the robot pursues the yet unvisited spatial goals and smaller off-color terrain clusters that appear near the environment boundary and are caused by the glossy floor that carries the color of the nearby objects.

Compared to the simulation, the robot needs a larger amount of the measurements to learn the terrains (see Fig. 10D and Fig.10G) and there are more terrain clusters (see Fig. 10E and Fig. 10H). It suggests that the real environment is noisier and contains multiple differently colored areas, which is in line with our observations regarding the glossy floor material. Nevertheless, the traversal costs learned by the robot for the individual gaits (see Fig. 10F and Fig. 10I) are within expectations, as is the overall planning cost depicted in Fig. 10B and gait selection visualized in Fig. 10C.

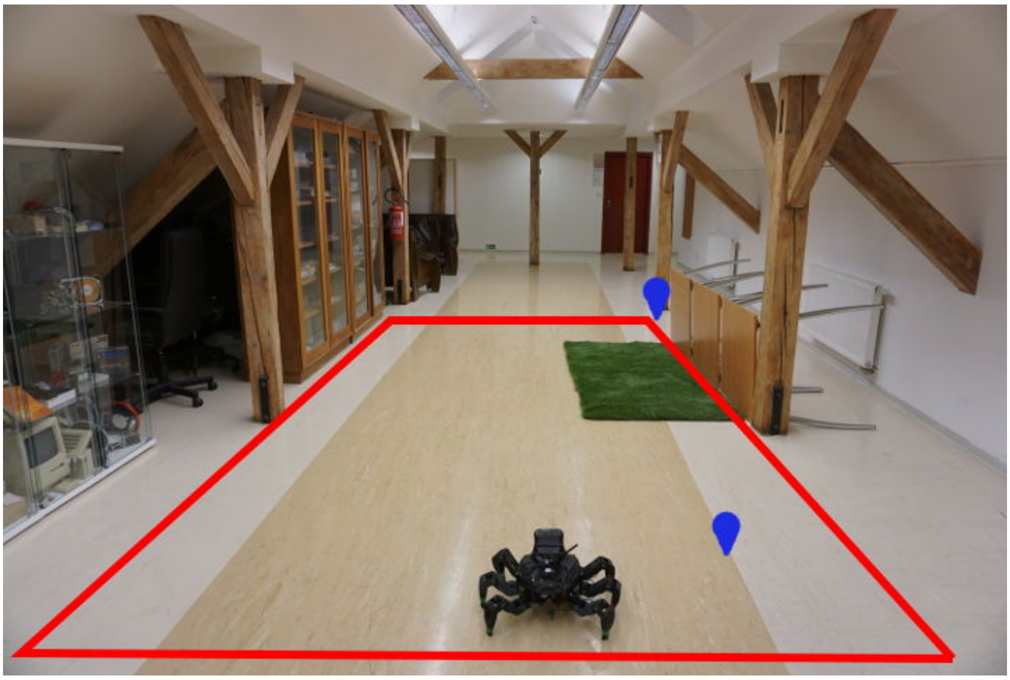


Figure 9. Deployment $2\,\mathrm{m} \times 6\,\mathrm{m}$ large area with a green artificial turf. The area boundary is in red, and the waypoints of the test tour are depicted in blue. The shown robot is at the starting position.

The test run scenarios are set up similarly to the tours used in the simulated test; the robot is placed in front of the hard-to-traverse turf and tasked to reach a goal location behind the hard-to-traverse terrain, slightly out of the exploration bounds, see Fig. 9. The paths shown in Fig. 10K and Fig. 10L show that when using the baseline without the learned model, the robot tries to reach the goal directly over the turf, gets stuck, and needs to switch to the slow *tall* gait. On the other hand, when using the learned model, the robot avoids the hard-to-traverse areas and reaches its goal quickly using the *fast* gait. The performance in the presented run can be seen in Tbl. 5. Overall, we conclude that the real deployment confirms that the robot can actively learn the traversability as a part of the exploration mission and benefits from using such learned models.

6 DISCUSSION

The presented exploration system is proposed as a combination of spatial geometric modeling and learning terrain-gait traversal cost models. However, the system is designed to support additional models that do not describe the robot's traversal cost. Moreover, since the models are kept separate, there is no need to use the same feature set for each of them. Therefore, the approach is compatible with spatial models such as magnetism models (Karolj et al., 2020) or GP-based occupancy (Wang and Englot, 2016). The only requirement for a model is that it produces a set of learning goals in the environment that are resolved once particular information is sampled. Hence, the proposed system can be extended by including additional traversability models, such as modeling the passability of potentially non-rigid obstacles.

