EVORA: Deep Evidential Traversability Learning for Risk-Aware Off-Road Autonomy

Xiaoyi Cai, Siddharth Ancha, *Member, IEEE*, Lakshay Sharma, *Student Member, IEEE*, Philip R. Osteen, *Member, IEEE*, Bernadette Bucher, Stephen Phillips, *Member, IEEE*, Jiuguang Wang, *Member, IEEE*, Michael Everett, *Member, IEEE*, Nicholas Roy, *Senior Member, IEEE*, and Jonathan P. How, *Fellow, IEEE*

***Abstract*—Traversing terrain with good traction is crucial for achieving fast off-road navigation. Instead of manually designing costs based on terrain features, existing methods learn terrain properties directly from data via self-supervision to automatically penalize trajectories moving through undesirable terrain, but challenges remain in properly quantifying and mitigating the risk due to uncertainty in the learned models. To this end, we present evidential off-road autonomy (EVORA), a unified framework to learn uncertainty-aware traction model and plan risk-aware trajectories. For uncertainty quantification, we efficiently model both aleatoric and epistemic uncertainty by learning discrete traction distributions and probability densities of the traction predictor's latent features. Leveraging evidential deep learning, we parameterize Dirichlet distributions with the network outputs and propose a novel uncertainty-aware squared Earth Mover's Distance loss with a closed-form expression that improves learning accuracy and navigation performance. For risk-aware navigation, the proposed planner simulates state trajectories with the worst-case expected traction to handle aleatoric uncertainty and penalizes trajectories moving through terrain with high epistemic uncertainty. Our approach is extensively validated in simulation and on wheeled and quadruped robots, showing improved navigation performance compared to methods that assume no slip, assume the expected traction, or optimize for the worst-case expected cost.**

***Index Terms*—Autonomous robots, self-supervised learning, uncertainty quantification, off-road navigation.**

Manuscript received 8 March 2024; revised 4 July 2024; accepted 5 July 2024. Date of publication 22 July 2024; date of current version 9 August 2024. This work was supported in part by ARL under Grant W911NF-21-2-0150 and in part by ONR under Grant N00014-18-1-2832. Distribution Statement A: Approved for public release: distribution unlimited. This article was recommended for publication by Associate Editor S. Caron and Editor S. Behnke upon evaluation of the reviewers' comments. *(Corresponding author: Xiaoyi Cai.)*

Xiaoyi Cai, Siddharth Ancha, Lakshay Sharma, Nicholas Roy, and Jonathan P. How are with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: xyc@mit.edu; sancha@mit.edu; lakshays@mit.edu; nickroy@mit.edu; jhow@mit.edu).

Philip R. Osteen is with the DEVCOM Army Research Laboratory, Adelphi, MD 20783 USA (e-mail: philip.r.osteen.civ@army.mil).

Bernadette Bucher is with the University of Michigan, Ann Arbor, MI 48109 USA, and also with the Boston Dynamics AI Institute, Cambridge, MA 02142 USA (e-mail: bucherb@umich.edu).

Stephen Phillips and Jiuguang Wang are with the Boston Dynamics AI Institute, Cambridge, MA 02142 USA (e-mail: sphillips@theaiinstitute.com; jw@theaiinstitute.com).

Michael Everett is with the Northeastern University, Boston, MA 02115 USA (e-mail: m.everett@northeastern.edu).

Supplementary materials are available at https://xiaoyi-cai.github.io/evora.

Digital Object Identifier 10.1109/TRO.2024.3431828

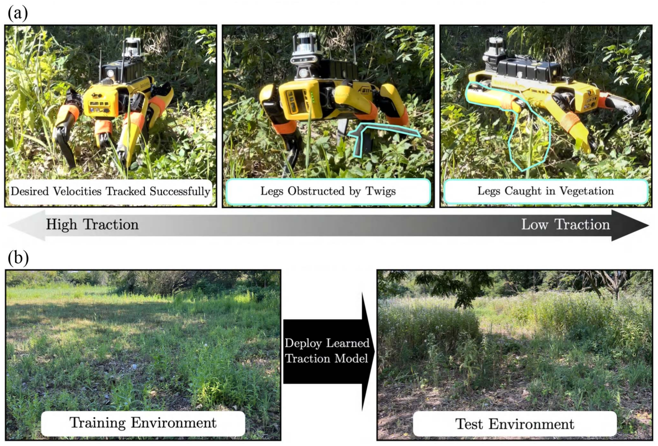

(a)
Desired Velocities Tracked Successfully
Legs Obstructed by Twigs
Legs Caught in Vegetation
High Traction
Low Traction
(b)
Training Environment
Deploy Learned Traction Model
Test Environment

Fig. 1. EVORA captures both aleatoric and epistemic uncertainty when learning a terrain traction model, where traction is defined as the ratio between achieved and commanded velocities. (a) *Aleatoric uncertainty* is the inherent and irreducible uncertainty due to partial observability. For example, visually similar terrain may have different traction values due to complex interactions between the robot and vegetation. (b) *Epistemic uncertainty* is the model uncertainty due to the distribution shift between training and test environments, limiting the reliability of the learned model at test time.

I. Introduction

Autonomous robots are increasingly being deployed in harsh off-road environments like mines, forests, and deserts [1], [2], [3], where both geometric and semantic understanding of the environments is required to identify nongeometric hazards (e.g., mud puddles, slippery surfaces) and geometric nonhazards (e.g., tall grass and foliage) in order to achieve reliable navigation. To this end, recent approaches manually assign navigation costs based on the semantic classification of the terrain [4], [5], [6], requiring significant human expertise to label and train a classifier sufficiently accurate and rich in order to achieve desired risk-aware behaviors. Alternatively, self-supervised learning can be used to learn a model of traversability directly from navigation data [7], [8], [9] to automatically assign higher costs for undesirable terrain during planning. Because self-supervised data collection in the real world can be slow and expensive, collecting more data is not beneficial unless we properly quantify and mitigate the risk due to uncertainty in the learned models. Uncertainty manifests in two forms, as illustrated in Fig. 1 in the context of off-road navigation.

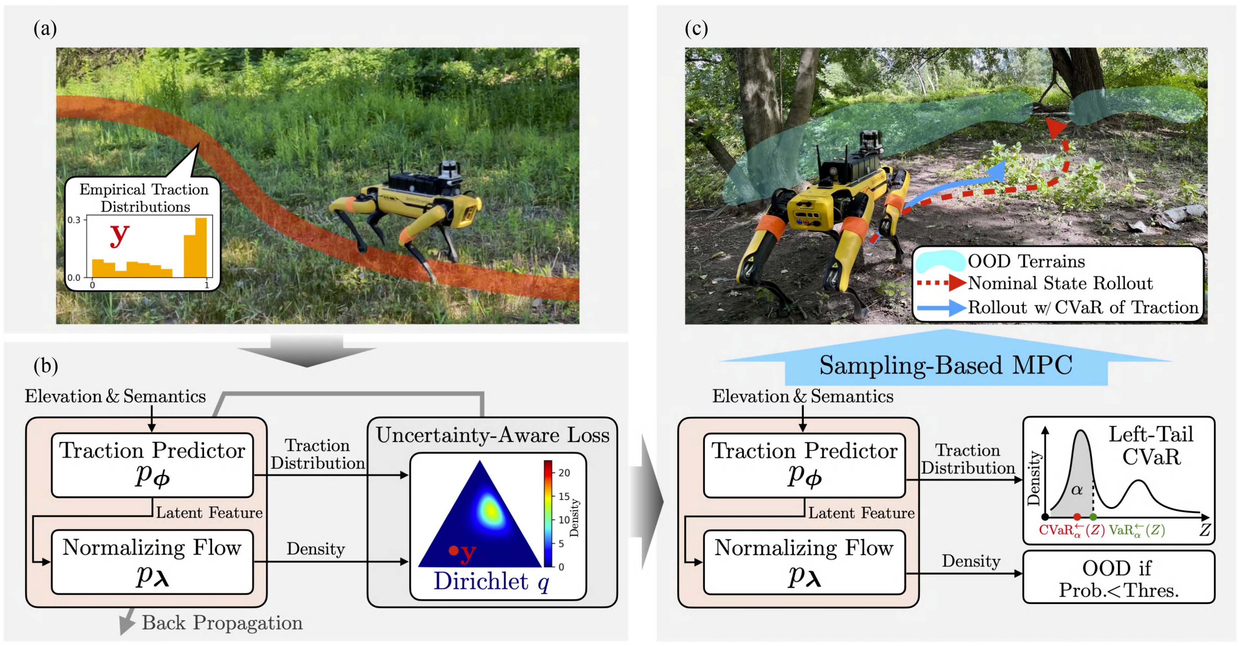


Fig. 2. Overview of the proposed *uncertainty-aware traversability learning* and *risk-aware navigation* methods. (a) For data collection, we drive the robot over interesting terrain to record traction values, robot positions, and build a semantic elevation map. We generate the training dataset offline by extracting semantic and elevation features of the terrain and estimating empirical traction distributions along the traversed path. (b) Leveraging evidential deep learning [16], we learn categorical distributions over discretized traction values to capture *aleatoric uncertainty* and estimate *epistemic uncertainty* by using a normalizing flow network [17] to learn the densities of the traction predictor's latent features. The overall architecture is trained with the proposed uncertainty-aware loss defined for the Dirichlet distribution parameterized by the network outputs. (c) To handle aleatoric uncertainty, we propose a risk-aware planner that uses the left-tail CVaR of the traction distribution to forward simulate the robot states when using the sampling-based model predictive control (MPC) method [18]. To handle epistemic uncertainty, we threshold the densities of the traction predictor's latent features in order to identify and avoid OOD terrain with unreliable traction predictions via auxiliary planning costs.

Aleatoric uncertainty is the inherent and irreducible uncertainty due to partial observability. For example, two patches of terrain may be indistinguishable to the onboard sensors but lead to different vehicle behaviors—such uncertainty cannot be reduced by collecting more data. *Epistemic uncertainty* is due to out-of-distribution (OOD) inputs encountered at test time that are not well represented in the training data. Because it is often undesirable to collect OOD data in dangerous situations such as collisions and falling at the edge of a cliff, there can exist a large gap between training datasets and the various real-world scenarios encountered by the robot. Most existing work in off-road navigation has focused on either aleatoric uncertainty [10], [11] by learning distributions of system parameters instead of point estimates, or epistemic uncertainty [12], [13], [14], [15] by identifying OOD terrain, but limited effort has been made to quantify both types of uncertainty and mitigate the associated risk during planning.

To achieve fast and reliable off-road navigation, this work considers both the upstream *uncertainty-aware traversability learning* problem and the downstream *risk-aware navigation* problem. Recognizing the interdependence of the two problems, our proposed pipeline evidential off-road autonomy (EVORA), tightly integrates the proposed uncertainty-aware traversability model into the proposed risk-aware planner (see Fig. 2 for an overview). To plan fast trajectories, we model traversability with terrain *traction* that captures the "slip" or the ratio between achieved and commanded velocities (for example, wet terrain that causes the robot's wheels to slip and reduce its intended velocity has low traction). Moreover, we efficiently quantify both aleatoric and epistemic uncertainty by learning the empirical traction distributions and probability densities of the traction predictor's latent features. Because real-world traction distributions may be multimodal, as shown in Fig. 1(a) where vegetation with similar appearance may lead to different traction values, we learn categorical distributions over discretized traction values to capture multimodality. By leveraging the evidential deep learning technique proposed in [16], we parametrize Dirichlet distributions (the conjugate priors for the categorical distributions) with neural network (NN) outputs, and propose a novel uncertainty-aware loss based on the squared Earth Mover's Distance (EMD) [19]. Our loss, which can be computed efficiently in closed-form, better captures the relationship among discretized traction values than the conventional cross-entropy (CE)-based losses [20]. To handle aleatoric uncertainty, we propose a risk-aware planner that simulates state trajectories using the worst-case expected traction, which is shown to achieve improved or competitive performance compared to state-of-the-art methods that rely on the nominal traction [11], the expected traction [21] or that optimize for the worst-case expected cost [22]. To mitigate the risk due to epistemic uncertainty, the proposed method imposes a confidence threshold on the densities of the traction predictor's latent space features to identify OOD terrain and avoid moving through it using auxiliary planning costs. The overall approach is extensively analyzed in simulation and hardware with wheeled and quadruped robots, demonstrating feasibility and improved navigation performance in practice.

A. Related Work

1) Traversability Analysis: Suitability of terrain for navigation can be assessed in various ways, e.g., based on proprioceptive measurements [23], [24], geometric features [1], [2], [25] and combinations of geometric and semantic features [3], [4], [26] (see the survey in [27]). Due to the difficulty of handcrafting planning costs based on terrain features, self-supervised learning is increasingly being adopted to learn task-relevant traversability representations. For example, Li et al. [28] proposed learning the support surfaces underneath dense vegetation for legged robot locomotion, and Gasparino et al. [21] modeled terrain traction that captures how well the robot can follow the desired velocities. However, these methods do not account for the aleatoric and epistemic uncertainty due to the noisiness and scarcity of real-world data. To capture aleatoric uncertainty, Ewen et al. [10] and Cai et al. [11] learned multimodal terrain properties via Gaussian mixture models or categorical distributions. To capture epistemic uncertainty, Frey et al. [12] and Schmid et al. [13] measured the trained NNs' ability to reconstruct terrain similar to the terrain types traversed in the past, and Seo et al. [29] trained a binary classifier for unfamiliar terrain. In comparison, Endo et al. [15] and Lee et al. [14] leveraged Gaussian process (GP) regression to quantify epistemic uncertainty, but they used a homoscedastic noise model that assumes the noise variance is globally constant. While Murphy et al. [30] adopted heteroscedastic GPs that can handle input-dependent noise, the predictive distributions are not analytically tractable and require approximations.

In contrast, our work explicitly quantifies both the aleatoric and epistemic uncertainty in the learned traction model that predicts the ratio between achieved and commanded velocities. While we learn traction just like Gasparino et al. [21], our model is uncertainty-aware and can be used to achieve risk-aware navigation. In comparison, Frey et al. [12] used the difference between achieved and commanded velocities in the planning objective, but they assumed no slip when simulating the state rollouts. In contrast, our traction model can be used to simulate state rollouts under the worst-case expected traction condition, which is shown by our results to achieve better performance than methods that assume nominal traction.

2) Uncertainty Quantification and OOD Detection: Uncertainty quantification is well studied in the machine learning literature (see the survey in [31]) with effective techniques such as Bayesian dropout [32], model ensembles [33], and evidential methods [34]. In the off-road navigation literature, ensemble methods have been a popular choice [35], [36], [37], because they typically outperform methods based on Bayesian dropout [38]. In comparison, evidential methods are better suited for real-world deployment, because they only require a single network evaluation without imposing high computation or memory requirements. Therefore, we leverage the evidential method proposed by Charpentier et al. [16] to directly parameterize the conjugate prior distribution of the target distribution with NN outputs in order to quantify both aleatoric and epistemic uncertainty. Moreover, we propose an uncertainty-aware loss based on the squared EMD proposed by Hou et al. [19] to better capture the relationship among the discrete traction values, resulting in more accurate traction predictions that in turn improve the downstream risk-aware planner's navigation performance.

When deploying the learned traction model, we explicitly identify OOD terrain based on the estimated epistemic uncertainty, which is an instance of the general OOD detection problem (see the survey in [39]). For example, reconstruction-based method adopted by Seo et al. [40] and density-based method adopted by Ancha et al. [41] have shown promising results for off-road navigation to identify unsafe terrain. Similar to Ancha et al. [41], our approach is a density-based approach that explicitly captures the normalized probability density under the training data distribution. Alternatively, energy-based approaches proposed by Liu et al. [42] and Grathwohl et al. [43] do not require explicit density normalization, and similar ideas have been adopted by Castaneda et al. [44] to avoid OOD states. Instead of solely focusing on OOD detection and mitigation, this work quantifies and mitigates the risk due to both aleatoric and epistemic uncertainty. While OOD terrain with high epistemic uncertainty should be avoided at test time, in-distribution terrain may still lead to high aleatoric uncertainty in the predicted traction due to complex vehicle-terrain interactions. Therefore, the risk due to aleatoric uncertainty should be mitigated separately to improve navigation performance by allowing the robot to tradeoff the likelihood of experiencing low traction with the potential time savings from traversing terrain with uncertain traction.

3) Risk-Aware Planning: The risk of traversing terrain with uncertain traversability values has been represented as costmaps by Fan et al. [45] and Triest et al. [35], where conditional value at risk (CVaR) can be used to measure the cost of encountering worst-case expected failures, which satisfies a group of axioms important for rational risk assessment [46]. Instead of costmaps, navigation performance has also been assessed based on the expected future states by Gibson et al. [47] or the expected terrain traction by Gasparino et al. [21]. However, these methods rely on either the nominal or the expected system behavior, which may provide a poor indication of the actual performance when the vehicle-terrain interaction is noisy (i.e., high aleatoric uncertainty). Alternatively, Wang et al. [22] proposed to directly optimize the CVaR of the planning objective, which can be estimated by evaluating each control sequence over samples of uncertain parameters, but this approach is computationally expensive. Similar to our approach, the recent work of Lee et al. [36] quantified both aleatoric and epistemic uncertainty using probabilistic ensembles [48] and planned risk-aware trajectories by penalizing both types of uncertainty, but it relied on the expected system behaviors.