Besides, we approach the traversal cost prediction so that it supports any cost model that is additive along the traversed path, such as time to traverse or consumed energy. In addition, individual cost predictors describe the gaits of a hexapod walking robot, but they can also describe any discrete set of robot configurations. Hence, the approach is viable for any mobile robot that describes its motion experience using an additive cost and can also be used to model the energy a tracked robot consumes, e.g., with

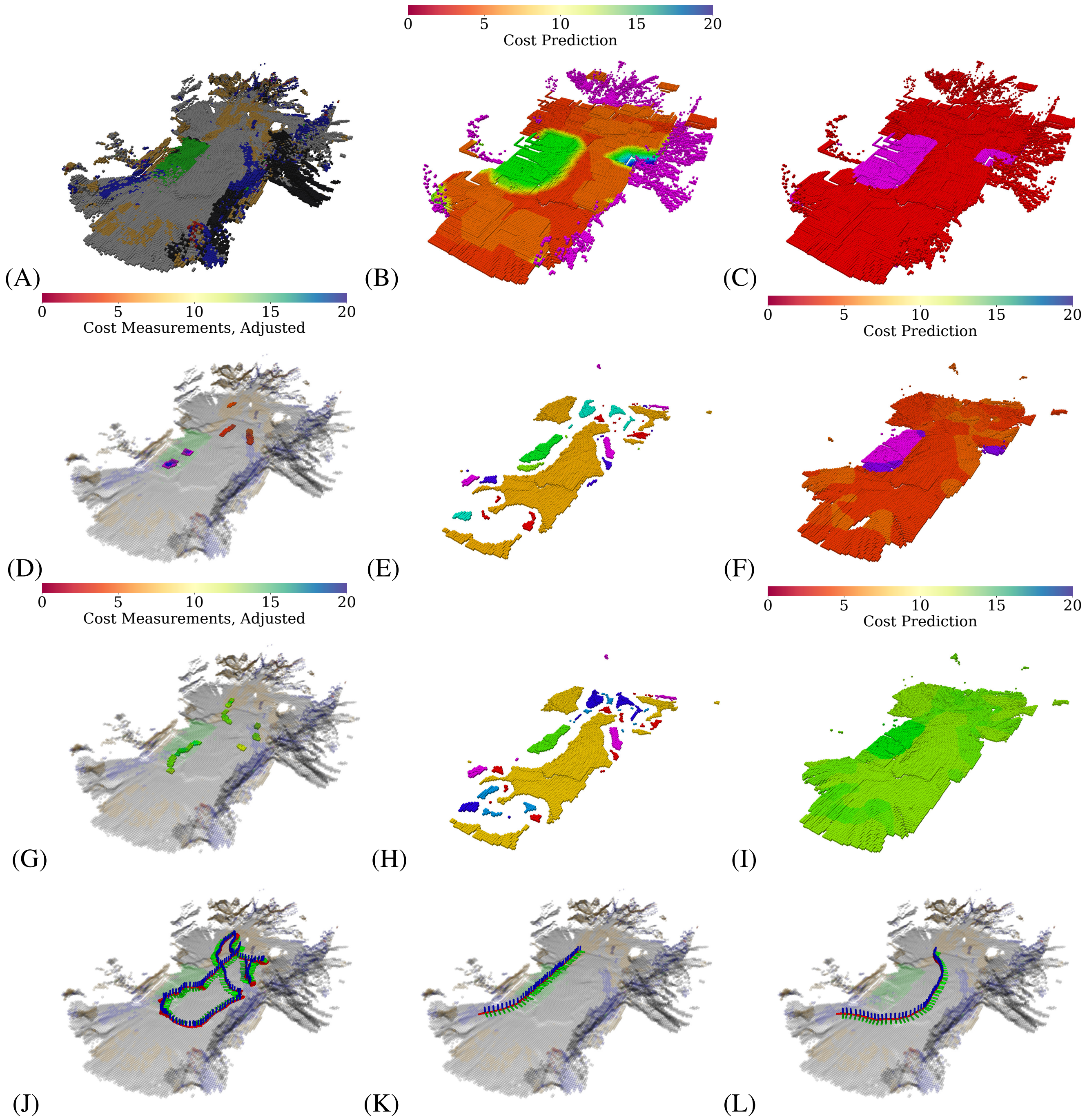


Figure 10. The environment evaluation and the real robot exploration run; (A) the dominant color in the histogram feature; (B) merged cost used for planning; (C) selected gait (*fast* in red, *tall* in purple); (D) costs used for learning the *fast* gait model (adjusted by hyperbolic tangents), visualized over the terrain appearance; (E) clusters used in the *fast* gait model (arbitrary colors used to distinguish clusters); (F) *fast* gait cost predictions assigned by the dilated clusters. (G) costs used for learning the *tall* gait model (adjusted by hyperbolic tangents), visualized over the terrain appearance; (H) clusters used in the *tall* gait model (arbitrary colors used to distinguish clusters); (I) *tall* gait cost predictions assigned by the dilated clusters. (J) exploration run; (K) test-tour run using the baseline model without the learned traversal costs; (L) test-tour run using the learned traversal costs.

adjustable flippers. A particular limitation of the cost modeling used in the presented approach is that we assume that the individual gaits are switched for free w.r.t. the cost (i.e., instantaneously for cost modeled as the time to traverse), while in practice, the gait requires some time to exhibit its properties. In this paper, we leave the question of how to predict gait-change cost open for future work.

The used cost model goal generation stems from the idea that adding new observations does not increase GP uncertainty if the hyper-parameters are fixed (Rasmussen and Williams, 2006). Therefore, sampling new measurements should not increase uncertainty and thus not spawn new goals in areas containing none. In practice, even though we use fixed GP hyper-parameters, the non-increasing nature of the uncertainty does not strictly hold for the approximated information gain since, in addition to the GP hyper-parameters, the information gain also depends on the terrain clusters and the costs and descriptors in the learning set, all of which might drift during the exploration. However, the robot behavior demonstrated in both evaluation setups shown in Fig. 7J and Fig. 10J suggests that the assumption holds in general. The robot clears the areas corresponding to the individual terrains (goals) and is not compelled to return to previously visited areas.

The primary limitation of the proposed approach is identified in its inability to compare the utility of the goals originating from the different models. We are motivated to build a modular system that would support different model types; therefore, the proposed decoupled approach considers each goal equally valued, regardless of the source model. This limits how the models are used since the goal utility, such as the information gain, is relegated to be used only inside the particular model to determine which environment features (locations or terrain types) are goals to use in creating an instance of the GTSP. The proposed approach provides a non-myopic solution to visit the goals reported by the individual models, where the models are also non-myopic since each can report multiple goals. Myopic models that would report their respective highest utility goal (potentially with multiple sampling sites) can be used. However, similarly to the myopic planner with the results reported in Tbl. 5, the time to explore would likely increase since the GTSP planner would lack the information on where to go after the current goals are sampled, and thus the exploration path would often change significantly. Integrating goal utility into the decoupled planning and using alternative utility functions such as the GP-UCB remains the subject of future work.

7 CONCLUSION

In this paper, we present a system for autonomous mobile robot exploration that incorporates active learning of traversal cost models in addition to spatial model building. During the exploration, the robot builds the spatial geometric model of the environment and learns the traversal cost models, each comprising a Gaussian Process regressor and a Growing Neural Gas terrain clustering scheme. The geometric model is used to determine areas passable by the robot, while the cost models predict the traversal costs over the passable terrains from the terrain's appearance. Each cost model corresponds to a particular hexapod walking robot locomotion gait. The robot approaches exploration in a decoupled manner, creating a set of goals for the spatial exploration and for each traversal cost model. The exploration path is planned by solving an instance of the Generalized Traveling Salesman Problem over the goals that are sets of possible sites of visits to improve the particular model. The proposed system has been evaluated in simulation setup and real experimental deployment with two different walking gaits. The results suggest that the proposed system yields the robot to explore the environment and learn the traversal cost models. The learned models benefit the robot's operation in the environment. In future work, we plan to model the gait change costs, include additional traversability models such as obstacle rigidity, and extend the proposed approach to support goal utility and exploration-exploitation models.

CONFLICT OF INTEREST STATEMENT

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

AUTHOR CONTRIBUTIONS

With the support of JF, MP and JB designed the proposed system. MP and JB performed the experiments and processed the data. MP, JB and JF wrote the manuscript. All the authors contributed to the manuscript and approved the submitted version.

FUNDING

The work was supported by the Czech Science Foundation (GAČR) under research project No. 18-18858S and 19-20238S. The support under the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" is also gratefully acknowledged.

ACKNOWLEDGMENTS

We would like to thank Petr Čížek and Jiří Kubík for their help with the hexapod walking robot maintenance.

SUPPLEMENTAL DATA

Supplementary Video 1. A commented exploration run using the hexapod walking robot.

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

REFERENCES

Azpúrna, H., Campos, M. F. M., and Macharet, D. G. (2021). Three-dimensional Terrain Aware Autonomous Exploration for Subterranean and Confined Spaces. In *IEEE International Conference on Robotics and Automation (ICRA)*. 2443–2449. doi:10.1109/ICRA48506.2021.9561099

Baleia, J., Santana, P., and Barata, J. (2015). On Exploiting Haptic Cues for Self-Supervised Learning of Depth-Based Robot Navigation Affordances. *Journal of Intelligent & Robotic Systems* 80, 455–474. doi:10.1007/s10846-015-0184-4

Bayer, J. and Faigl, J. (2019). On Autonomous Spatial Exploration with Small Hexapod Walking Robot using Tracking Camera Intel RealSense T265. In *2019 European Conference on Mobile Robots (ECMR)*. 1–6. doi:10.1109/ECMR.2019.8870968