While we adopt a similar strategy used by Lee et al. [36] for handling epistemic uncertainty via auxiliary penalties, we use the worst-case expected system parameters for forward simulation to assess the risk due to aleatoric uncertainty. Our approach is computationally more efficient than the method

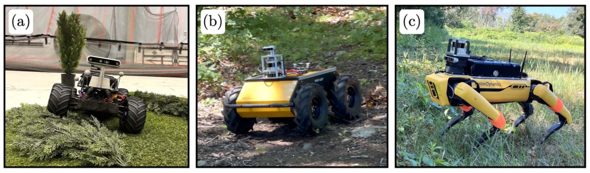


Fig. 3. Example ground robots that can be modeled with unicycle or bicycle dynamics models. (a) RC car. (b) Differential-drive robot. (c) Legged robot.

proposed by Wang et al. [22] and produces behaviors more robust to multimodal terrain properties observed in the real world compared to methods proposed by Lee et al. [36] and Gasparino et al. [21] based on the expected system behaviors.

B. Contributions

We present EVORA, an off-road navigation pipeline that tightly integrates the solutions to the uncertainty-aware traversability learning problem and the risk-aware motion planning problem. We explicitly quantify both the epistemic uncertainty to understand when the predicted traction values are unreliable due to novel terrain and the aleatoric uncertainty to enable the downstream planner to mitigate risk due to noisy traction estimates. The main contributions of this work are as follows.

1) A probabilistic traversability model based on traction distributions (aleatoric uncertainty), with the ability to identify unreliable predictions via the densities of the traction predictor's latent features (epistemic uncertainty).
2) A novel uncertainty-aware loss based on the squared EMD loss [19] with a closed-form expression derived in this work that improves traction prediction accuracy, OOD detection performance, and downstream navigation performance when used together with the uncertainty-aware cross entropy (UCE) loss [16].
3) A risk-aware planner based on the CVaR of traction to handle aleatoric uncertainty. Our planner outperforms methods that assume the nominal traction [11] or the expected traction [21], and achieves improved or competitive performance compared to the method optimizing for the CVaR of cost [22] in both simulation and hardware.
4) A further extension of the risk-aware planner to handle epistemic uncertainty by avoiding OOD terrain, which improves the navigation success rate in simulation and reduces human interventions in hardware experiments.

The preliminary conference version of this work appeared in [49], which proposed to learn traction distributions and use CVaR of traction for planning. This work extends the prior work by using the evidential learning technique proposed in [16] for model training, and deriving a new uncertainty-aware EMD^2 loss based on [19] to improve learning performance. The new methods introduced in this work not only improve the accuracy of traction prediction and OOD detection but also lead to faster navigation. By adding extensive hardware experiments, this work provides stronger evidence of the performance improvements provided by the risk-aware planner proposed in the conference version [49] compared to the state-of-the-art methods [11], [21], [22].

II. Problem Overview

We consider the problem of fast navigation to a goal for a ground vehicle whose dynamics depend on the terrain traction. Because traction values can be uncertain, we introduce dynamical models whose traction values are random variables in Section II-A. Moreover, we introduce the planning objective that measures the time-to-goal in Section II-B and discuss the challenges of minimizing the time-to-goal in Section II-C.

A. Dynamical Models With Traction Parameters

Consider the discrete time system

$$\mathbf{x}_{t+1} = F(\mathbf{x}_t, \mathbf{u}_t, \boldsymbol{\psi}_t) \tag{1}$$

where $\mathbf{x}_t \in \mathbf{X} \subseteq \mathbb{R}^n$ is the state vector such as the position and orientation of the robot, $\mathbf{u}_t \in \mathbb{R}^m$ is the control input provided to the robot, and $\boldsymbol{\psi}_t \in \boldsymbol{\Psi} \subseteq \mathbb{R}^r$ is the parameter vector that captures terrain traction. We consider two models that are useful approximations of the dynamics of a wide range of robots, as shown in Fig. 3. Applicable to both differential-drive and legged robots, the unicycle model is defined as

$$\begin{bmatrix} p^x_{t+1} \\ p^y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} p^x_t \\ p^y_t \\ \theta_t \end{bmatrix} + \Delta \cdot \begin{bmatrix} \psi_{1,t} \cdot v_t \cdot \sin(\theta_t) \\ \psi_{1,t} \cdot v_t \cdot \cos(\theta_t) \\ \psi_{2,t} \cdot \omega_t \end{bmatrix} \tag{2}$$

where $\mathbf{x}_t = [p^x_t, p^y_t, \theta_t]^\top$ contains the X, Y positions and yaw, $\mathbf{u}_t = [v_t, \omega_t]^\top$ contains the commanded linear and angular velocities, $\boldsymbol{\psi}_t = [\psi_{1,t}, \psi_{2,t}]^\top$ contains the linear and angular traction values $0 \le \psi_{1,t}, \psi_{2,t} \le 1$, and $\Delta > 0$ is the time interval. Intuitively, traction captures the "slip," or the ratio between achieved and commanded velocities. The bicycle model is applicable for Ackermann-steering robots and is defined as

$$\begin{bmatrix} p^x_{t+1} \\ p^y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} p^x_t \\ p^y_t \\ \theta_t \end{bmatrix} + \Delta \cdot \begin{bmatrix} \psi_{1,t} \cdot v_t \cdot \cos(\theta_t) \\ \psi_{1,t} \cdot v_t \cdot \sin(\theta_t) \\ \psi_{2,t} \cdot v_t \cdot \tan(\delta_t)/L \end{bmatrix} \tag{3}$$

where L is the wheelbase, $\mathbf{u}_t = [v_t, \delta_t]^\top$ contains the commanded linear velocity and steering angle, and $\boldsymbol{\psi}_t$ plays the same role as in the unicycle model. The reference point for the bicycle model in (3) is located at the center between the two rear wheels.

B. Planning Objective

We adopt the minimum-time objective proposed in [11], but other objectives for goal reaching could also be used. Intuitively, the objective only assigns stage costs by accumulating the elapsed time before any state falls in the goal region. If the state trajectory does not intersect the goal region, the terminal cost further penalizes the estimated time-to-goal. Given a function $C^{\text{dist}}(\mathbf{x}_t)$ that measures the Euclidean distance between $\mathbf{x}_t$ and the goal, the minimum-time objective is defined over a state

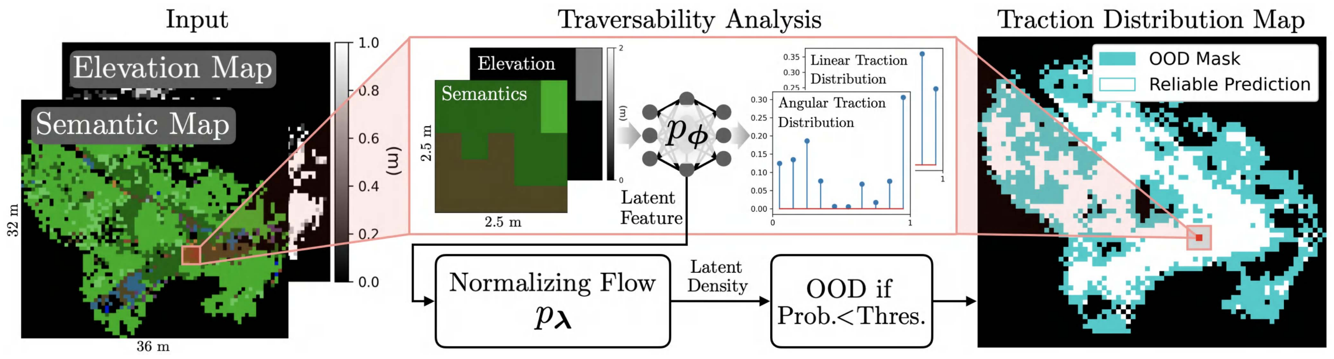


Fig. 4. Proposed traversability pipeline maps elevation and semantic features to traction distributions that capture aleatoric uncertainty, and densities for latent features that capture epistemic uncertainty. Terrain regions are deemed OOD and later avoided during planning if the densities for the latent features are below a threshold. When the densities for latent features are above the threshold, the predicted traction distributions are reliable and inform downstream risk-aware planners (Section IV) to tradeoff the risk of immobilization with the time savings of traversing regions with uncertain traction.

trajectory $\mathbf{x}_{0:T}$ from time 0 to T

$$C(\mathbf{x}_{0:T}) := C^{\text{term}}(\mathbf{x}_{0:T}) + \sum_{t=0}^{T-1} C^{\text{stage}}(\mathbf{x}_{0:t}) \tag{4}$$

where the total cost consists of terminal and stage costs

$$C^{\text{term}}(\mathbf{x}_{0:T}) = \frac{C^{\text{dist}}(\mathbf{x}_T)}{s^{\text{default}}}\left(1 - \mathbb{1}^{\text{done}}(\mathbf{x}_{0:T})\right) \tag{5}$$

$$C^{\text{stage}}(\mathbf{x}_{0:t}) = \Delta\left(1 - \mathbb{1}^{\text{done}}(\mathbf{x}_{0:t})\right) \tag{6}$$

where $s^{\text{default}} > 0$ is the default speed for estimating time-to-go and $\Delta > 0$ is the constant time interval. To avoid accumulating costs after arriving at the goal, we use an indicator function $\mathbb{1}^{\text{done}}(\mathbf{x}_{0:t})$ that equals 1 if any state in $\mathbf{x}_{0:t}$ has reached the goal, and equals 0 otherwise. Although Δ is a constant, the number of time steps required to reach the goal changes according to the robot speed. Intuitively, this objective encourages the robot to reach the goal as quickly as possible.

C. Key Challenges

While objective (4) can be optimized by finding an optimal control sequence via nonlinear optimization techniques such as model predictive path integral control (MPPI [18, Algorithm 2]), the terrain traction varies across terrain types and needs to be learned from real-world data. However, real-world terrain traction is uncertain since visually and geometrically similar terrain may have different traction properties (aleatoric uncertainty), and the traction models can only be trained on limited data (epistemic uncertainty). Even if uncertainty in terrain traction is quantified accurately, designing risk-aware planners that mitigate the risk of failure under this uncertainty is still challenging. To address these challenges, we introduce our proposed uncertainty-aware traversability model and the risk-aware planner in Sections III and IV, respectively.

III. Uncertainty-Aware Traversability Model

In this section, we first introduce the traction distribution predictor that captures aleatoric uncertainty, and the latent space density estimator that captures epistemic uncertainty. An overview of the traversability analysis pipeline is shown in Fig. 4. Then, we review the evidential method proposed by [16] in the context of traction learning, and propose a new uncertainty-aware loss to improve learning performance.

A. Aleatoric Uncertainty Captured in Traction Distribution

Let $\boldsymbol{\Psi} = \{\psi^1, \ldots, \psi^B\}$ be a set of $B > 0$ discretized traction values (ratios between achieved and commanded velocities), and $\boldsymbol{O}$ be a set of terrain features containing elevation values and one-hot vectors of semantic labels. We want to model the distribution over $\boldsymbol{\Psi}$ given an input terrain feature vector $\mathbf{o} \in \boldsymbol{O}$

$$p_{\phi}(\mathbf{o}) : \boldsymbol{O} \to \mathbb{R}^B_{\geq 0}. \tag{7}$$

We use $\text{Cat}(p_\phi(\mathbf{o}))$ to denote a categorical distribution over $\boldsymbol{\Psi}$ that captures the aleatoric uncertainty due to environment factors that affect traction but are not captured in the terrain features $\mathbf{o}$. Note that (7) can be learned by an NN parameterized by ϕ that can be trained using an empirically collected dataset $\{(\mathbf{o}, \psi)_k\}_{k=1}^K$ where $K > 0$. While we do not explicitly account for the uncertainty in the terrain features (e.g., noisy elevation estimates, or misclassification of terrain types due to visual similarity) or other factors such as the design of the low-level velocity controller, these unmodeled effects will manifest in the empirical dataset and can still be implicitly captured in the learned traction distributions.

We use categorical distributions as convenient alternatives to Gaussian mixture models and normalizing flows [17] for learning multimodal traction distributions observed in practice, because they do not require tuning the number of clusters, generate bounded distributions by construction, and converge faster than normalizing flows based on our empirical experience while achieving similar accuracy. Since we only need to discretize 1-D linear and angular traction values, we avoid the common problem of an exponentially increasing number of bins when discretizing high-dimensional spaces. Therefore, categorical distributions with a relatively small number of discrete bins suffice for our task.

Examples of real-world data collection and offline dataset generation can be found in Fig. 5. The semantic and geometric information about the environment can be built by using a semantic octomap [50] that temporally fuses semantic point

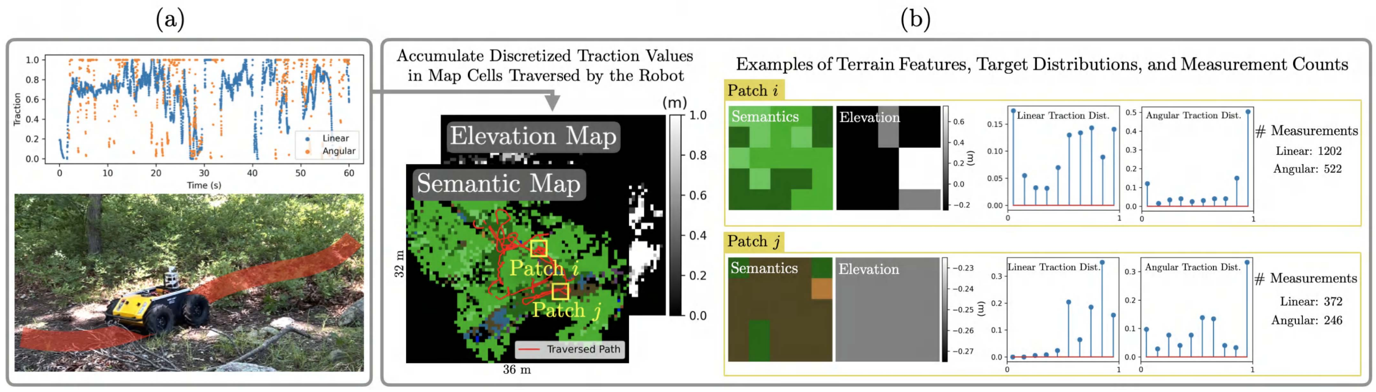


Fig. 5. Data collection and offline dataset generation. (a) Example of real-world data collection using a Clearpath Husky. The robot is manually driven for 10 min while recording the traversed path, traction values, and building semantic and elevation maps of the environment. The traction values are recorded at 20 Hz and only a subset of the collected traction values are shown for clarity, where the discontinuity in traction values occurs when linear or angular commands are not sent. (b) During offline dataset generation, traction values are discretized and accumulated via histograms stored in traversed map cells. The input to the traction predictor consists of semantic and elevation patches. Example terrain types include vegetation (light green), grass (dark green), dirt (light brown), and mulch (dark brown). The predicted and empirical traction distributions are used to compute the training loss, and the associated measurement counts used to obtain the empirical traction distributions can be used to weight the training loss to discount rarely visited terrain. (a) Data collection. (b) Offline datatset generation.

clouds. We use PointRend [51] trained on the RUGD off-road navigation dataset [52] with 24 semantic categories to segment RGB images and subsequently project the semantics onto lidar point clouds. During offline dataset generation, we obtain the empirical linear and angular traction distributions by accumulating discretized traction measurements in histograms stored in every terrain cell traversed by the robot. The measurement counts are also stored so that, during training, we can weight the loss for each cell by the measurement counts to discount rarely visited terrain. In practice, we learn the linear and angular traction distributions separately. We use a shared encoder (convolutional layers followed by fully connected layers) to process the semantic and elevation patches of the terrain. The shared encoder is followed by two fully connected decoder heads with soft-max outputs for predicting the linear and angular traction distributions.

B. Epistemic Uncertainty Captured in Latent Space Density

Due to limited training data, the predicted traction distributions for novel parts of the terrain may be unreliable and lead to degraded navigation performance in those regions. To measure epistemic uncertainty, we want to estimate the density of the latent feature $\mathbf{z^o} \in \mathbb{R}^H$ obtained from an intermediate layer of the traction predictor p_ϕ (7) based on the terrain feature $\mathbf{o}$. The density estimator is defined as

$$p_{\boldsymbol{\lambda}}(\mathbf{z^o}) : \mathbb{R}^H \to \mathbb{R}_{\geq 0} \tag{8}$$

where we use a normalizing flow parameterized by $\boldsymbol{\lambda}$ to learn (8). At a high level, a normalizing flow works by transforming an arbitrary target distribution into a simple base distribution, such as a standard normal, via a sequence of invertible and differentiable mappings. Then, the density of a sample $\mathbf{z^o}$ can be computed by the change of variable formula [17]—it is the product of the density of the transformed sample under the base distribution, and the change in volume measured by the determinant of the Jacobian of the transformation. When selecting the latent space features, it is crucial to ensure that they contain task-relevant information. To this end, we use the latent features produced by the shared terrain feature encoder, because they contain information useful for predicting both linear and angular traction distributions.