Bayer, J. and Faigl, J. (2020). Speeded up elevation map for exploration of large-scale subterranean environments. In *2020 Modelling and Simulation for Autonomous Systems (MESAS)*. 190–202

Bayer, J. and Faigl, J. (2021). Decentralized topological mapping for multi-robot autonomous exploration under low-bandwidth communication. In *European Conference on Mobile Robots (ECMR)*. 1–7

Bekhti, M. A. and Kobayashi, Y. (2016). Prediction of Vibrations as a Measure of Terrain Traversability in Outdoor Structured and Natural Environments. In *Image and Video Technology*. 282–294. doi:10.1007/978-3-319-29451-3_23

Belter, D., Wietrzykowski, J., and Skrzypczyński, P. (2019). Employing Natural Terrain Semantics in Motion Planning for a Multi-Legged Robot. *Journal of Intelligent & Robotic Systems* 93, 723–743. doi:10.1007/s10846-018-0865-x

Binney, J. and Sukhatme, G. S. (2012). Branch and bound for informative path planning. In *IEEE International Conference on Robotics and Automation (ICRA)*. 2147–2154. doi:10.1109/ICRA.2012.6224902

Bourgault, F., Makarenko, A. A., Williams, S. B., Grocholsky, B., and Durrant-Whyte, H. F. (2002). Information based adaptive robotic exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 540–545. doi:10.1109/IRDS.2002.1041446

Bradley, D. M., Chang, J. K., Silver, D., Powers, M., Herman, H., Rander, P., et al. (2015). Scene Understanding for a High-mobility Walking Robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1144–1151. doi:10.1109/IROS.2015.7353514

Brown, D. and Webster, G. (2010). Now a stationary research platform, NASA's Mars rover Spirit starts a new chapter in red planet scientific studies. *NASA Press Release*

Carrillo, H., Dames, P., Kumar, V., and Castellanos, J. A. (2018). Autonomous robotic exploration using a utility function based on Rényi's general theory of entropy. *Autonomous Robots* 42, 235–256. doi:10.1007/s10514-017-9662-9

Charrow, B., Liu, S., Kumar, V., and Michael, N. (2015). Information-theoretic mapping using Cauchy-Schwarz Quadratic Mutual Information. In *IEEE International Conference on Robotics and Automation (ICRA)*. 4791–4798. doi:10.1109/ICRA.2015.7139865

Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., and Hutter, M. (2020). Graph-based subterranean exploration path planning using aerial and legged robots. *Journal of Field Robotics* 37, 1363–1388. doi:https://doi.org/10.1002/rob.21993

Faigl, J. and Kulich, M. (2013). On determination of goal candidates in frontier-based multi-robot exploration. In *European Conference on Mobile Robots (ECMR)*. 210–215. doi:10.1109/ECMR.2013.6698844

Faigl, J., Kulich, M., and Přeučil, L. (2012). Goal assignment using distance cost in multi-robot exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 3741–3746. doi:10.1109/IROS.2012.6385660

Faigl, J. and Čížek, P. (2019). Adaptive locomotion control of hexapod walking robot for traversing rough terrains with position feedback only. *Robotics and Autonomous Systems* 116, 136–147. doi:10.1016/j.robot.2019.03.008

Fankhauser, P., Bloesch, M., Gehring, C., Hutter, M., and Siegwart, R. (2014). Robot-centric elevation mapping with uncertainty estimates. In *Mobile Service Robotics* (World Scientific). 433–440

Forouhar, M., Čížek, P., and Faigl, J. (2021). SCARAB II: A small versatile six-legged walking robot. In *5th Full-Day Workshop on Legged Robots at IEEE International Conference on Robotics and Automation (ICRA)*. 1–2

Fritzke, B. (1994). A growing neural gas network learns topologies. In *Conference on Neural Information Processing Systems (NIPS)*. 625–632

Gonzalez, R. and Iagnemma, K. (2018). Slippage estimation and compensation for planetary exploration rovers. State of the art and future challenges. *Journal of Field Robotics* 35, 564–577. doi:10.1002/rob.21761

Guastella, D. C. and Muscato, G. (2021). Learning-Based Methods of Perception and Navigation for Ground Vehicles in Unstructured Environments: A Review. *Sensors* 21, 73. doi:10.3390/s21010073

Guerrero, E., Bonin-Font, F., and Oliver, G. (2021). Adaptive Visual Information Gathering for Autonomous Exploration of Underwater Environments. *IEEE Access* 9, 136487–136506. doi:10.1109/ACCESS.2021.3117343

Haddeler, G., Chan, J., You, Y., Verma, S., Adiwahono, A. H., and Meng Chew, C. (2020). Explore Bravely: Wheeled-Legged Robots Traverse in Unknown Rough Environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 7521–7526. doi:10.1109/IROS45743.2020.9341610

Helsgaun, K. (2000). An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic. *European Journal of Operational Research* 126, 106–130