For interpretation purposes, we design a simple confidence score $g(\mathbf{z^o})$ for input feature $\mathbf{o}$ based on the maximum density $p^{\max} \in \mathbb{R}_{\geq 0}$ and minimum density $p^{\min} \in \mathbb{R}_{\geq 0}$ observed for the latent features of terrain in the training dataset

$$g(\mathbf{z^o}) = \frac{p_{\boldsymbol{\lambda}}(\mathbf{z^o}) - p^{\min}}{p^{\max} - p^{\min}}. \tag{9}$$

During deployment, terrain features with a confidence score below some threshold $g^{\text{thres}} \in \mathbb{R}$ are deemed OOD; these regions with OOD terrain features can be explicitly avoided during planning via auxiliary penalties. A principled way to set g^{thres} is to use the κth percentile of the densities obtained from all the terrain features in the training dataset, where a higher value of $\kappa \in [0, 100]$ will cause more terrain features to be classified as OOD at test time. Because of the normalization in (9), $g^{\text{thres}} = 0$ and $g^{\text{thres}} = 1$ conveniently correspond to the 0th and 100th percentiles. Note that the threshold can be selected offline, and $g^{\text{thres}} = 0$ can be used if the robot should only avoid terrain features with densities lower than densities observed during training. This strategy improves the navigation success rate when the learned traction models are deployed in environments unseen during training, both in simulations (see Section VIII) and in hardware experiments (see Section IX-B).

C. Evidential Deep Learning

While the traction predictor and the density estimator can be trained sequentially, Charpentier et al. [16] have shown that joint training using evidential deep learning can improve OOD detection performance while retaining prediction accuracy. In this section, we review the method and training loss proposed in [16], where NN outputs parameterize Dirichlet distributions (the conjugate priors of categorical distributions).

The Dirichlet distribution $q = \mathrm{Dir}(\boldsymbol{\beta})$ with concentration parameters $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_B]^\top \in \mathbb{R}^B_{>0}$ is a hierarchical distribution over categorical distributions $\mathrm{Cat}(\mathbf{p})$, where $\mathbf{p} \in \mathbb{R}^B_{\geq 0}$ is a normalized probability mass function (PMF) over $B > 0$ bins, i.e., $\sum_{b=1}^B p_b = 1$. The parameters $\mathbf{p}$ of the lower level categorical distribution $\mathrm{Cat}(\mathbf{p})$ are sampled from the higher level Dirichlet distribution, i.e., $\mathbf{p} \sim \mathrm{Dir}(\boldsymbol{\beta})$. The mean (also called the *expected PMF*) of the Dirichlet distribution is given by $\mathbb{E}_{\mathbf{p}\sim q}[\mathbf{p}] = \boldsymbol{\beta}/\sum_{b=1}^B \beta_b$. The expected PMF captures aleatoric uncertainty. The sum of the parameters $\boldsymbol{\beta}$, i.e., $\sum_{b=1}^B \beta_b$ represents how concentrated the Dirichlet distribution is around its mean. Therefore, $\sum_{b=1}^B \beta_b$ is also known as the concentration parameter, and corresponds to the "total evidence" of a data point observed in the training set. Higher evidence corresponds to lower epistemic uncertainty. Given a prior Dirichlet belief $\mathrm{Dir}(\boldsymbol{\beta}^{\text{prior}})$ and the input feature $\mathbf{o}$, the NN performs an input-dependent posterior update

$$\boldsymbol{\beta}^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}} = \boldsymbol{\beta}^{\text{prior}} + n^{\mathbf{o}}_{\boldsymbol{\lambda}} p_{\boldsymbol{\phi}}(\mathbf{o}) \tag{10}$$

$$n^{\mathbf{o}}_{\boldsymbol{\lambda}} = N p_{\boldsymbol{\lambda}}(\mathbf{z}_{\mathbf{o}}) \tag{11}$$

where the posterior Dirichlet distribution $q^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}} = \mathrm{Dir}(\boldsymbol{\beta}^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}})$ depends on the predicted traction $p_{\boldsymbol{\phi}}(\mathbf{o})$ (7) and the predicted evidence $n^{\mathbf{o}}_{\boldsymbol{\lambda}}$ that is proportional to the density for the latent feature $p_{\boldsymbol{\lambda}}(\mathbf{z}_{\mathbf{o}})$ (8) weighted by a fixed certainty budget $N > 0$. The posterior Dirichlet distribution leads to the expected traction PMF

$$\mathbf{p}^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}} = \frac{n^{\text{prior}} \mathbf{p}^{\text{prior}} + n^{\mathbf{o}}_{\boldsymbol{\lambda}} p_{\boldsymbol{\phi}}(\mathbf{o})}{n^{\text{prior}} + n^{\mathbf{o}}_{\boldsymbol{\lambda}}} \tag{12}$$

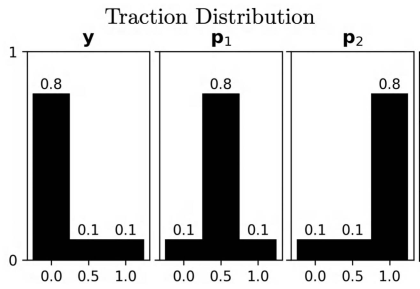
where $n^{\text{prior}} = \sum_{b=1}^B \beta_b^{\text{prior}}$ and $\mathbf{p}^{\text{prior}} = \boldsymbol{\beta}^{\text{prior}}/n^{\text{prior}}$. We use a flat prior by setting $\boldsymbol{\beta}^{\text{prior}} = \mathbf{1}_B$, where $\mathbf{1}_B \in \mathbb{R}^B$ is a vector of all ones, such that $\mathrm{Dir}(\boldsymbol{\beta}^{\text{prior}})$ is a uniform distribution over all PMFs. Based on this formulation proposed by [16], the posterior Dirichlet distribution $q^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}}$ and expected traction distribution $\mathbf{p}^{\mathbf{o}}_{\boldsymbol{\phi},\boldsymbol{\lambda}}$ both depend on the traction predictor, density estimator, and the input features. While the analysis of loss functions we perform below is for a generic Dirichlet distribution $q = \mathrm{Dir}(\boldsymbol{\beta})$ and PMF $\mathbf{p}$ for notational convenience, the posterior Dirichlet distribution and its expected PMF should be substituted by (10, 11, 12) during training.

Given the target PMF $\mathbf{y} \in \mathbb{R}^B_{\geq 0}$ that contains the empirically estimated traction distribution, the traction predictor and the normalizing flow can be trained jointly with the following UCE loss [16]

$$L^{\text{UCE}}(q, \mathbf{y}) - H(q) \tag{13}$$

where $L^{\text{UCE}}(q, \mathbf{y}) := \mathbb{E}_{\mathbf{p}\sim q}[-\sum_{b=1}^B y_b \log p_b]$ is defined as the expected CE loss and $H(q)$ is an entropy term that encourages smoothness of q. Note that both L^{UCE} and $H(q)$ depend on $\boldsymbol{\beta}$ (see Appendix A for details).

The ablation study in [16] has shown that training with (13) improves OOD detection performance while retaining similar accuracy achieved using the conventional CE loss. However, the key limitation of CE-based losses in our use case is that they treat the prediction errors across bins independently. The independence assumption is undesirable for learning traction, where bins are obtained by discretizing continuous traction values. These bins are ordered—bins closer to each other should be treated more similarly than bins far apart. We address this limitation by proposing a new loss function based on the squared EMD [19] that has been shown to achieve better accuracy than CE-based losses when bins are ordered.



Error Table

	$\mathbf{p}_1, \mathbf{y}$	$\mathbf{p}_2, \mathbf{y}$
EMD^2	0.49	0.98
CE	2.09	2.09

Fig. 6. Difference between EMD^2 and CE. Given the ground truth (GT) $\mathbf{y}$ and the predictions $\mathbf{p}_1$ and $\mathbf{p}_2$, CE produces the same values while EMD^2 penalizes $\mathbf{p}_2$ more. In practice, EMD^2 is more desirable because it accounts for the cross-bin relationship among the discretized traction values.

D. Uncertainty-Aware Squared Earth Mover's Distance

Intuitively the EMD between two distributions measures the minimum cost of transporting the probability mass of one distribution to the other, which has a closed-form solution for two categorical distributions defined by PMFs with the same number of bins [19]. Given a predicted PMF $\mathbf{p} \in \mathbb{R}^B_{\geq 0}$ and the target $\mathbf{y} \in \mathbb{R}^B_{\geq 0}$, the normalized EMD with l-norm for B equally spaced bins can be computed in closed-form [19]

$$\mathrm{EMD}(\mathbf{p}, \mathbf{y}) = \left(\frac{1}{B}\right)^{\frac{1}{l}} \|\mathrm{cs}(\mathbf{p}) - \mathrm{cs}(\mathbf{y})\|_l \tag{14}$$

where $\mathrm{cs} : \mathbb{R}^B \to \mathbb{R}^B$ is the cumulative sum operator. For convenience during training, we use $l = 2$ for Euclidean distance and optimize the squared EMD loss (EMD^2), dropping the constant factor. The toy example in Fig. 6 clearly shows that EMD^2 better captures the physical meaning of the predicted PMFs than CE, which ignores the relationship between bins.

As EMD^2 is only defined for PMFs, a naïve strategy is to compare the target $\mathbf{y}$ to the expected PMF from the predicted Dirichlet q, which leads to the following loss function (ignoring the constant multiplicative term):

$$\begin{aligned} L^{\text{EMD}^2}(q, \mathbf{y}) &:= \|\mathrm{cs}(\overline{\mathbf{p}}) - \mathrm{cs}(\mathbf{y})\|_2^2 \\ &= \mathrm{cs}(\overline{\mathbf{p}})^\top \mathrm{cs}(\overline{\mathbf{p}}) + \eta(q, \mathbf{y}) \end{aligned} \tag{15}$$

where $\overline{\mathbf{p}} := E_{\mathbf{p}\sim q}[\mathbf{p}] = \boldsymbol{\beta}/\beta_0$ is the expected PMF, $\beta_0 := \sum_{b=1}^B \beta_b$ is the total evidence and

$$\eta(q, \mathbf{y}) := -2\, \mathrm{cs}(\overline{\mathbf{p}})^\top \mathrm{cs}(\mathbf{y}) + \mathrm{cs}(\mathbf{y})^\top \mathrm{cs}(\mathbf{y}). \tag{16}$$

Note that $\mathrm{cs}(\overline{\mathbf{p}})$ can also be written as $\mathrm{cs}(\boldsymbol{\beta})/\beta_0$ due to the linearity of the cumulative sum operator. However, L^{EMD^2} is invariant to the total evidence β_0 of the Dirichlet distribution,

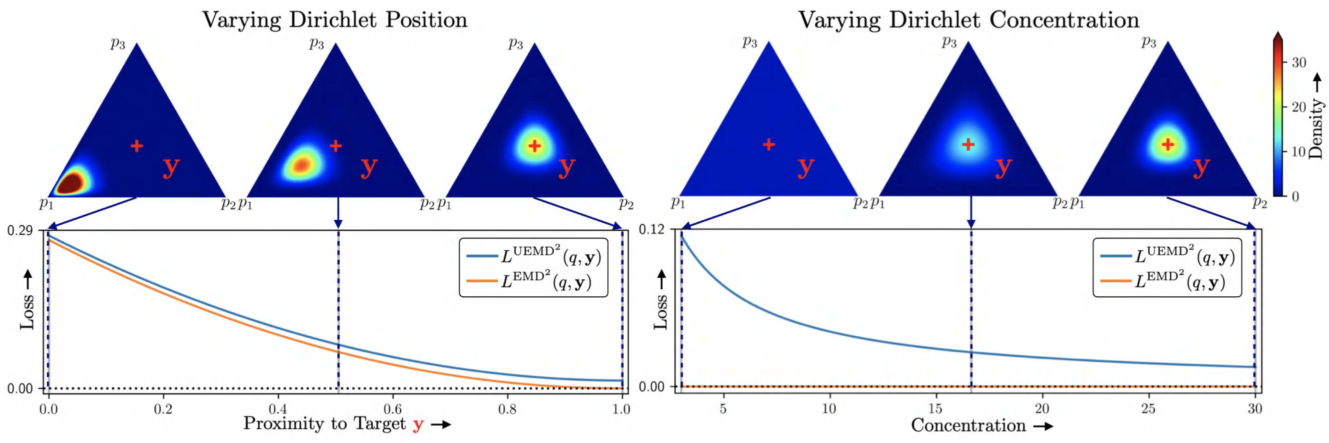


Fig. 7. Analyzing the difference between the standard EMD^2 loss and our proposed UEMD^2 loss on a toy example with three bins p_1, p_2, p_3. Each blue triangle represents a predicted Dirichlet distribution q visualized as a probability density over the 3-simplex; each point inside the simplex corresponds a categorical distribution over the three bins. The red cross $+$ denotes the location of the target label distribution $\mathbf{y}$ in the training set. A Dirichlet distribution can be parametrized by two quantities: the position of its mean and the concentration around its mean. *Left:* Varying the position of the Dirichlet while keeping its concentration fixed. In this case, both losses behave similarly and as desired—they encourage the predicted Dirichlet to be close to the target label distribution. *Right:* Varying the concentration of the Dirichlet while keeping its position fixed to the GT. Since EMD^2 only depends on the position of the Dirichlet mean, it is constant with respect to varying concentration. However, our proposed UEMD^2 encourages the predicted Dirichlet to have a high concentration (low epistemic uncertainty). Learning to predict low epistemic uncertainty for in-distribution training examples is essential for calibrated uncertainty prediction and detecting OOD examples, as opposed to being indifferent to the concentration.

as illustrated in the toy example in Fig. 7, so the epistemic uncertainty cannot be learned accurately.

Similar to the approach in [16] that uses the expectation of the CE loss that depends on $\boldsymbol{\beta}$, we propose the uncertainty-aware EMD^2 (UEMD^2) loss defined as the expectation of the EMD^2 given the Dirichlet q

$$L^{\mathrm{UEMD}^2}(q,\mathbf{y}) := \mathbb{E}_{\mathbf{p}\sim q}\left[\mathrm{EMD}^2(\mathbf{p},\mathbf{y})\right]. \tag{17}$$

The following theorem states that our proposed UEMD^2 loss can be computed in a closed form.

Theorem 1: Let $q = \mathrm{Dir}(\boldsymbol{\beta})$ be a Dirichlet distribution and let $\mathrm{Cat}(\mathbf{y})$ be a categorical distribution. Then, a closed-form expression exists for $L^{\mathrm{UEMD}^2}(q,\mathbf{y})$ given by

$$L^{\mathrm{UEMD}^2}(q,\mathbf{y}) = \mathrm{cs}(\overline{\mathbf{p}})^\top \frac{\mathrm{cs}(\boldsymbol{\beta}) + \mathbf{1}_B}{\beta_0 + 1} + \eta(q,\mathbf{y}) \tag{18}$$

where $\overline{\mathbf{p}} = \mathbb{E}_{\mathbf{p}\sim q}[\mathbf{p}]$, and η is defined in (16).

Proof: See Appendix B. □

Due to structural similarity to L^{EMD^2} (15), the proposed loss (18) also penalizes the EMD^2 error to encourage accurate traction predictions. In addition, the proposed loss penalizes low concentration $\boldsymbol{\beta}$ to encourage low epistemic uncertainty as shown in Fig. 7. In fact, it can be proved that L^{UEMD^2} is always greater or equal to L^{EMD^2} using Jensen's inequality and the convexity[1] of L^{EMD^2}. While (18) can be directly used as a loss function, EMD^2-based loss may not always converge to the desired local optima as observed by [19]. To address this issue, we follow [19] by considering a loss function that combines both EMD^2-based and CE-based loss terms. Therefore, we consider the following multiobjective optimization:

$$L^{\mathrm{UCE}}(q,\mathbf{y}) + w_1\, L^{\mathrm{UEMD}^2}(q,\mathbf{y}) - w_2 H(q) \tag{19}$$

[1]Taking cumulative sum and squared difference are convex operations.

where the entropy $H(q)$ encourages smoothness and the weights $w_1, w_2 \geq 0$ are hyperparameters. In practice, we compute (19) for the predicted linear and angular traction distributions separately and average the loss values. As simulation results suggest in Section V-C, the multiobjective loss (19) leads to more stable training and better generalization to test data.

IV. Planning With Learned Traction Distribution

While OOD terrain causing high epistemic uncertainty should be avoided, in-distribution terrain may still lead to high aleatoric uncertainty due to complex vehicle-terrain interactions. Therefore, we propose a risk-aware planner that tradeoffs the risk of immobilization with potential time savings from traversing terrain that leads to high aleatoric uncertainty.

A. Conditional Value at Risk

We adopt the CVaR as a risk metric because it satisfies a group of axioms important for rational risk assessment [46]. The conventional definition of CVaR assumes the worst-case occurs at the right tail of the distribution. We define CVaR for a random variable Z at level $\alpha \in (0, 1]$ for both the right and left tails of its distribution (see Fig. 8) as follows:

$$\mathrm{CVaR}_\alpha^{\rightarrow}(Z) := \frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\tau^{\rightarrow}(Z)\, d\tau \tag{20}$$

$$\mathrm{CVaR}_\alpha^{\leftarrow}(Z) := \frac{1}{\alpha}\int_0^\alpha \mathrm{VaR}_\tau^{\leftarrow}(Z)\, d\tau \tag{21}$$

where the right and left values at risk (VaR) are defined as

$$\mathrm{VaR}_\alpha^{\rightarrow}(Z) := \min\{z \mid p(Z > z) \leq \alpha\} \tag{22}$$

$$\mathrm{VaR}_\alpha^{\leftarrow}(Z) := \max\{z \mid p(Z < z) \leq \alpha\}. \tag{23}$$

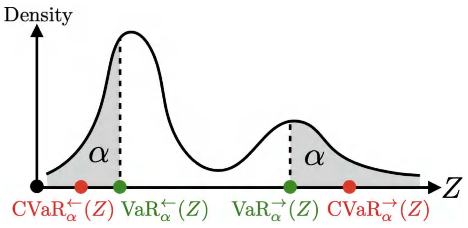


Fig. 8. This work defines two versions of CVaR to capture the worst-case expected values at either the left tail as $\text{CVaR}^{\leftarrow}_{\alpha}(Z)$ or the right tail as $\text{CVaR}^{\rightarrow}_{\alpha}(Z)$ for some random variable Z, where the worst-case scenarios constitute $\alpha \in (0, 1]$ portion of total probability. The left-tail and right-tail VaR are defined as $\text{VaR}^{\leftarrow}_{\alpha}(Z)$ and $\text{VaR}^{\rightarrow}_{\alpha}(Z)$.

Intuitively, $\text{CVaR}^{\rightarrow}_{\alpha}(Z)$ and $\text{CVaR}^{\leftarrow}_{\alpha}(Z)$ capture the expected outcomes that fall in the right tail and left tail of the distribution, respectively, where each tail occupies α portion of the total probability. Note that the right-tail definitions are suitable for costs to be minimized, and the left-tail definitions are suitable for low traction values. When $\alpha = 1$, either definition of CVaR is equivalent to the mean of the distribution $\mathbb{E}[Z]$.

B. Risk-Aware Planning

To account for the risk due to uncertain traction, we first present an existing approach [22] that optimizes for the right-tail CVaR of the planning objective (CVaR-Cost), and then propose a more computationally efficient method that accounts for the left-tail CVaR of traction (CVaR-Dyn). Lastly, we discuss the advantages and limitations of these two methods.

1) Worst-Case Expected Cost (CVaR-Cost [22]): Given the initial state $\mathbf{x}_0$, we want to find a control sequence $\mathbf{u}_{0:T-1}$ that minimizes the worst-case expected value of the nominal objective C (4) given uncertain terrain traction

$$\min_{\mathbf{u}_{0:T-1}} \quad \text{CVaR}^{\rightarrow}_{\alpha}(C(\widetilde{\mathbf{x}}_{0:T})) \tag{24}$$

$$\text{s.t.} \quad \widetilde{\mathbf{x}}_{t+1} = F(\widetilde{\mathbf{x}}_t, \mathbf{u}_t, \widetilde{\boldsymbol{\psi}}_t) \tag{25}$$

$$\widetilde{\boldsymbol{\psi}}_t \sim \text{Cat}(\mathbf{p}^{\widetilde{\mathbf{o}}_t}_{\boldsymbol{\phi},\boldsymbol{\lambda}}) \tag{26}$$

$$\widetilde{\mathbf{o}}_t \text{ is the terrain feature at } \widetilde{\mathbf{x}}_t \tag{27}$$

$$\widetilde{\mathbf{x}}_0 = \mathbf{x}_0 \quad \forall t \in \{0, \ldots, T-1\} \tag{28}$$

where traction $\widetilde{\boldsymbol{\psi}}_t$ is realized based on the predicted traction PMF $\mathbf{p}^{\widetilde{\mathbf{o}}_t}_{\boldsymbol{\phi},\boldsymbol{\lambda}}$ (12) after observing the terrain feature $\widetilde{\mathbf{o}}_t$. Due to the uncertain traction, the original objective $C(\widetilde{\mathbf{x}}_{0:T})$ is now a random variable that depends on the realization of the state trajectory. Note that this approach is inspired by [22], but we additionally handle terrain-dependent traction distributions.

In practice, optimizing (24) using MPPI requires a subroutine that empirically estimates the right-tail CVaR of the objective by collecting $M > 0$ realizations of the nominal objective $\{C(\widetilde{\mathbf{x}}^m_{0:T})\}^M_{m=1}$ for each candidate control sequence by using sampled traction values. To exploit GPU parallelization, we pregenerate $M > 0$ traction maps where each map cell contains sampled traction values. As a result, each candidate control sequence can be evaluated in parallel for all the pregenerated traction maps. While the sampled traction maps can be reused, the computation can still grow prohibitively as the map size grows. Therefore, we propose a cheaper cost design that accounts for the left-tail CVaR of terrain traction.

2) Worst-Case Expected Terrain Traction (CVaR-Dyn): Given the initial state $\mathbf{x}_0$, we want to find a control sequence $\mathbf{u}_{0:T-1}$ that minimizes the nominal objective C (4) using the state trajectory obtained with the worst-case expected traction

$$\min_{\mathbf{u}_{0:T-1}} \quad C(\bar{\mathbf{x}}_{0:T}) \tag{29}$$

$$\text{s.t.} \quad \bar{\mathbf{x}}_{t+1} = F(\bar{\mathbf{x}}_t, \mathbf{u}_t, \bar{\boldsymbol{\psi}}_t) \tag{30}$$

$$\bar{\boldsymbol{\psi}}_t = \begin{bmatrix} \text{CVaR}^{\leftarrow}_{\alpha}(\bar{\psi}_{1,t}) \\ \text{CVaR}^{\leftarrow}_{\alpha}(\bar{\psi}_{2,t}) \end{bmatrix}, \quad \begin{bmatrix} \bar{\psi}_{1,t} \\ \bar{\psi}_{2,t} \end{bmatrix} \sim \text{Cat}(\mathbf{p}^{\bar{\mathbf{o}}_t}_{\boldsymbol{\phi},\boldsymbol{\lambda}}) \tag{31}$$

$$\bar{\mathbf{o}}_t \text{ is the terrain feature at } \bar{\mathbf{x}}_t \tag{32}$$

$$\bar{\mathbf{x}}_0 = \mathbf{x}_0 \quad \forall t \in \{0, \ldots, T-1\} \tag{33}$$

where $\bar{\boldsymbol{\psi}}_t$ contains the left-tail CVaR of the linear and angular traction based on the predicted traction PMF $\mathbf{p}^{\bar{\mathbf{o}}_t}_{\boldsymbol{\phi},\boldsymbol{\lambda}}$ (12) after observing the terrain feature $\bar{\mathbf{o}}_t$. When $\alpha = 1$, the expected values of the traction parameters are used, equivalent to the planning approach used by [21].

3) Advantages and Limitations: Both CVaR-Cost and CVaR-Dyn leverage intuitive notions of risk based on the worst-case expected cost and traction, respectively. Moreover, they are simple to tune with a single risk parameter α regardless of the number of terrain types. Note that CVaR-Cost is a general algorithm that handles uncertainty in the planning problem [22]. In comparison, CVaR-Dyn is computationally cheaper, but it exploits the intuition that low traction usually worsens time-to-goal. However, such a relationship between system parameters and task performance may not hold for more complicated systems and different task objectives.

V. Evaluation of Traversability Learning Pipeline

The proposed evidential traversability learning method is benchmarked using a synthetic terrain dataset (Section V-A) designed to simulate data scarcity during real-world data collection and provide ground truth (GT) traction distributions and OOD terrain masks. Several variants of the proposed loss (19) are compared based on prediction accuracy and OOD detection performance (Section V-C). To highlight the benefits of joint training and our UEMD^2 loss (18), we provide an ablation study in Section V-D. After analyzing the proposed planner in Section VI, Section VII contains key results that show improved navigation performance due to our proposed loss.

As comparing uncertainty quantification methods is not the main focus of this work, we refer interested readers to [16] that has demonstrated the computational advantages, learning accuracy, and OOD detection performance of the NN architecture used in this work compared to the other state-of-the-art uncertainty quantification methods.

A. Synthetic 3-D Terrain Datasets

The synthetic dataset contains randomly generated 3-D terrain with GT traction distributions generated based on geometric

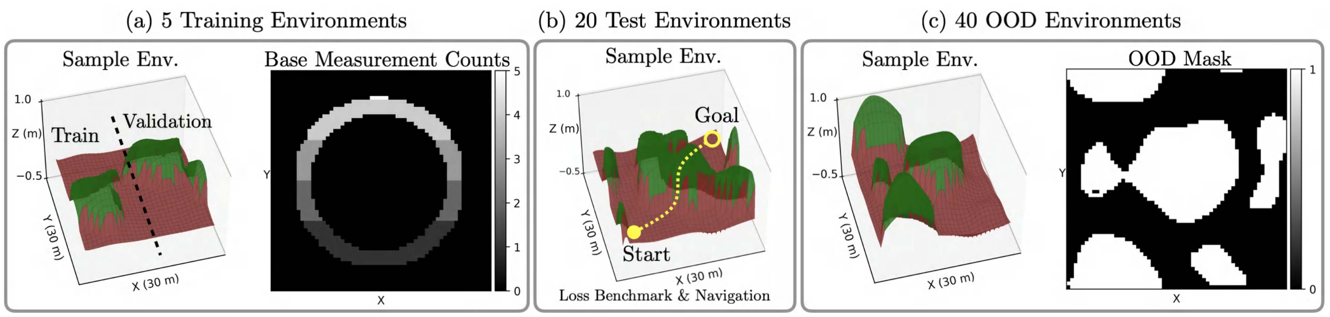
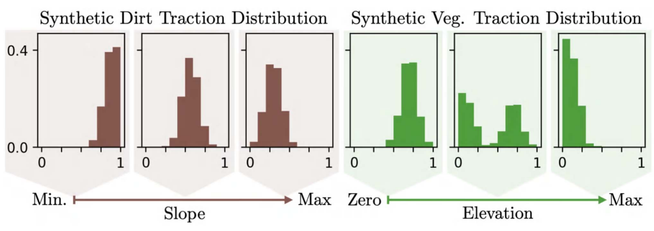


Fig. 9. Synthetic 3-D terrain dataset with dirt (brown) and vegetation (green) semantic types. (a) In each training environment, there are limited traction measurements along a prespecified circular path to mimic real-world data collection with limited coverage. Each environment is split into two for cross validation. Furthermore, we analyze the effect of a varying number of measurements by multiplying the base measurement counts (see Fig. 10). (b) Test environments contain novel terrain features for analyzing the traction prediction accuracy. To support the key argument that EMD^2 is a better indicator for navigation performance, models trained with different loss functions are deployed in the test environments for go-to-goal tasks (see Section VII). (c) Compared to the test environments, the OOD dataset additionally provides binary masks for the novel terrain with elevation and/or slope not observed during training. (a) Example training environment. (b) Example test environment. (c) Example OOD environment.

TABLE I
SYNTHETIC TERRAIN DATASET FOR BENCHMARKING LOSS FUNCTIONS



Dataset Types	Elevation Range (m)		Max Slope		Veg. Ratio	OOD Ratio
	Dirt	Veg.	Dirt	Veg.		
Train	-0.2–0.0	0.3–0.7	0.3	0.4	0.2	/
Test	-0.3–0.0	0.5–1.8	0.7	0.9	0.3	/
OOD (I)	-0.5–0.1	0.4–1.8	0.7	1.0	0.3	0.5
OOD (II)	-0.6–2.0	/	0.9	/	0.0	0.5

The GT traction distributions for dirt are unimodal Gaussian distributions whose mean increases with terrain slope that indicates roughness of the terrain. Traction distributions for vegetation are based on elevation, where the traction is bimodal for intermediate elevation and unimodal at the minimum and maximum elevations. Note that OOD dataset (I) consists of mixed terrain types, but OOD (II) contains no vegetation to ensure that learned models do not rely solely on semantics for OOD detection.

properties (elevation and slope) and semantic types (dirt and vegetation); details are available in Table I. Note that terrain slopes are only used for generating the GT traction distributions, but are not used as inputs to the NN. For simplicity, we use the same traction distribution for both linear and angular components, and dependencies only exist between dirt and terrain slope, and vegetation and terrain elevation. While more complex traction distributions can be designed, our dataset is sufficient for supporting our contributions.

In total, there are 5 training, 20 test, and 40 OOD environments that are 30 m in width and height, 0.5 m in resolution, as well as different elevations, slopes, and vegetation ratios, where the training dataset is intentionally small to examine the generalization of learned models. Every training environment is split into equal parts for training and cross validation, respectively. The synthetic environments are selectively visualized in Fig. 9. To simulate real-world data collection, traction samples are only obtained along a circular path. Moreover, we consider the impact of increasing the number of samples by multiplying the base measurement counts by factors 10^k where $k \in \{0, \ldots, 4\}$. For training environments, traction samples are accumulated in traversed terrain cells via histograms to obtain empirical traction distributions, and measurement counts are also stored to weight training loss in order to discount rarely visited terrain. In test environments, we use GT traction distributions to measure the prediction accuracy of trained models. In OOD environments, terrain with slope and elevation values unseen during training is deemed OOD. The associated OOD masks are the GT used to benchmark OOD detection performance. An example of the OOD masks is visualized in Fig. 9(c).

B. Model Training

We use the same network architecture for all the loss functions, where the traction predictor consists of a shared encoder (convolutional layers followed by fully connected layers) to process the semantic and elevation patches of the terrain, and two fully connected decoder heads with soft-max outputs for predicting the linear and angular traction distributions. The latent space features from the shared encoder are passed to a radial flow [53] and we use a constant certainty budget that scales exponentially with the latent dimension for numerical purposes [16]. During training, we follow the two-step procedure outlined in [16]. First, we jointly train the traction distribution predictor and the flow network. After convergence, we freeze the traction predictor and only fine-tune the flow network. This strategy improves OOD detection accuracy. However, we observe no improvement by performing "warm-up training" for the flow network prescribed by [16].

Hyperparameter sweeps are conducted over learning rates in $[1\text{e}{-4}, 3\text{e}{-4}, 1\text{e}{-3}]$ for the Adam optimizer, and entropy weights in $[0, 1\text{e}{-6}, 1\text{e}{-5}]$ when UEMD^2 and UCE are used separately. For the weighted sum of UEMD^2 and UCE, we fix the UCE term and consider additional weights for UEMD^2 in $[0.1, 1, 10]$. For each combination of hyperparameters, we train the model with five random seeds and select the best model based on validation EMD^2 error averaged over the seeds, because empirically we have found that selecting models

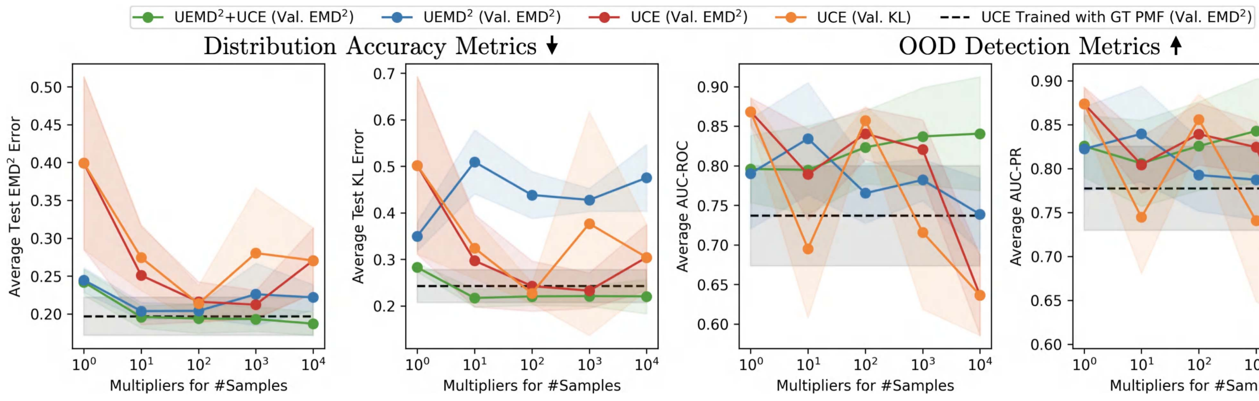


Fig. 10. Prediction errors measured in EMD^2 and KL divergence (the lower, the better), and OOD detection accuracy measured in AUC-ROC and AUC-PR (the higher, the better). The legend for each loss function is followed in parentheses with the selection criteria used for choosing hyperparameters. The results show the average and standard deviations. Overall, the proposed weighted sum of UEMD^2 and UCE leads to the best prediction accuracy and steadily improving OOD detection performance when given more training samples. Due to the distribution shift between the training and test environments, too much training data lead to degrading prediction accuracy for the other loss designs. In addition, compared to EMD^2-based losses, UCE is worse at capturing the cross-bin relationship among the discrete traction values, resulting in worse prediction accuracy and unstable OOD detection performance.

based on validation EMD^2 instead of Kullback–Leibler (KL) divergence leads to improved performance for *all* models. To guarantee fairness for the state-of-the-art and not clutter the figures, we only present the results for models selected based on validation KL divergence for the UCE loss.

TABLE II
ABLATION STUDY FOR UEMD^2 AND JOINT TRAINING

Loss	Test EMD^2 ↓	AUC-ROC ↑	AUC-PR ↑
UEMD^2 (Joint)	$\mathbf{0.204 \pm 0.01}$	$\mathbf{0.834 \pm 0.07}$	$\mathbf{0.840 \pm 0.05}$
EMD^2 (Joint)	0.236 ± 0.02	0.802 ± 0.04	0.830 ± 0.03
EMD^2 (Disjoint)	0.228 ± 0.03	0.665 ± 0.14	0.770 ± 0.07

Results shown are the mean and standard deviation values obtained across random seeds. Best values are marked in bold.