Hollinger, G. A. and Sukhatme, G. S. (2014). Sampling-based robotic information gathering algorithms. *International Journal of Robotics Research* 33, 1271–1287. doi:10.1177/0278364914533443

Homberger, T., Bjelonic, M., Kottege, N., and Borges, P. V. K. (2016). Terrain-dependant Control of Hexapod Robots using Vision. In *International Symposium on Experimental Robotics (ISER)*. 92–102. doi:10.1007/978-3-319-50115-4_9

Karolj, V., Viseras, A., Merino, L., and Shutin, D. (2020). An Integrated Strategy for Autonomous Exploration of Spatial Processes in Unknown Environments. *Sensors* 20, 3663. doi:10.3390/s20133663

Kottege, N., Parkinson, C., Moghadam, P., Elfes, A., and Singh, S. P. N. (2015). Energetics-informed hexapod gait transitions across terrains. In *IEEE International Conference on Robotics and Automation (ICRA)*. 5140–5147. doi:10.1109/ICRA.2015.7139915

Krajník, T., Fentanes, J. P., Santos, J. M., and Duckett, T. (2017). Fremen: Frequency map enhancement for long-term mobile robot autonomy in changing environments. *IEEE Transactions on Robotics* 33, 964–977. doi:10.1109/TRO.2017.2665664

Krüsi, P., Bosse, M., and Siegwart, R. (2016). Driving on Point Clouds: Motion Planning, Trajectory Optimization, and Terrain Assessment in Generic Nonplanar Environments. *Journal of Field Robotics* 34, 940–984. doi:10.1002/rob.21700

Lin, B. and Song, S. (1993). Dynamic modeling, stability and energy efficiency of a quadrupedal walking machine. In *IEEE International Conference on Robotics and Automation (ICRA)*. 657–670. doi:10.1002/rob.8104

Luo, W. and Sycara, K. (2018). Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*. 6359–6364. doi:10.1109/ICRA.2018.8460473

Ma, K.-C., Liu, L., Heidarsson, H. K., and Sukhatme, G. S. (2018). Data-driven learning and planning for environmental sampling. *Journal of Field Robotics* 35, 643–661. doi:10.1002/rob.21767

Makarenko, A. A., Williams, S. B., Bourgault, F., and Durrant-Whyte, H. F. (2002). An experiment in integrated exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 534–539 vol.1. doi:10.1109/IRDS.2002.1041445

Mann, H. B. and Whitney, D. R. (1947). On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics* 18, 50–60. doi:10.1214/aoms/1177730491

Martin, S. and Corke, P. (2014). Long-term exploration tours for energy constrained robots with online proprioceptive traversability estimation. In *IEEE International Conference on Robotics and Automation (ICRA)*. 5778–5785. doi:10.1109/ICRA.2014.6907708

Mayuku, O., Surgenor, B. W., and Marshall, J. A. (2021). A Self-Supervised Near-to-Far Approach for Terrain-Adaptive Off-Road Autonomous Driving. In *IEEE International Conference on Robotics and Automation (ICRA)*. 14054–14060. doi:10.1109/ICRA48506.2021.9562029

McGhee, R. B. and Frank, A. A. (1968). On the stability properties of quadruped creeping gaits. *Mathematical Biosciences* 3, 331–351. doi:10.1016/0025-5564(68)90090-4

Moravec, H. and Elfes, A. (1985). High resolution maps from wide angle sonar. In *1985 IEEE International Conference on Robotics and Automation Proceedings*. 116–121. doi:10.1109/ROBOT.1985.1087316

Noon, C. E. (1988). *The Generalized Traveling Salesman Problem*. Ph.D. thesis, University of Michigan

Noon, C. E. and Bean, J. C. (1993). An Efficient Transformation Of The Generalized Traveling Salesman Problem. *Information Systems and Operational Research (INFOR)* 31, 39–44. doi:10.1080/03155986.1993.11732212

O'Callaghan, S., Ramos, F. T., and Durrant-Whyte, H. (2009). Contextual occupancy maps using Gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*. 1054–1060. doi:10.1109/ROBOT.2009.5152754

Ossenkopf, M., Castro, G., Pessacg, F., Geihs, K., and De Cristóforis, P. (2019). Long-Horizon Active SLAM system for multi-agent coordinated exploration. In *European Conference on Mobile Robots (ECMR)*. 1–6. doi:10.1109/ECMR.2019.8870952

O'Meadhra, C., Tabib, W., and Michael, N. (2019). Variable Resolution Occupancy Mapping Using Gaussian Mixture Models. *Robotics and Automation Letters* 4, 2015–2022. doi:10.1109/LRA.2018.2889348

Papadakis, P. (2013). Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence* 26, 1373–1385. doi:10.1016/j.engappai.2013.01.006