C. Prediction Accuracy and OOD Detection Performance

Variations of the proposed loss function (19) are compared in terms of prediction accuracy and OOD detection performance. The prediction accuracy is measured by EMD^2 and KL divergence by comparing the predicted and the GT traction distributions. The accuracy of OOD detection using the densities of latent features is measured by the area under the receiver operating characteristic curve (AUC-ROC) and the area under the precision-recall curve (AUC-PR) with respect to the GT OOD masks. Note that AUC-ROC and AUC-PR are standard metrics for binary classification that are invariant to scale and offset. Intuitively, a score of 0.5 means the classifier is as good as random guesses, and a score of 1 indicates a perfect classifier. To show the best performance achievable by the state-of-the-art with unlimited traction samples during training, we include models trained with UCE using the *GT traction distributions* in the training environments. The benchmark results are in Fig. 10, where we report the average values and the standard deviations over all map cells, test environments, and random seeds.

The main takeaway from the benchmark is that the models trained with the proposed weighted sum of UEMD^2 and UCE achieve the best prediction accuracy in *both* EMD^2 and KL divergence. Furthermore, the weighted-sum objective leads to more stable improvements in test performance for both prediction accuracy and OOD detection as training samples become more abundant. Interestingly, too many training samples lead to degrading prediction accuracy achieved by the other loss designs at test time. Because we do not observe worsening accuracy on the validation dataset, the degrading test performance can be attributed to the distribution shift between the training and test environments as shown in Table I. Notably, compared to EMD^2-based losses, UCE does not capture the cross-bin relationship of the traction distribution, which leads to worse regularized latent space that causes unstable OOD detection performance (even for UCE trained with GT traction distributions in the training environments).

D. Ablation Study for UEMD^2 and Joint Training

While the benefits of using uncertainty-aware loss and joint training have been established in [16] for UCE, we present a similar ablation study for UEMD^2 for completeness in Table II. We set the sample multiplier to 10 for simplicity, but similar conclusions can be drawn with more samples. The takeaway is that both joint training and uncertainty awareness are required to achieve improved accuracy in EMD^2 and OOD detection. Despite these improvements, the results in Fig. 10 show that both UEMD^2 and UCE are required to achieve more consistent and steadily improving performance in prediction accuracy and OOD detection performance.

VI. Evaluation of Risk-Aware Planners

Using simulated 2-D semantic environments whose terrain traction has high aleatoric uncertainty, we show that the proposed CVaR-Dyn outperforms existing approaches [11], [21] that assume the nominal traction or the expected traction, while achieving competitive performance compared to CVaR-Cost [22]. Moreover, we discuss the advantages and limitations

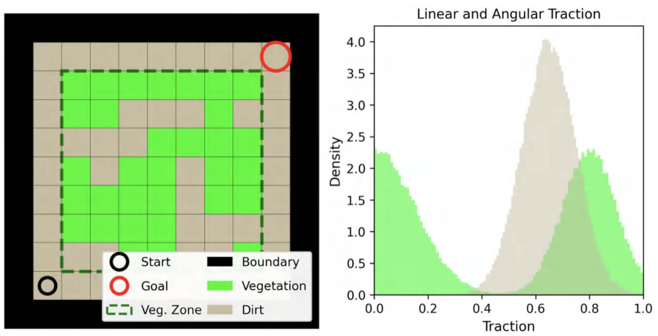


Fig. 11. Simulation environment where a robot has to move from start to goal as fast as possible within the bounded arena. Linear and angular traction parameters share the same distribution for simplicity. Vegetation terrain patches are randomly sampled at the center of the vegetation zone.

of CVaR-Dyn and CVaR-Cost compared to the approach that assumes nominal traction while penalizing trajectories moving through terrain with high aleatoric uncertainty. For simplicity, we consider a grid world where dirt and vegetation cells have known traction distributions, as shown in Fig. 11. Vegetation cells are randomly spawned with increasing probabilities at the center of the arena, and a robot may get stuck due to vegetation's bi-modal traction distribution. The mission is successful if the robot reaches the goal without encountering zero-traction regions, colliding with obstacles, or getting stuck in local minima (e.g., when the robot does not move or just repeats circular trajectories without progressing to the goal).

A. Planner Implementation

We adopt MPPI [18, Algorithm 2] because it is derivative-free and parallelizable on GPU. The planners run in a receding horizon fashion with 100 timesteps at 0.1 s intervals. The maximum linear and angular speeds are 3 m/s and π rad/s, and the noise standard deviations for the control signals are 2 m/s and 2 rad/s. The number of control rollouts is 1024, and the number of sampled traction maps is 1024 (only for CVaR-Cost). We use PMFs with 20 uniform bins to approximate the traction distribution. A computer with Intel Core i9 CPU and Nvidia GeForce RTX 3070 GPU is used for the simulations, where the majority of the computation happens on the GPU. The CVaR-Cost planner is the most expensive to compute, but it is able to replan at 15 Hz while sampling new control actions and maps with dimensions of 200×200. Planners that do not sample traction maps can be executed at over 50 Hz.

B. Navigation Performance

We compare the proposed CVaR-Dyn against CVaR-Cost [22], WayFAST [21] that uses the expected traction, and the technique in [11] that assumes the nominal traction while adjusting the time cost with the CVaR of linear traction. Note that WayFAST is a vision-based navigation approach that predicts the expected terrain traction from images, but our analysis only focuses on the use of the expected traction values and its impact on the navigation performance. We vary the conservativeness of all the methods (other than WayFAST) by changing the quantile $\alpha \in (0, 1]$ for computing the CVaR. Overall, we sample 40 different semantic maps and five random realizations of traction parameters for every semantic map. The traction parameters are drawn before starting each trial and remain fixed. The benchmark results can be found in Fig. 12. The takeaway is that the proposed CVaR-Dyn achieves better or similar success rate and time-to-goal when compared to CVaR-Cost if the risk tolerance α is sufficiently low. In addition, both CVaR-Dyn and CVaR-Cost outperform the other methods that use the nominal or the expected traction values.

To compare CVaR-based methods against another baseline that plans with the nominal traction while imposing an auxiliary penalties for vegetation terrain with high aleatoric uncertainty, we focus on the most challenging setting with 70% vegetation, where it is easy to get stuck in local minima. To adjust risk tolerance, we consider $\alpha \in (0, 1]$ for CVaR-based methods and vegetation penalty $w \geq 0$ for the planner that assumes nominal traction. The benchmark result in Fig. 13 shows the tradeoffs between success rate and time-to-goal achieved by different methods (WayFAST included as a special case of CVaR-Dyn when $\alpha = 1$). Overall, all methods except WayFAST can be tuned to improve success rate and time-to-goal. Assigning high vegetation penalties leads to the best success rate, because the robot always avoids the vegetation terrain. On the other hand, CVaR-Dyn and CVaR-Cost can achieve better time-to-goal at a lower success rate, which may be desirable for more risk-tolerant and time-critical missions. As α decreases further, CVaR-Dyn's performance first plateaus and then worsens, because the state rollouts become too short when using the worst-case expected traction, making CVaR-Dyn susceptible to local minima. While CVaR-Cost also experiences worsening performance as α decreases, its conservativeness is caused by the greater difficulty of estimating CVaR of objective. Note that CVaR-Cost takes about 60 ms on average to produce a solution, which is more computationally expensive than the other methods, which only take about 5 ms. Overall, none of the methods completely dominates the others. Therefore, when domain knowledge is available, auxiliary penalties for undesirable terrain can be used together with CVaR-based methods to improve performance (see Section VIII). While the simulation shows comparable performance achieved by CVaR-Dyn, CVaR-Cost, and the baseline, we show that CVaR-Dyn achieves the best performance in practice (see Section IX).

VII. Optimizing for EMD Improves Navigation

To support the key argument that EMD^2 is a better metric than KL divergence for measuring the quality of learned traction distributions for navigation, we evaluate the navigation performances when using models trained with different losses presented in Section V. The models are deployed in the same test environments visualized in Fig. 9, where each map is 30 m in width and height and the start and goal positions are at the opposite diagonal corners. To not clutter the results, we only focus on the proposed CVaR-Dyn planner with $\alpha = 0.4$ and the same MPPI setup in Section VI-A, but similar trends can be observed with different choices of α. Consistent with the loss

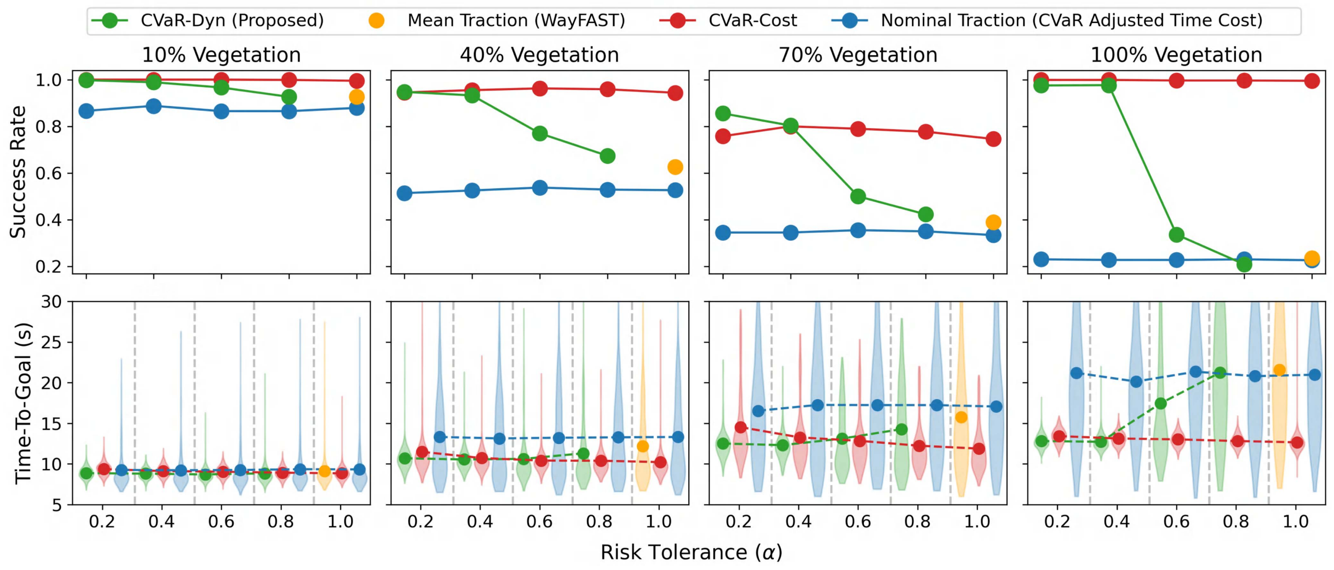


Fig. 12. Success rates and time-to-goal achieved by the proposed CVaR-Dyn, CVaR-Cost [22], WayFAST [21] that use the expected traction and the method that assumes nominal traction [11] (i.e., no slip). Note that a mission is successful if the robot reaches the goal. We show the distributions of time-to-goal and their average values. Overall, when the risk tolerance is sufficiently low (e.g., $\alpha = 0.4$), CVaR-Dyn achieves a similar or better success rate and time-to-goal compared to the CVaR-Cost planner and outperforms both WayFAST and the method that assumes nominal traction.

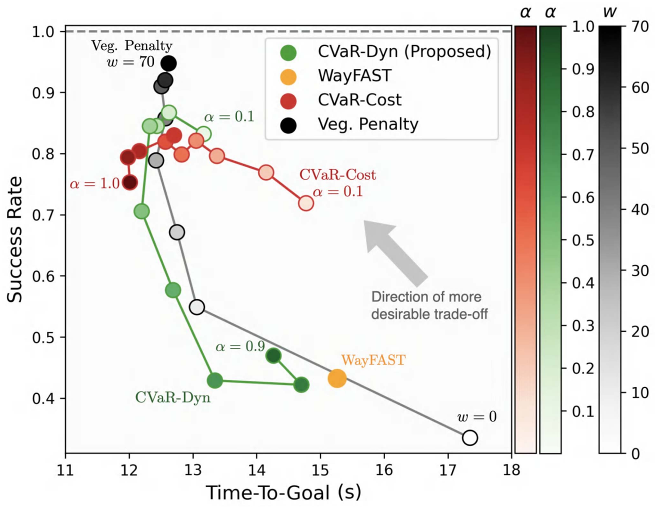


Fig. 13. Tradeoffs between success rate and time-to-goal in the most challenging scenario of 70% vegetation, where success is achieved if the goal is reached. CVaR-Dyn and CVaR-Cost both achieve better tradeoffs than WayFAST by being in the upper left of the figure. When the success rate is below 0.9, CVaR-Dyn and CVaR-Cost achieve better tradeoffs than the method that assumes nominal traction while imposing auxiliary penalty $w > 0$ for states entering vegetation terrain. However, the success rates of CVaR-Dyn and CVaR-Cost plateau and eventually degrade as α decreases, because the planners become more risk-averse and susceptible to local minima.

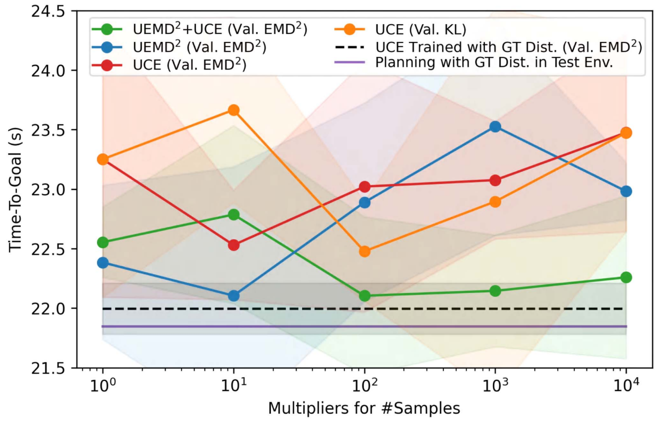


Fig. 14. Navigation performance using learned traction models trained with different loss designs in the test environments shown in Fig. 9. The results show the average values and the standard deviations over all test environments, sampled traction maps, and random seeds. Note that the navigation performance of the proposed hybrid loss approaches the best possible navigation performance using the GT traction models in the test environments and the best possible navigation performance of the state-of-the-art UCE loss trained with the GT traction distributions in the training environments.

benchmark in Section V, each loss is trained with five random seeds and five levels of data abundance. For each of the 20 test maps, we consider five randomly sampled traction maps and run the mission three times. The final results averaged over training seeds can be found in Fig. 14, where all trials are successful, so the success rate is omitted.

Importantly, when the amount of training data is low, UEMD^2 outperforms UCE in time-to-goal, even though UEMD^2 leads to worse KL error than UCE loss as shown in Fig. 10. This validates our intuition that EMD^2 captures the cross-bin information of discretized traction values better, which facilitates the learning of traction distribution in low-data regime and leads to better navigation performance. As more training data becomes available, the proposed weighted sum of UEMD^2 and UCE outperforms the other loss designs. Due to the distribution shift between the training and test environments as discussed in Section V-C, the traction prediction accuracy may degrade when given too much training data, resulting in degrading navigation performance observed in Fig. 14. However, the proposed hybrid loss is less susceptible to the distribution shift, thus sustaining the

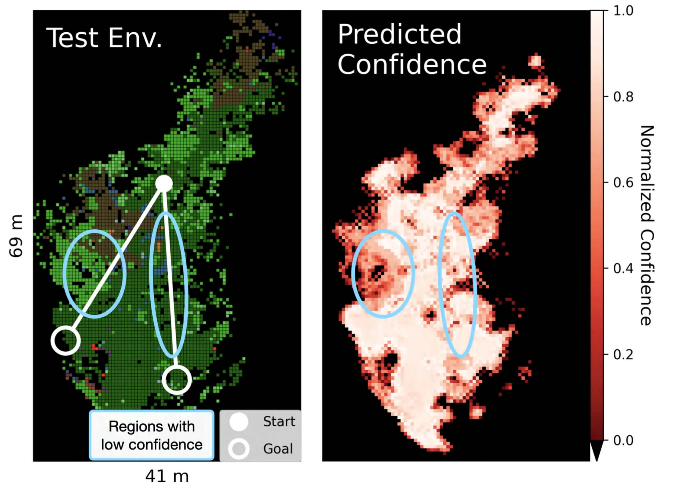


Fig. 15. (Left) In the test environment, the simulated robot has to reach two goals selected to highlight the danger of using unreliable network predictions. (Right) The latent density-based confidence score (9) indicates the amount of epistemic uncertainty for the predicted traction distribution, where unknown terrain and known terrain with negative scores are shown in black. Note that the brown semantic region (mulch) at the top has confidence below zero due to the presence of unknown cells, in contrast to the brown semantic region to the left with much fewer unknown cells.

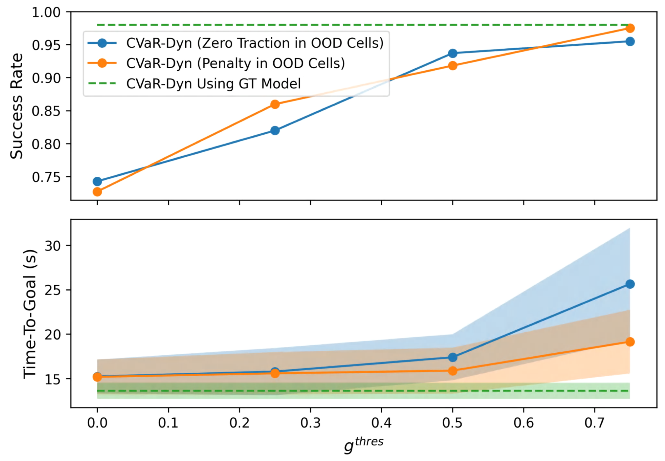


Fig. 16. Navigation success rate improves by avoiding OOD terrain. Note that the shaded areas indicate standard deviations. The OOD terrain is handled by either assigning zero traction (blue) or imposing penalties (orange). The performance of the planner that uses the GT traction is included to show the best possible performance. Overall, higher g^{thres} improves the success rate at the cost of worse time-to-goal. However, auxiliary penalties for OOD terrain make it easier for the planner to find solutions that lead to the goal. Notably, the average success rate when $g^{\text{thres}} = 0.75$ approaches 1, indicating that the learned traction model generalizes well to terrain with high confidence values (low epistemic uncertainty) in the test environment.

navigation performance better than the other methods. Furthermore, the navigation performance of the proposed hybrid loss approaches the best possible performance of the state-of-the-art UCE loss when trained on GT traction distributions in the training environments, indicating a good generalization of our approach using only the limited data collected along circular paths in the training environments. For reference, the figure also provides the lower bound for the time-to-goal based on the GT traction models in the test environments.