Pasolli, E. and Melgani, F. (2011). Gaussian process regression within an active learning scheme. In *IEEE International Geoscience and Remote Sensing Symposium*. 3574–3577. doi:10.1109/IGARSS.2011.6049994

Prudent, Y. and Ennaji, A. (2005). An incremental growing neural gas learns topologies. In *International Joint Conference on Neural Networks (IJCNN)*. vol. 2, 1211–1216. doi:10.1109/IJCNN.2005.1556026

Prágr, M., Čížek, P., Bayer, J., and Faigl, J. (2019a). Online incremental learning of the terrain traversal cost in autonomous exploration. In *Robotics: Science and Systems (RSS)*. 1–10

Prágr, M., Čížek, P., and Faigl, J. (2018). Cost of Transport Estimation for Legged Robot Based on Terrain Features Inference from Aerial Scan. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE), 1745–1750. doi:10.1109/IROS.2018.8593374

Prágr, M., Čížek, P., and Faigl, J. (2019b). Incremental learning of traversability cost for aerial reconnaissance support to ground units. In *2018 Modelling and Simulation for Autonomous Systems (MESAS)*. 412–421. doi:10.1007/978-3-030-14984-0_30

Quann, M., Ojeda, L., Smith, W., Rizzo, D., Castanier, M., and Barton, K. (2020). Off-road ground robot path energy cost prediction through probabilistic spatial mapping. *Journal of Field Robotics* 37, 421–439. doi:https://doi.org/10.1002/rob.21927

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., et al. (2009). ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*. 1–6

Ramos, F. and Ott, L. (2016). Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *International Journal of Robotics Research* 35, 1717–1730. doi:10.1177/0278364916684382

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning (Cambridge, Mass: MIT Press)

Rhodes, C., Liu, C., and Chen, W.-H. (2020). Informative Path Planning for Gas Distribution Mapping in Cluttered Environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 6726–6732. doi:10.1109/IROS45743.2020.9341781

Rényi, A. (1961). On measures of entropy and information. In *Berkeley Symposium on Mathematical Statistics and Probability*. 547–561

Schultz, A. C., Adams, W., and Yamauchi, B. (1999). Integrating Exploration, Localization, Navigation and Planning with a Common Representation. *Autonomous Robots* 6, 293–308. doi:10.1023/A:1008936413435

Shi, Y., Wang, N., Zheng, J., Zhang, Y., Yi, S., Luo, W., et al. (2020). Adaptive Informative Sampling with Environment Partitioning for Heterogeneous Multi-Robot Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 11718–11723. doi:10.1109/IROS45743.2020.9341711

Singh, A., Krause, A., Guestrin, C., Kaiser, W., and Batalin, M. (2007). Efficient planning of informative paths for multiple robots. In *International Joint Conference on Artifical Intelligence*. 2204–2211

Sofman, B., Lin, E., Bagnell, J. A., Cole, J., Vandapel, N., and Stentz, A. (2006). Improving Robot Navigation Through Self-Supervised Online Learning. *Journal of Field Robotics* 23, 1059–1075. doi:10.1002/rob.20169

Srinivas, N., Krause, A., Kakade, S., and Seeger, M. (2010). Gaussian process optimization in the bandit setting: no regret and experimental design. In *Intl. Conf. International Conference on Machine Learning (ICML)* (Haifa, Israel), 1015–1022

Stachniss, C., Grisetti, G., and Burgard, W. (2005). Information Gain-based Exploration Using Rao-Blackwellized Particle Filters. In *Robotics: Science and Systems (RSS)*. 1–8. doi:10.15607/RSS.2005.I.009

Stelzer, A., Hirschmüller, H., and Görner, M. (2012). Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. *International Journal of Robotics Research* 31, 381–402. doi:10.1177/0278364911435161

Vallvé, J. and Andrade-Cetto, J. (2015). Potential information fields for mobile robot exploration. *Robotics and Autonomous Systems* 69, 68–79. doi:10.1016/j.robot.2014.08.009

Viseras, A., Shutin, D., and Merino, L. (2019). Robotic Active Information Gathering for Spatial Field Reconstruction with Rapidly-Exploring Random Trees and Online Learning of Gaussian Processes. *Sensors* 19. doi:10.3390/s19051016

Wang, J. and Englot, B. (2016). Fast, accurate gaussian process occupancy maps via test-data octrees and nested Bayesian fusion. In *IEEE International Conference on Robotics and Automation (ICRA)*. 1003–1010. doi:10.1109/ICRA.2016.7487232

Wermelinger, M., Fankhauser, P., Diethelm, R., Krüsi, P., Siegwart, R., and Hutter, M. (2016). Navigation planning for legged robots in challenging terrain. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1184–1189. doi:10.1109/IROS.2016.7759199

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *CIRA* (IEEE), 146–151. doi:10.1109/CIRA.1997.613851

Zlot, R. and Stentz, A. (2006). Market-based multirobot coordination for complex tasks. *International Journal of Robotics Research* 25, 73–101