VIII. Benefits of Avoiding OOD Terrain

We demonstrate the benefits of the proposed density-based confidence score (9) for detecting terrain with high epistemic uncertainty. To simulate training and test environments, we leverage the data collected in two distinct forests using Clearpath Husky, where the first one (shown in Fig. 5) is used for training, and the second one (whose semantic top–down view is shown in Fig. 15) is used as the test environment. The environment models are built using semantic octomaps [50] that fuse lidar points and segmented RGB images based on the 24 semantic categories in the RUGD dataset [52]. The traction values are drawn from the test environment's empirical traction distributions learned by a separate NN as the proxy GT. We use the proposed CVaR-Dyn with a low risk tolerance $\alpha = 0.2$ to handle the noisy terrain traction. Two specific start-goal pairs are selected to highlight the most challenging parts of the test environment with novel features. Each start-goal pair is repeated ten times for each selected confidence threshold g^{thres}. We investigate two ways to prevent the planner from entering terrain that is classified as OOD by either assigning zero traction, or adding large penalties. The mission is deemed successful if the goal is reached.

As shown in Fig. 16, the success rate improves by up to 30% as the confidence threshold g^{thres} increases, because the robot avoids regions with unreliable traction predictions. Interestingly, using CVaR-Dyn with soft penalties for OOD terrain leads to better time-to-goal while retaining a similar success rate, because the auxiliary costs for OOD terrain make it easier for the planner to find trajectories that avoid the OOD terrain. Therefore, it is advantageous to use CVaR-Dyn with auxiliary costs when domain knowledge is available to achieve both a high success rate and fast navigation in practice.

IX. Hardware Experiments

To evaluate the effectiveness and feasibility of EVORA (the overall framework for uncertainty-aware traversability learning and risk-aware planning) in practice, we design two experiment scenarios—an indoor race track scenario with fake vegetation using an RC car (Section IX-A) and a more challenging outdoor scenario using a legged robot (Section IX-B). For both scenarios, the robots use onboard sensors to map the environments online at test time, introducing more uncertainty due to motion blurs, lighting changes, and incomplete maps. While both scenarios show that the proposed CVaR-Dyn planner leads to the best navigation performance, the outdoor scenario also shows the benefits of avoiding OOD terrain. In practice, the control signals generated by MPPI are very noisy, so we plan in the derivative space of the nominal control [54] to generate smooth trajectories.

A. Indoor Racing With an RC Car

The goal of the indoor experiments is to show the performance benefits of the proposed planner for mitigating the risk due to aleatoric uncertainty in a controlled environment.

1) Experiment Setup: An overview of the indoor setup is provided in Fig. 17, which shows the 9.6 m by 8 m for consistency with 0.33 m by 0.25 m RC car arena populated with turf and fake trees used to mimic outdoor vegetation. The 0.33 m by 0.25 m RC car is equipped with a RealSense D455 depth camera, an

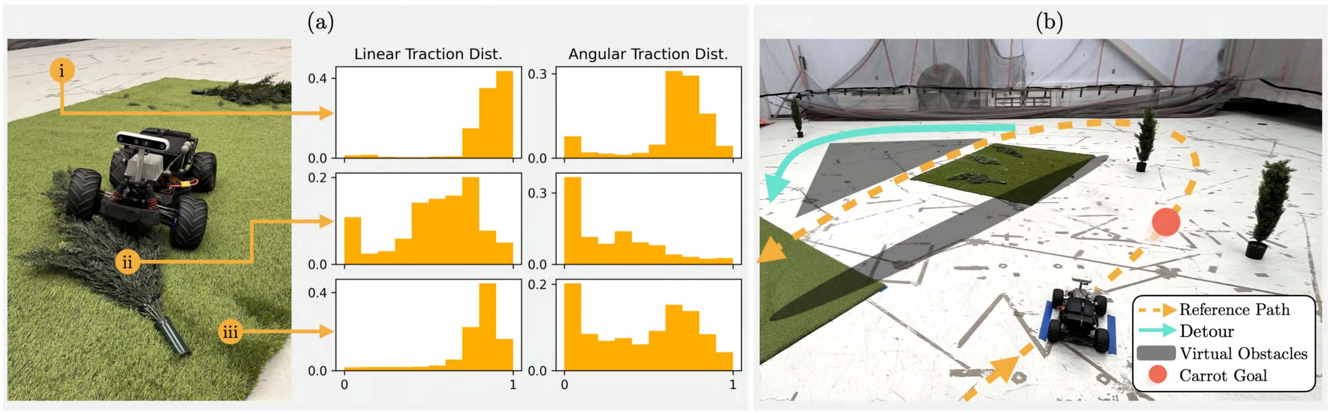


Fig. 17. Training and test environments used for the indoor racing experiments. (a) Training environment consists of a single turf with two fallen trees for simulating bushes. Learned linear and angular traction distributions are visualized for selected regions with (i) hard floor, (ii) fallen tree, and (iii) turf. Note that the bimodality of traction distribution over the vegetation may cause the robot to slow down significantly. (b) Test environment contains two turfs, three fallen trees, three standing trees, and virtual obstacles. The robot is tasked to drive two laps following a carrot goal along the reference path while deciding between a detour without vegetation and a shorter path with vegetation. (a) Learned traction in training environment. (b) Test environment.

Intel Core i7 CPU, and an Nvidia RTX 2060 GPU. The robot runs onboard traction prediction, motion planning, and online elevation mapping with 0.1 m resolution, but Vicon is used for pose and velocity estimation. The robot identifies vegetation by extracting green image pixels instead of using a standalone NN semantic classifier to conserve GPU resources. The bicycle model (3) is used for this experiment, and the traction values are obtained by analyzing the commanded linear velocities, steering angles, and the GT velocities from Vicon.

The traction model is trained based on 10 min of driving data with the proposed loss function (19), where UEMD^2 and UCE are both weighted by 1 and the entropy term is weighted by $1\text{e}{-5}$ based on empirical tuning. The learned traction distributions are visualized in Fig. 17(a) to highlight multimodality. At deployment time, the robot runs two laps around the race track along the ellipsoidal reference path, while deciding between a shorter path covered with vegetation or a less risky detour, as shown in Fig. 17(b). We design a moving goal region along the reference path, called the “carrot goal,” that maintains a constant 75° offset from the robot’s projected position on the ellipsoidal reference path. In addition to CVaR-Cost and the proposed CVaR-Dyn, we consider an intelligent baseline that assumes nominal traction but assigns auxiliary penalties for low-lying vegetation between 5 cm and 15 cm that could cause unfavorable driving conditions. All methods avoid the trees via auxiliary penalties. All planners consider 1024 rollouts while planning at 20 Hz with 5 s of look-ahead. Due to computational constraints, CVaR-Cost only considers 400 traction map samples. We set the maximum linear speed and steering angle to be 1.5 m/s and 30°.

2) Aleatoric Uncertainty Results: The qualitative and quantitative results comparing planners’ abilities to mitigate the risk due to aleatoric uncertainty are summarized in Figs. 18 and 19. We consider three risk tolerances $\alpha \in \{0.6, 0.8, 1\}$ for CVaR-Dyn, CVaR-Cost, and vegetation penalties $w \in \{10, 20, 100\}$ for the baseline that assumes nominal traction while penalizing states entering vegetation terrain. We present results for WayFAST [21] separately, but it is a special case of CVaR-Dyn when $\alpha = 1$. We repeat the race five times and each race consists of two laps. Overall, CVaR-Dyn with $\alpha = 0.8$ achieves the best time-to-goal and success rate. Qualitative visualizations in Fig. 18 show that the baseline and WayFAST both suffer from noisy real-world traction, causing wide turns. In comparison, CVaR-Cost and the proposed CVaR-Dyn handle the noisy terrain traction better by producing smoother trajectories. Different from CVaR-Dyn, the CVaR-Cost planner more frequently takes the detour and sometimes gets stuck in local minima near obstacles.

B. Outdoor Navigation With a Legged Robot

Compared to the indoor setting, the outdoor experiments introduce more diverse terrain types and uncertainty in perception due to lighting changes and rough motions. In addition to benchmarking the planners’ abilities to handle aleatoric uncertainty, the outdoor tests also demonstrate the benefits of mitigating epistemic uncertainty by avoiding OOD terrain, as well as the applicability of our approach on a legged robot.

1) Experiment Setup: An overview of the outdoor setup is shown in Fig. 20. A Boston Dynamics Spot robot is fitted with a RealSense D455, an Ouster OS0 lidar, and an Nvidia Jetson AGX Orin with good power efficiency but less powerful computation than the computers used in previous experiments. The unicycle model (2) is used for this experiment, and traction values are obtained by comparing the commanded velocities and Spot’s built-in odometry. The environment model is built using a semantic octomap [50] with 0.2 m resolution that fuses lidar points and segmented RGB images based on the 24 semantic categories in the RUGD dataset [52]. The traction model is trained based on 5 min of walking data with the proposed loss function (19) with the same weights used for the indoor experiment. The learned traction distributions are selectively visualized in Fig. 20(a) to highlight multimodality. As shown in Fig. 20(b), we choose 2 start-goal pairs for testing the planners and assessing the benefits of avoiding OOD terrain, respectively.

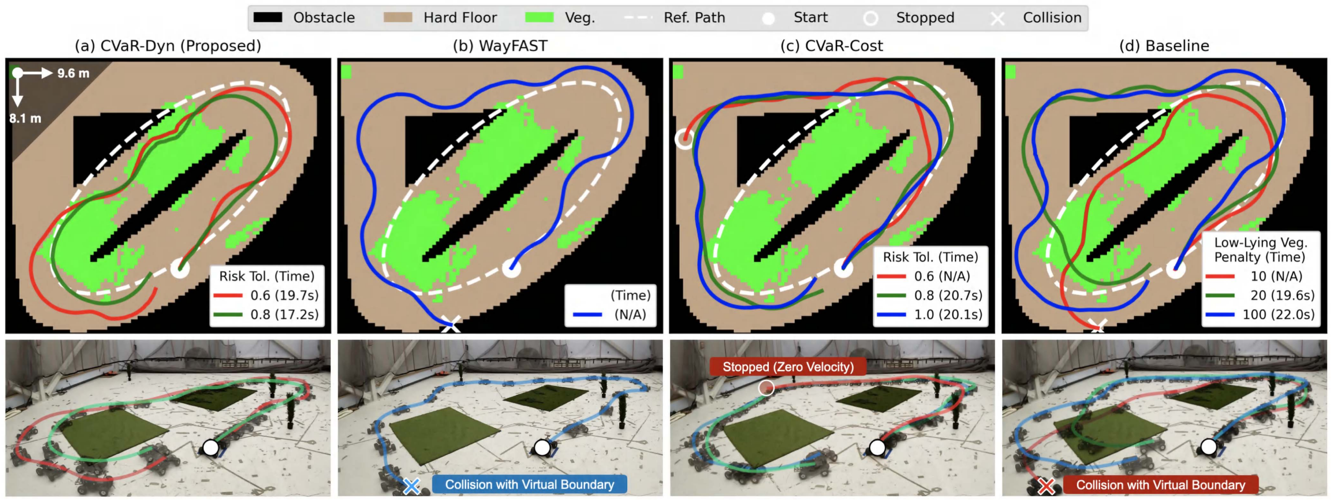


Fig. 18. Representative trials of the indoor experiments for highlighting the failure modes of the planners. The top-down semantic maps are shown in the top row and the time-lapse photos are shown in the bottom row. We only show the first lap out of the two laps for clarity. (a) As α decreases, the proposed CVaR-Dyn becomes more risk-averse and takes wider turns in order to enter the shortcut. (b) WayFAST (CVaR-Dyn with $\alpha = 1$) does not account for the risk of under-steering, so it always turns too late for the shortcut. (c) CVaR-Cost consistently takes the detour to avoid the vegetation terrain. As α decreases, the planner becomes more risk-averse and sometimes stops near obstacles. (d) When the soft penalty is low, the baseline is more risk-tolerant and takes the shortcut, but the experienced traction differs significantly from the nominal traction, causing more collisions. As the soft penalty increases, the planner becomes more conservative and takes the detour, but planning with nominal traction leads to significant understeering that limits performance.

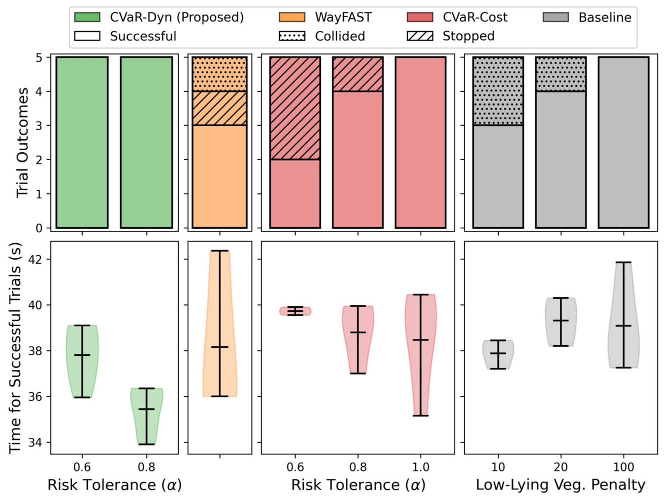


Fig. 19. Outcomes and mission time for the indoor experiments over 5 trials. We show the distributions of mission time and the maximum, average, and minimum values. The proposed CVaR-Dyn with $\alpha = 0.8$ achieves the best time-to-goal with 100% success rate. As α reduces, CVaR-Dyn and CVaR-Cost both lead to worse time-to-goal. Notice that CVaR-Cost stops near obstacles in many occasions when $\alpha < 1$. In comparison, the baseline and WayFAST lead to worse time-to-goal and a higher chance of collision.

All planners avoid the terrain with elevation greater than 1.4 m via auxiliary penalties, and the baseline assigns soft costs for the grass and bush semantic types with elevations less than 1.4 m. While the 1.4 m height threshold is much higher than the robot's step height, the selected test environments do not have short and rigid obstacles in order to analyze the planners' ability to handle tall vegetation. Due to limited GPU resources shared by semantic classification, traction prediction, and motion planning, the planners can only reliably plan at 5 Hz with 8 s of look-ahead and 800 control rollouts, and CVaR-Cost is only allowed 200 traction map samples. The maximum linear and angular velocities are 1 m/s and 90°/s.

2) Aleatoric Uncertainty Results: The qualitative and quantitative results comparing planners' abilities to mitigate the risk due to aleatoric uncertainty are shown in Figs. 21 and 22. We consider three round trips to and from the goal (six trials in total) for each method. Overall, CVaR-Dyn with $\alpha = 0.9$ achieves the best time-to-goal and success rate, consistent with the indoor experiments in Section IX-A. The CVaR-Cost planner is more conservative by staying far from the bushes. In comparison, the baseline and WayFAST both suffer from noisy real-world traction, causing wide turns. Notably, when the soft penalties for grass and bush semantic types are too high, the baseline planner gets stuck in local minima, thus requiring human interventions and long mission time.

3) Epistemic Uncertainty Results: Different from previous experiments, the goal of the OOD terrain avoidance experiment is to show the benefit of mitigating the risk due to epistemic uncertainty. Therefore, we only use the proposed planner CVaR-Dyn with $\alpha = 0.9$, but similar conclusions still hold if we change the underlying local planner to CVaR-Cost or another baseline method to mitigate the risk due to aleatoric uncertainty. We execute three round trips in total.

The qualitative and quantitative results for the OOD avoidance experiments are shown in Figs. 23 and 24. We consider the terrain as OOD if the normalized densities for the traction predictor's latent features are below 0 (i.e., the 0th percentile of the densities observed for all the training data), but a more conservative threshold may be used based on empirical tuning. Compared to the training environment shown in Fig. 20, the test environment shown in Fig. 23 contains much taller vegetation that is not in the training dataset. As a result, the traction predictions for

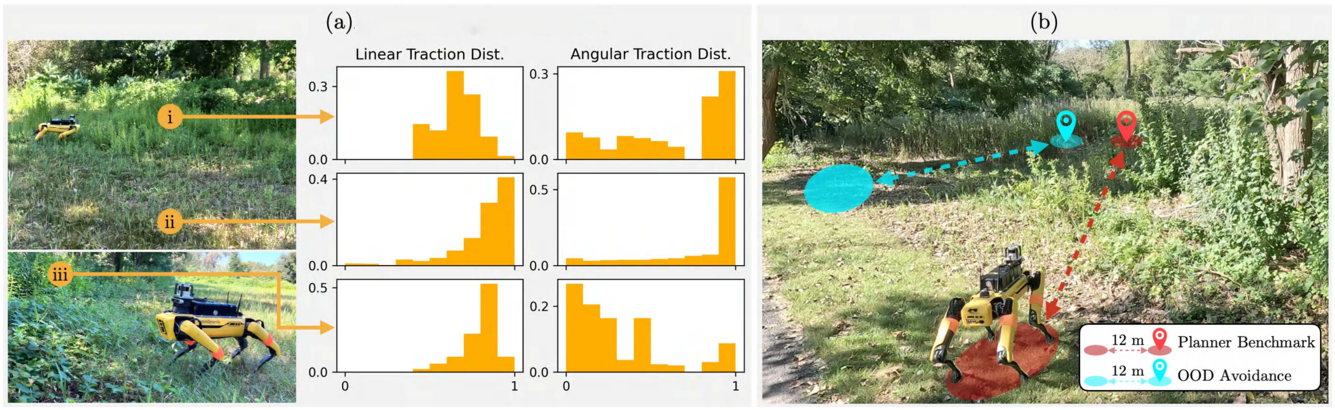


Fig. 20. Outdoor training and test environments with a legged robot. (a) Outdoor environment consists of vegetation terrain with different heights and densities. Predicted linear and angular traction distributions are visualized for selected regions with (i) tall grass, (ii) short grass, and (iii) dense bushes. Unlike wheeled robots, a legged robot typically has good linear traction through vegetation, but angular traction may exhibit multimodality due to the greater difficulty of turning. (b) Two start-goal pairs are used to benchmark the planners and analyze the benefits of avoiding OOD terrain. (a) Learned traction in training environment. (b) Test environment.

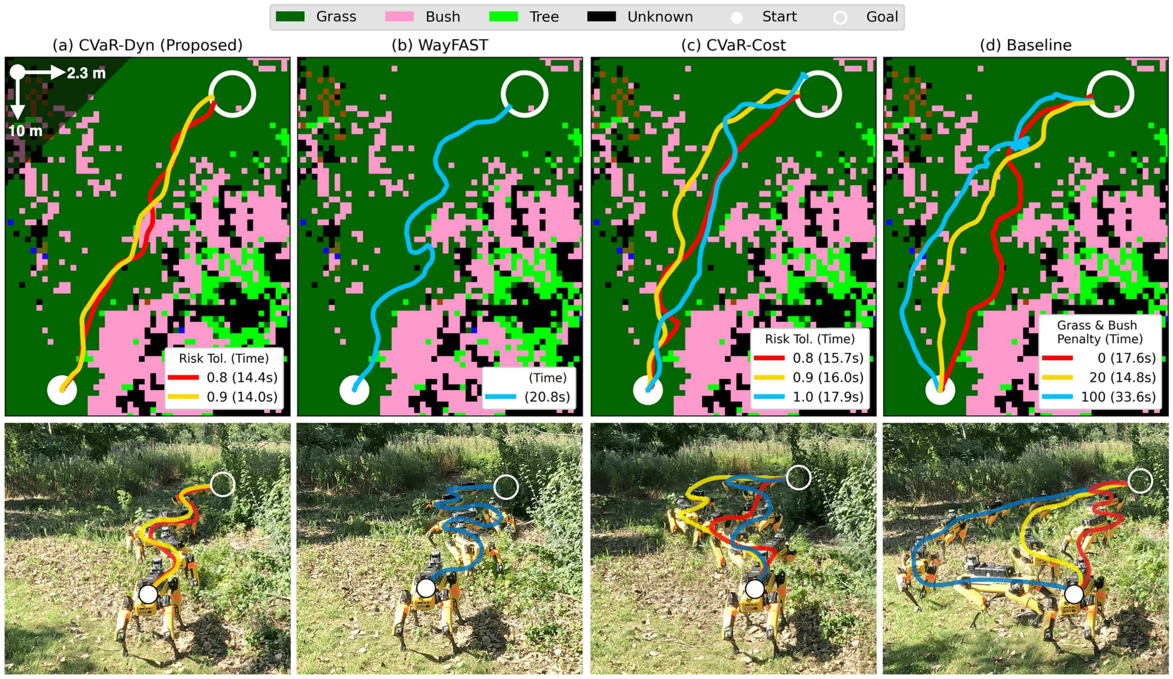


Fig. 21. Representative trials of the outdoor experiments. The top-down semantic maps are shown in the top row and the time-lapse photos are shown in the bottom row. (a) Proposed CVaR-Dyn with $\alpha < 1$ handles the noisy terrain traction well and produces less wavy trajectories compared to other methods. (b) WayFAST (CVaR-Dyn when $\alpha = 1$) relies on the expected traction that provides a poor indication of the actual trajectory outcome, causing the constant correction in headings. (c) CVaR-Cost is more conservative compared to CVaR-Dyn by staying further away from bushes and achieving longer time-to-goal. (d) Baseline assumes nominal traction that leads to understeering. As soft penalties increase, the robot becomes more averse to tall grass and bushes. As most of the test area is filled with grass or bush, the baseline with large soft penalties struggles to find feasible plans to achieve the goal in subsequent trials.

the tall vegetation produce high epistemic uncertainty and the associated terrain is marked as OOD. Without avoiding OOD terrain, the robot is prone to getting stuck in local minima and requires human interventions to drive the robot to areas with feasible trajectories to the goal. In contrast, the planner that avoids OOD terrain achieves better time-to-goal without requiring human interventions.

C. Takeaways From the Hardware Experiments

In summary, the hardware experiments have demonstrated that the proposed CVaR-Dyn is an attractive choice in practice, without incurring the extra computation required by CVaR-Cost that samples additional traction maps or requiring human expertise in designing semantics-based costs for potentially a large

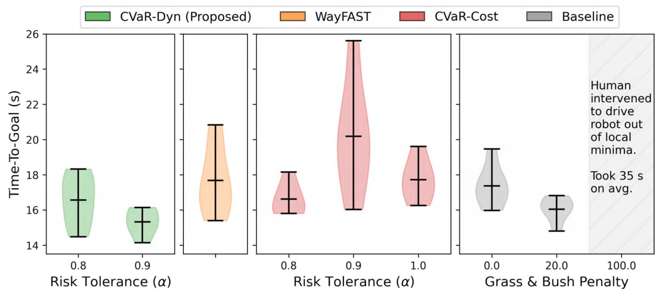


Fig. 22. Distributions of time-to-goal for the local planner benchmark with maximum, average, and minimum values. Each planner completes three round trips or six trials in total. The proposed CVaR-Dyn with $\alpha = 0.9$ outperforms CVaR-Cost that requires more computation, WayFAST (CVaR-Dyn with $\alpha = 1$) that plans with the expected traction, and the baseline that plans with the nominal traction and assigns soft penalties for grass and bushes.

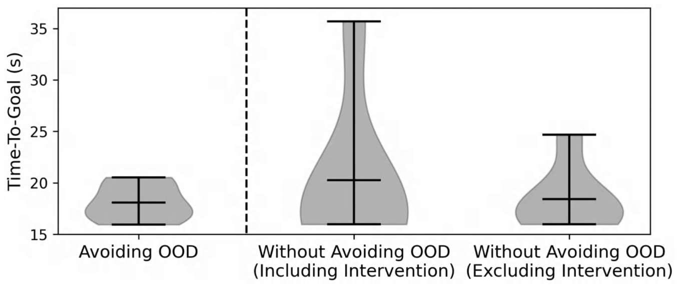


Fig. 24. Distributions of time-to-goal for the OOD avoidance tests over six trials (three round trips), with maximum, average, and minimum values. By avoiding OOD terrain, the planner is less susceptible to local minima and achieves better time-to-goal by avoiding terrain with features unseen during training.

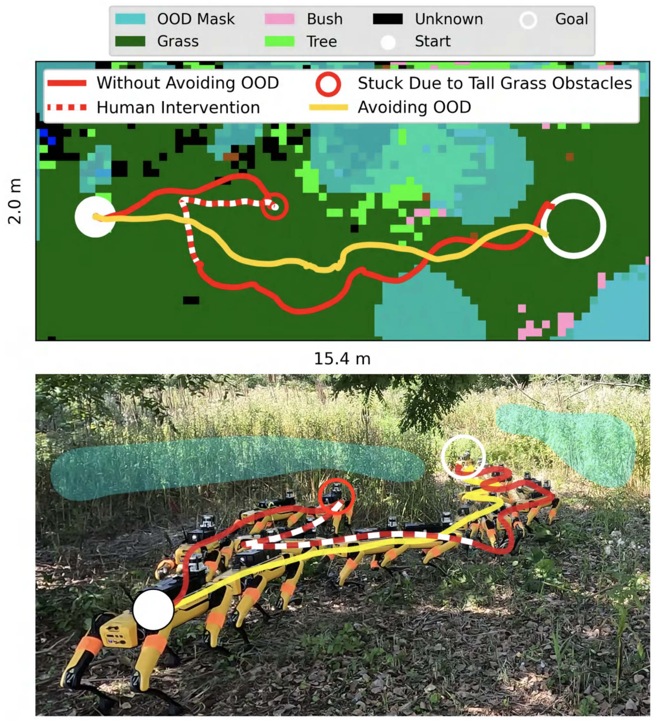


Fig. 23. Representative planner behaviors that show the benefits of avoiding OOD terrain, where the semantic top-down map and the time-lapse photo are shown on the top and bottom. Without OOD avoidance, the robot is susceptible to local minima due to imperfect online map and noisy terrain traction, requiring human interventions to teleoperate the robot to a region with feasible plans to goal. In contrast, assigning auxiliary penalties for OOD terrain makes it easier for the planner to find trajectories to goal.

variety of terrain types. In addition, the ability to estimate epistemic uncertainty allows us to identify and avoid OOD terrain with unreliable traction predictions, thus improving navigation success rate and reducing human interventions.

X. Limitations and Future Work

From the modeling standpoint, this work focuses on 2-D robot models, but models with six degrees of freedom are needed for more challenging terrain [36], [55], [56]. In addition, we use a semantic octomap [50] to model the environment, but computationally cheaper alternatives [10], [57] can be used instead. Moreover, our work relies on the accuracy of the semantic segmentation module, so the proposed pipeline may fail if the test environments look too different from the training environments (e.g., due to lighting and seasonal changes). Therefore, the risk due to the uncertainty in the perception modules needs to be addressed separately [41].

From a data collection standpoint, this work requires empirical traction distributions for training, which may be difficult to attain for high-dimensional features such as RGB images. While the proposed loss can be used to train against instantaneous traction measurements directly, the performance benefits of using EMD^2-based loss need to be reassessed. Moreover, uncertainty-guided data collection methods [37], [58] can be used to collect informative training samples.

From the planning standpoint, this work proposes to simulate state trajectories using the CVaR of traction, but more investigations are needed to generalize the idea to systems with more parameters and different performance metrics. Moreover, our planner avoids OOD terrain in new environments, but online adaptation can be performed [12] if human supervision is available. Lastly, the proposed approach can be paired with a global planner that exploits far-field knowledge [59].

XI. Conclusion

This work presented EVORA, a unified framework for uncertainty-aware traversability learning based on evidential deep learning and risk-aware planning based on CVaR. EVORA modeled uncertain terrain traction via empirical distributions (aleatoric uncertainty) and identified OOD terrain based on the densities of traction predictor's latent features (epistemic uncertainty). By leveraging the proposed uncertainty-aware squared EMD loss, we improved the network's prediction accuracy, OOD detection performance, and the downstream navigation performance. To handle aleatoric uncertainty, the proposed risk-aware planner simulated state trajectories based on the left-tail CVaR of the traction distributions. To handle epistemic uncertainty, we proposed to assign auxiliary costs to terrain whose latent features had low densities, leading to higher navigation success rates. The overall pipeline had been analyzed via extensive simulations and hardware experiments, demonstrating improved navigation performance across different ground robotic platforms.

Appendix A
UCE Loss and Dirichlet Entropy [16]

Given $q = \text{Dir}(\boldsymbol{\beta})$ and the target PMF $\mathbf{y}$

$$L^{\text{UCE}}(q, \mathbf{y}) := \mathbb{E}_{\mathbf{p}\sim q}\left[-\sum_{b=1}^{B} y_b \ \log p_b \right] \tag{34}$$

$$= -\sum_{b=1}^{B} y_b(\Psi(\beta_b) - \Psi(\beta_0)) \tag{35}$$

where Ψ is the digamma function and $\beta_0 := \sum_{b=1}^{B} \beta_b$ is the overall evidence. In addition, the entropy of q is

$$H(q) = \log \mathcal{B}(\boldsymbol{\beta}) + (\beta_0 - B)\Psi(\beta_0) - \sum_{b=1}^{B}(\beta_b - 1)\Psi(\beta_b) \tag{36}$$

where $\mathcal{B}$ denotes the beta function.

Appendix B
Proof of Theorem 1

We proceed directly from the definition of UEMD^2 (17) and simplify the notation by making the expectation over $\mathbf{p} \sim \text{Dir}(\boldsymbol{\beta})$ implicit. Recall that $\mathbf{y}$ is the target PMF, $\text{cs}(\cdot)$ is the cumulative sum operator, and we denote $\text{cs}_b(\cdot)$ as the bth entry of the cumulative sum vector

$$\text{UEMD}^2(\boldsymbol{\beta}, \mathbf{y}) := \mathbb{E}\left[\text{EMD}^2(\mathbf{p}, \mathbf{y})\right] \tag{37}$$

$$= \mathbb{E}\left[\sum_{b=1}^{B} (\text{cs}_b(\mathbf{p}) - \text{cs}_b(\mathbf{y}))^2\right] \tag{38}$$

$$= \sum_{b=1}^{B} \mathbb{E}\left(\sum_{i=1}^{b} p_i - \sum_{i=1}^{b} y_i\right)^2 \tag{39}$$

$$= \sum_{b=1}^{B}\left[\mathbb{E}\left(\sum_{i=1}^{b} p_i\right)^2 - 2\mathbb{E}\sum_{i=1}^{b} p_i \underbrace{\sum\nolimits_{i=1}^{b} y_i}_{\text{cs}_b(\mathbf{y})} + \mathbb{E}\underbrace{\left(\sum\nolimits_{i=1}^{b} y_i\right)^2}_{\text{cs}_b(\mathbf{y})^2}\right]. \tag{40}$$

After separating out the constant additive term $\text{cs}_b(\mathbf{y})^2$, expanding the remaining terms, and moving the expectation inside the summation, we obtain

$$= \sum_{b=1}^{B}\left(\sum_{i=1}^{b} \underbrace{\mathbb{E}[p_i^2]}_{(a)} + 2\sum_{1\le i<j\le b} \underbrace{\mathbb{E}[p_i p_j]}_{(b)} - 2\text{cs}_b(\mathbf{y})\sum_{i=1}^{b} \underbrace{\mathbb{E}[p_i]}_{(c)}\right) + \text{cs}(\mathbf{y})^\top \text{cs}(\mathbf{y}). \tag{41}$$

The terms (a–c) in (41) can be easily derived in closed-form based on the standard properties of a Dirichlet distribution (namely, the variance, covariance, and mean)

$$\text{(a) } \mathbb{E}[p_i^2] = \text{Var}(p_i) + \mathbb{E}[p_i]^2 \tag{42}$$

$$= \frac{\beta_i(\beta_0 - \beta_i)}{\beta_0^2(\beta_0 + 1)} + \frac{\beta_i^2}{\beta_0^2} \tag{43}$$

$$= \frac{\beta_i + \beta_i^2}{\beta_0(\beta_0 + 1)} \tag{44}$$

where $\beta_0 := \sum_{b=1}^{B} \beta_b$. When $i \neq j$

$$\text{(b) } \mathbb{E}[p_i p_j] = \text{Cov}(p_i, p_j) + \mathbb{E}[p_i]\mathbb{E}[p_j] \tag{45}$$

$$= \frac{-\beta_i\beta_j}{\beta_0^2(\beta_0 + 1)} + \frac{\beta_i\beta_j}{\beta_0^2} \tag{46}$$

$$= \frac{\beta_i\beta_j}{\beta_0(\beta_0 + 1)}. \tag{47}$$

Lastly, (c) $\mathbb{E}[p_i] = \frac{\beta_i}{\beta_0}$. Substituting (a–c) in (41), we obtain

$$= \sum_{b=1}^{B}\left(\sum_{i=1}^{b} \frac{\beta_i + \beta_i^2}{\beta_0(\beta_0 + 1)} + 2\sum_{1\le i<j\le b} \frac{\beta_i\beta_j}{\beta_0(\beta_0 + 1)} - 2\text{cs}_b(\mathbf{y})\sum_{i=1}^{b} \frac{\beta_i}{\beta_0}\right) + \text{cs}(\mathbf{y})^\top \text{cs}(\mathbf{y}) \tag{48}$$

$$= \frac{1}{\beta_0(\beta_0 + 1)}\sum_{b=1}^{B}\left(\sum_{i=1}^{b} \beta_i + \sum_{i=1}^{b} \beta_i^2 + 2\sum_{1\le i<j\le b} \beta_i\beta_j\right) - \frac{2}{\beta_0}\sum_{b=1}^{B}\left(\text{cs}_b(\mathbf{y})\sum_{i=1}^{b} \beta_i\right) + \text{cs}(\mathbf{y})^\top \text{cs}(\mathbf{y}) \tag{49}$$

$$= \frac{1}{\beta_0(\beta_0 + 1)}\sum_{b=1}^{B}\left(\underbrace{\sum\nolimits_{i=1}^{b} \beta_i}_{\text{cs}_b(\boldsymbol{\beta})} + \underbrace{\left(\sum\nolimits_{i=1}^{b} \beta_i\right)^2}_{\text{cs}_b(\boldsymbol{\beta})^2}\right) - \frac{2}{\beta_0}\sum_{b=1}^{B}\left(\text{cs}_b(\mathbf{y})\underbrace{\sum\nolimits_{i=1}^{b} \beta_i}_{\text{cs}_b(\boldsymbol{\beta})}\right) + \text{cs}(\mathbf{y})^\top \text{cs}(\mathbf{y}) \tag{50}$$

$$= \frac{\text{cs}(\boldsymbol{\beta})}{\beta_0}^\top \frac{\text{cs}(\boldsymbol{\beta}) + \mathbb{1}_B}{(\beta_0 + 1)} - 2\frac{\text{cs}(\boldsymbol{\beta})}{\beta_0}^\top \text{cs}(\mathbf{y}) + \text{cs}(\mathbf{y})^\top \text{cs}(\mathbf{y}) \tag{51}$$

$$= \text{cs}(\overline{\mathbf{p}})^\top \frac{\text{cs}(\boldsymbol{\beta}) + \mathbb{1}_B}{\beta_0 + 1} + \eta(q, \mathbf{y}) \tag{52}$$

where $\overline{\mathbf{p}} = \boldsymbol{\beta}/\beta_0 = E_{\mathbf{p}\sim\text{Dir}(\boldsymbol{\beta})}[\mathbf{p}]$ and η is defined in (16). ■

Acknowledgment

The authors would like to thank Harrison Busa, Velin Dimitrov, Brennan Vanderlaan, and Leonor Fermoselle for their help on the Spot robots payload, and Brian Okorn and Surya Singh for their feedback on the paper. The authors would also like to thank Andrew Fishberg for his help on the RC car. The experiments using the Spot robot were completed during Xiaoyi Cai's internship at the Boston Dynamics AI Institute.