APPENDIX

1 TERRAIN CLUSTER EROSION AND DILATION

In practice, it is not desirable to place cost exploration goals at the boundaries of terrains classes because, in such areas, a real robot with the imprecise path following might fail to traverse the correct terrain, and the descriptors in such areas might be distant from the prototype $\mathrm{ta}(T)$. Besides, it might not be possible to acquire enough samples to learn the traversal cost on a small terrain area of a particular class. Hence, after assigning the terrain classes to cells, we erode cells that border different (or already eroded) terrain class using

$$T^{--}(\nu) = \begin{cases} T^{-}(\nu) & \text{if } \forall \nu' \in 8\mathrm{nb}(\nu) : T^{-}(\nu) = T^{-}(\nu'), \\ \varnothing & \text{otherwise,} \end{cases} \tag{25}$$

where $\varnothing$ is the eroded non-class terrain, T^{-} and T^{--} are the class assignments before and after an erosion step, respectively, and the erosion process is repeated $n_{\text{erode}}^{\text{steps}}$-times.

As a result of the erosion, some cells are assigned the eroded non-class $\varnothing$ with no prototype to use. Hence, when assigning cost predictions for path planning, we first dilate the terrain classes by selecting the most common class in the vicinity as

$$T^{++}(\nu) = \begin{cases} \operatorname{argmax}_{T \in \mathcal{T}} \sum_{\nu' \in 8\mathrm{nb}^{n_{\text{dilate}}^{\text{size}}}(\nu)} |T = T^{+}(\nu')| & \text{if } \exists \nu' \in 8\mathrm{nb}^{n_{\text{dilate}}^{\text{size}}}(\nu) : T^{+}(\nu') \neq \varnothing, \\ \varnothing & \text{otherwise,} \end{cases} \tag{26}$$

where $8\mathrm{nb}^{n_{\text{dilate}}^{\text{size}}}$ is the $n_{\text{dilate}}^{\text{size}}$-times repeated neighborhood function $8\mathrm{nb}$, T^{+} and T^{++} are the class assignments before and after a dilation step, respectively, and the dilation process is repeated $n_{\text{dilate}}^{\text{steps}}$-times.

2 GAUSSIAN PROCESS REGRESSION

Assuming function $f(x)$ that is observed with the noise ϵ

$$y = f(x) + \epsilon, \quad \epsilon \in \mathcal{N}(0, \sigma_{\epsilon}^{2}), \tag{27}$$

Gaussian Process (GP) is defined as the distribution

$$f(x) \sim \mathcal{GP}(m(x), K(x, x')), \tag{28}$$

where $m(x)$ is the mean

$$m(x) = E\left[f(x)\right], \tag{29}$$

and $K(x, x')$ is the covariance

$$K(x, x') = E\left[(f(x) - m(x))\left(f(x') - m(x')\right)\right]. \tag{30}$$

Given the training data X, the GP regressor's predictions and the query X_* are

$$\begin{aligned} \mu(X_*) &= K(X, X_*)\left[K(X, X) + \sigma_{\epsilon}^{2} I\right]^{-1} y, \\ (\sigma(X_*))^2 &= K(X_*, X_*) \\ &\quad - K(X, X_*)^T \left[K(X, X) + \sigma_{\epsilon}^{2} I\right]^{-1} K(X, X_*), \end{aligned} \tag{31}$$

where $K(X, X')$ is the covariance function.

3 INCREMENTAL GROWING NEURAL GAS

The *Incremental Growing Neural Gas* (IGNG) is a soft-computing clustering approach proposed by Prudent and Ennaji (2005). The approach builds on the *Growing Neural Gas* (GNG) (Fritzke, 1994), which adapts a graph topology to continually provided measurements. However, unlike the GNG, which is enlarged after a fixed number of measurement adaptation steps, the IGNG is only grown when adapting to a value x that is out of the bounds of the current structure.

Algorithm 11: Incremental Growing Neural Gas: Adaptation

Input: Ω – IGNG structure with terrain classes $\mathcal{T}$; x – Adapted measurement for the terrain descriptor $\mathtt{ta}$.

Output: Ω – IGNG structure for the terrain classes $\mathcal{T}$).

Procedure `adaptIGNG` (Ω, x)

- $\omega_1 \leftarrow \mathrm{argmin}_{\omega \in \Omega_{\text{neurons}}} \|x, \omega\|$ // Find the closest neuron to the adapted measurement.
- $\omega_2 \leftarrow \mathrm{argmin}_{\omega \in \Omega_{\text{neurons}}/\omega_1} \|x, \omega\|$ // Find the second closest.
- **if** $|\Omega_{neurons}| = 0 \vee \|x, \omega_1\| > \sigma^{IGNG}$ **then** // If there are no neurons or the closest is too far.
  - $\Omega_{\text{neurons}} \leftarrow \Omega \cup \omega_{\text{new}}, \omega_{\text{new}} = x$ // Add the measurement as a new neuron.
- **else**
  - **if** $|\Omega_{neurons}| = 1 \vee \|x, \omega_2\| > \sigma^{IGNG}$ **then** // If there is only 1 neuron or the second closest is too far.
    - $\Omega_{\text{neurons}} \leftarrow \Omega_{\text{neurons}} \cup \omega_{\text{new}}, \omega_{\text{new}} = x$ // Add the measurement as a new neuron.
    - $\Omega_{\text{connections}} \leftarrow \Omega_{\text{connections}} \cup (x, \omega_1)$ // Connect the new neuron with the closest.
  - **else**
    - $\omega_1 \leftarrow \omega_1 + \epsilon_1^{\text{IGNG}}(x - \omega_1)$ // Warp the closest neuron to the measurement.
    - **for** $\omega_{nb} \in nb(\omega_1)$ **do** // For each neighbor of the closest neuron.
      - $\omega_{\text{nb}} \leftarrow \omega_{\text{nb}} + \epsilon_{\text{nb}}^{\text{IGNG}}(x - \omega_{\text{nb}})$ // Warp it to the measurement.
      - $a(\omega_1, \omega_{\text{nb}}) \leftarrow a(\omega_1, \omega_{\text{nb}}) + 1$ // And age their connections.
    - **if** $(\omega_1, \omega_2) \in \Omega_{connections}$ **then** // If the first and closest are connect.
      - $a((\omega_1, \omega_2)) \leftarrow 0$ // Reset the connection age.
    - **else**
      - $\Omega_{\text{connections}} \leftarrow \Omega_{\text{connections}} \cup (\omega_1, \omega_2)$ // Otherwise insert new connection.
    - **for** $\omega_{nb} \in nb(\omega_1)$ **do** // For each neighbor of the closest neuron.
      - $a(\omega_{\text{nb}}) \leftarrow a(\omega_{\text{nb}}) + 1$ // Age the neighbor.
- **for** $(\omega_a, \omega_b) \in \Omega_{connections} : a((\omega_a, \omega_b)) > a_{max}^{IGNG}$ **do** // Find too old connections.
  - $\Omega_{\text{connections}} \leftarrow \Omega_{\text{connections}}/(\omega_a, \omega_b)$ // And remove them.
- **for** $\omega \in \Omega_{neurons} : a(\omega) \geq a_{mature}^{IGNG}$ **do** // Find isolated mature neurons.
  - **if** $\neg\exists \omega' \Omega_{neurons} : (\omega, \omega') \in \Omega_{connections}$ **then** // And remove them.
    - $\Omega_{\text{neurons}} \leftarrow \Omega_{\text{neurons}}/\omega$
- **return** Ω

The IGNG adaptation is summarized in Alg. 11, and it operates as follows[4]. The algorithm keeps a graph of neurons (graph vertices) and their connections (graph edges) and keeps an age value for each neuron and connection. When adapting to a new measurement x, the algorithm first finds the closest neuron ω_1 and the second closest neuron ω_2. If the graph is empty or the closest neuron is too far with $\|x - \omega_1\| > \sigma^{\text{IGNG}}$, a new embryo neuron ω_{new} with the age $a(\omega_{\text{new}}) = 1$ is inserted at x. If ω_1 is close enough, but the second closest ω_2 is not, or there is only one neuron in the graph, a new neuron is also inserted at x. Moreover, an edge between the new neuron and ω_1 is inserted into the graph with the age $a((\omega_1, \omega_{\text{new}})) = 0$.

If both ω_1 and ω_2 are close enough, ω_1 and all of its neighbors (neurons with an existing connection to ω_1) are warped towards x by ϵ_1^{IGNG} and $\epsilon_{\text{nb}}^{\text{IGNG}}$, respectively. Then, if there is already a connection between ω_1 and ω_2, its age is set to 0. Otherwise, the connection is created. Next, the ages of all neighbors $a(\omega_{\text{nb}})$ of ω_1 and their respective connections $a((\omega_1, \omega_{\text{nb}}))$ are incremented by one.

After adapting to the measurement, the graph is pruned to remove old connections and isolated neurons. In general, it is desirable for neurons to be old (since measurements were often observed near then) and for connections to be young (since measurements were recently observed along the edge). First, we identify neurons that are mature with $a(\omega) \geq a_{\text{mature}}^{\text{IGNG}}$. Then, connections that are too old with $a((\omega, \omega')) > a_{\text{max}}^{\text{IGNG}}$ are removed from the graph. If it leads to isolated mature neurons, these are also removed.

[4] The herein presented description is limited to the basic operation of the algorithm and omits its use for semi-supervised labeling since it is not used in the presented work. We refer the interested reader to Prudent and Ennaji (2005).