REFERENCES

[1] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A. Akbar Agha-mohammadi, "STEP: Stochastic traversability evaluation and planning for risk-aware off-road navigation," in *Proc. Robot. Sci. Syst.*, 2021. [Online]. Available: https://www.roboticsproceedings.org/rss17/p021.html

[2] F. Ruetz, N. Lawrancel, E. Hernández, P. Borges, and T. Peynot, "Forest-Trav: Accurate, efficient and deployable forest traversability estimation for autonomous ground vehicles," 2023, *arXiv:2305.12705*.

[3] X. Meng et al., "TerrainNet: Visual modeling of complex terrain for high-speed, off-road navigation," in *Proc. Robot. Sci. Syst.*, 2023. [Online]. Available: https://www.roboticsproceedings.org/rss19/p103.html

[4] A. Shaban, X. Meng, J. Lee, B. Boots, and D. Fox, "Semantic terrain classification for off-road autonomous driving," in *Proc. Conf. Robot Learn.*, 2022, pp. 619–629.

[5] Z. Chen, D. Pushp, and L. Liu, "CALI: Coarse-to-fine alignments based unsupervised domain adaptation of traversability prediction for deployable autonomous navigation," in *Proc. Robot. Sci. Syst.*, 2022. [Online]. Available: https://www.roboticsproceedings.org/rss18/p056.html

[6] T. Guan, D. Kothandaraman, R. Chandra, A. J. Sathyamoorthy, K. Weerakoon, and D. Manocha, "GA-Nav: Efficient terrain segmentation for robot navigation in unstructured outdoor environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8138–8145, Jul. 2022.

[7] G. Kahn, P. Abbeel, and S. Levine, "BADGR: An autonomous self-supervised learning-based navigation system," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1312–1319, Apr. 2021.

[8] X. Yao, J. Zhang, and J. Oh, "RCA: Ride comfort-aware visual navigation via self-supervised learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 7847–7852.

[9] J. Zürn, W. Burgard, and A. Valada, "Self-supervised visual terrain classification from unsupervised acoustic feature learning," *IEEE Trans. Robot.*, vol. 37, no. 2, pp. 466–481, Apr. 2021.

[10] P. Ewen, A. Li, Y. Chen, S. Hong, and R. Vasudevan, "These maps are made for walking: Real-time terrain property estimation for mobile robots," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7083–7090, Jul. 2022.

[11] X. Cai, M. Everett, J. Fink, and J. P. How, "Risk-aware off-road navigation via a learned speed distribution map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 2931–2937.

[12] J. Frey, M. Mattamala, N. Chebrolu, C. Cadena, M. Fallon, and M. Hutter, "Fast traversability estimation for wild visual navigation," in *Proc. Robot. Sci. Syst.*, 2023. [Online]. Available: https://www.roboticsproceedings.org/rss19/p054.html

[13] R. Schmid et al., "Self-supervised traversability prediction by learning to reconstruct safe terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 12419–12425.

[14] H. Lee, J. Kwon, and C. Kwon, "Learning-based uncertainty-aware navigation in 3D off-road terrains," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2023, pp. 10061–10068.

[15] M. Endo, T. Taniai, R. Yonetani, and G. Ishigami, "Risk-aware path planning via probabilistic fusion of traversability prediction for planetary rovers on heterogeneous terrains," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 11852–11858.

[16] B. Charpentier, O. Borchert, D. Zügner, S. Geisler, and S. Günnemann, "Natural posterior network: Deep Bayesian predictive uncertainty for exponential family distributions," in *Proc. Int. Conf. Learn. Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=tV3N0DWMxCg

[17] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 3964–3979, Nov. 2021.

[18] G. Williams et al., "Information theoretic MPC for model-based reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 1714–1721.

[19] L. Hou, C.-P. Yu, and D. Samaras, "Squared earth mover's distance-based loss for training deep neural networks," 2016, *arXiv:1611.05916*.

[20] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.

[21] M. V. Gasparino et al., "WayFAST: Navigation with predictive traversability in the field," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10651–10658, Oct. 2022.

[22] Z. Wang, O. So, K. Lee, and E. A. Theodorou, "Adaptive risk sensitive model predictive control with stochastic search," in *Proc. Learn. Dyn. Control Conf.*, 2021, pp. 510–522.

[23] F. G. Oliveira, A. A. Neto, D. Howard, P. Borges, M. F. Campos, and D. G. Macharet, "Three-dimensional mapping with augmented navigation cost through deep learning," *J. Int. Robot. Sys.*, vol. 101, no. 3, pp. 1–21, 2021.

[24] S. Otte, C. Weiss, T. Scherer, and A. Zell, "Recurrent neural networks for fast and robust vibration-based ground classification on mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 5603–5608.

[25] T. Overbye and S. Saripalli, "G-VOM: A GPU accelerated voxel off-road mapping system," in *Proc. IEEE Intell. Veh. Symp.*, 2022, pp. 1480–1486.

[26] A. J. Sathyamoorthy et al., "VERN: Vegetation-aware robot navigation in dense unstructured outdoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 11233–11240.

[27] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1373–1385, 2013.

[28] A. Li, C. Yang, J. Frey, J. Lee, C. Cadena, and M. Hutter, "Seeing through the grass: Semantic pointcloud filter for support surface learning," *IEEE Robot. Autom. Lett.*, vol. 8, no. 11, pp. 7687–7694, Nov. 2023.

[29] J. Seo, T. Kim, K. Kwak, J. Min, and I. Shim, "ScaTE: A. scalable framework for self-supervised traversability estimation in unstructured environments," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 888–895, 2023.

[30] L. Murphy, S. Martin, and P. Corke, "Creating and using probabilistic costmaps from vehicle experience," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 4689–4694.

[31] J. Gawlikowski et al., "A survey of uncertainty in deep neural networks," *Artif. Intell. Rev.*, vol. 56, no. 1, pp. 1513–1589, 2023.

[32] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, vol. 48, pp. 1050–1059.

[33] I. Osband, C. Blundell, A. Pritzel, and B. Van Roy, "Deep exploration via bootstrapped DQN," in *Proc. Int. Conf. Adv. Neural Inf. Process Syst.*, 2016, vol. 29, pp. 4033–4041.

[34] D. T. Ulmer, C. Hardmeier, and J. Frellsen, "Prior and posterior networks: A survey on evidential deep learning methods for uncertainty estimation," *Trans. Mach. Learn. Res.*, 2023. [Online]. Available: https://openreview.net/forum?id=xqS8k9E75c

[35] S. Triest, M. G. Castro, P. Maheshwari, M. Sivaprakasam, W. Wang, and S. Scherer, "Learning risk-aware costmaps via inverse reinforcement learning for off-road navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 924–930.

[36] H. Lee, T. Kim, J. Mun, and W. Lee, "Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control," *IEEE Robot. Autom. Lett.*, vol. 8, no. 11, pp. 7663–7670, Nov. 2023.

[37] T. Kim, J. Mun, J. Seo, B. Kim, and S. Hong, "Bridging active exploration and uncertainty-aware deployment using probabilistic ensemble neural network dynamics," in *Proc. Robot. Sci. Syst.*, 2023. [Online]. Available: https://www.roboticsproceedings.org/rss19/p086.html

[38] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Int. Conf. Adv. Neural Inf. Process Syst.*, vol. 30, 2017. [Online]. Available: https://papers.nips.cc/paper_files/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html

[39] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," 2021, *arXiv:2110.11334*.

[40] J. Seo, S. Sim, and I. Shim, "Learning off-road terrain traversability with self-supervisions only," *IEEE Robot. Autom. Lett.*, vol. 8, no. 8, pp. 4617–4624, Aug. 2023.

[41] S. Ancha, P. R. Osteen, and N. Roy, "Deep evidential uncertainty estimation for semantic segmentation under out-of-distribution obstacles," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024.

[42] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *Proc. Int. Conf. Adv. Neural Inf. Process Syst.*, 2020, vol. 33, pp. 21464–21475.

[43] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky, "Your classifier is secretly an energy based model and you should treat it like one," in *Proc. Int. Conf. Learn. Representations*, 2020.[Online]. Available: https://openreview.net/forum?id=Hkxzx0NtDB

[44] F. Castañeda, H. Nishimura, R. T. McAllister, K. Sreenath, and A. Gaidon, "In-distribution barrier functions: Self-supervised policy filters that avoid out-of-distribution states," in *Proc. Learn. Dyn. Control Conf.*, 2023, pp. 286–299.

[45] D. D. Fan, A.-A. Agha-mohammadi, and E. A. Theodorou, "Learning risk-aware costmaps for traversability in challenging environments," *IEEE Robot. Autom. Lett.*, vol. 7, no. 1, pp. 279–286, Jan. 2022.

[46] A. Majumdar and M. Pavone, "How should a robot assess risk? Towards an axiomatic theory of risk in robotics," in *Proc. Conf. Robot. Res.*, 2020, pp. 75–84.

[47] J. Gibson et al., "A multi-step dynamics modeling framework for autonomous driving in multiple environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 7959–7965.

[48] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proc. Int. Conf. Adv. Neural Inf. Process Syst.*, 2018, vol. 31, pp. 4759–4770.

[49] X. Cai, M. Everett, L. Sharma, P. R. Osteen, and J. P. How, "Probabilistic traversability model for risk-aware motion planning in off-road environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 11297–11304.

[50] A. Asgharivaskasi and N. Atanasov, "Active Bayesian multi-class mapping from range and semantic segmentation observations," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 1–7.

[51] A. Kirillov, Y. Wu, K. He, and R. Girshick, "PointRend: Image segmentation as rendering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9799–9808.

[52] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A RUGD dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 5000–5007.

[53] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. Mach. Learn.*, 2015, vol. 37, pp. 1530–1538.

[54] T. Kim, G. Park, K. Kwak, J. Bae, and W. Lee, "Smooth model predictive path integral control without smoothing," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10406–10413, Oct. 2022.

[55] A. Datar, C. Pan, and X. Xiao, "Learning to model and plan for wheeled mobility on vertically challenging terrain," 2023, *arXiv:2306.11611*.

[56] L. Sharma, M. Everett, D. Lee, X. Cai, P. Osteen, and J. P. How, "RAMP: A risk-aware mapping and planning pipeline for fast off-road ground robot navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 5730–5736.

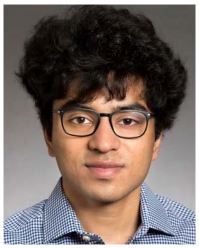
[57] G. Erni, J. Frey, T. Miki, M. Mattamala, and M. Hutter, "MEM: Multi-modal elevation mapping for robotics and learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 11011–11018.

[58] M. Endo and G. Ishigami, "Active traversability learning via risk-aware information gathering for planetary exploration rovers,," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11855–11862, Oct. 2022.

[59] E. Chen, C. Ho, M. Maulimov, C. Wang, and S. Scherer, "Learning-on-the-drive: Self-supervised adaptation of visual off-road traversability models," 2023, *arXiv:2306.15226*.

Xiaoyi Cai received the B.S. and M.S. degrees in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2017 and 2019, respectively. He is currently working toward the Ph.D. degree in Aeronautics and Astronautics from the Massachusetts Institute of Technology, Cambridge, MA, USA.

His research interests include multirobot coordination, risk-aware motion planning, and traversability analysis for off-road navigation.



Siddharth Ancha (Member, IEEE) received the B.Tech. degree in computer science and engineering from the Indian Institute of Technology, Guwahati, India, in 2015, the M.S. degree in computer science from the University of Toronto, Toronto, Canada, in 2017 and the Ph.D. degree in machine learning from Carnegie Mellon University, Pittsburgh, PA, USA, in 2022.

He is a Postdoctoral Researcher with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA, USA. His research interests include enabling robots to robustly and adaptively perceive their surroundings in order to autonomously and reliably operate in complex environments.

Lakshay Sharma (Student Member, IEEE) received the B.S., B.S.E., M.S., and M.S.E. degrees in physics, computer engineering, astrophysics, and robotics in 2021 from the University of Pennsylvania, Philadelphia, PA, USA. He is currently working toward the Ph.D. degree in mechanical engineering with the Massachusetts Institute of Technology, Cambridge, MA, USA.

His research interests include high-speed robot motion planning under uncertainty, social robot navigation, robot planner acceleration, and exoplanet detection.

Philip R. Osteen (Member, IEEE) received the B.S. degree in aerospace engineering and the M.S. degree in mechanical engineering from the University of Florida, Gainesville, FL, USA, in 2007 and 2009, respectively.

He then worked with the Robotics, Vision, and Control Group, University of Seville before joining DEVCOM Army Research Laboratory (ARL), Adelphi, MD, USA, in 2010. He is currently a Roboticist with the ARL. His research interests include label-efficient deployment of machine learning algorithms, multimodal data fusion, and autonomy in unstructured environments.

Bernadette Bucher received the B.S. degree in mathematics and economics in 2014, the M.A. degree in mathematics, the M.A. degree in economics from the University of Alabama, Tuscaloosa, AL, USA, in 2014, and the Ph.D. degree in computer science in 2023 from the GRASP Lab, University of Pennsylvania, Philadelphia, PA, USA, coadvised by Dr. Kostas Daniilidis and Dr. Nikolai Matni.

During her Ph.D., she interned with the Seattle Robotics Lab, NVIDIA Research. Prior to starting her Ph.D., she was a Senior Software Engineer with Lockheed Martin Corporation from 2014 to 2019. She is currently an Assistant Professor with the Robotics Department, University of Michigan, Ann Arbor, MI, USA. Previously, she worked as a Research Scientist with the Boston Dynamics AI Institute.

Stephen Phillips (Member, IEEE) received the B.S. degree in computer science from the University of California, Los Angeles, Los Angeles, CA, USA, in 2014, the M.S. degree in robotics in 2016, and the Ph.D. degree in computer science in 2021 from the University of Pennsylvania, Philadelphia, PA, USA.

He was a Visiting Assistant Professor of engineering with Swarthmore College from 2021 to 2023. He is currently with the Boston Dynamics AI Institute, Cambridge, MA, USA, as an Applied Scientist of Robotics. His research interests include machine learning, computer vision, multiimage matching, SLAM, and sensor fusion.

Jiuguang Wang (Member, IEEE) received the dual B.S. degrees in computer science and electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2008, the triple M.S. degrees in electrical and computer engineering, mechanical engineering, and robotics from Georgia Institute of Technology and Carnegie Mellon University, Pittsburgh, PA, USA, in 2009, 2010, and 2014, respectively, and the Ph.D. degree in robotics from Carnegie Mellon University in 2015.

Dr. Wang is currently the Research Lead for Dexterous Mobile Manipulation with the Boston Dynamics AI Institute, Cambridge, MA, USA, and previously served as the Director of Engineering with Optimus Ride, Inc. His current research interests include contact-rich and whole-body robotic manipulation, semantic navigation, and task and motion planning.

Michael Everett (Member, IEEE) received the S.B., S.M., and Ph.D. degrees in mechanical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2015, 2017, and 2020, respectively.

He was a Postdoctoral Associate and Research Scientist with the Department of Aeronautics and Astronautics, MIT. He was a Visiting Faculty Researcher with Google Research. He joined Northeastern University, Boston, MA, USA, in 2023, where he is currently an Assistant Professor with the Department of Electrical and Computer Engineering and Khoury College of Computer Sciences, Northeastern University. His research interests include the intersection of machine learning, robotics, and control theory, with specific interests in the theory and application of safe and robust neural feedback loops.

Nicholas Roy (Senior Member, IEEE) received the Ph.D. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, USA, in 2003.

He is currently a Professor with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, USA, and a Member of the Computer Science and Artificial Intelligence Laboratory (CSAIL), MIT. His research interests include mobile robotics, decision-making under uncertainty, human–computer interaction, and machine learning.

Jonathan P. How (Fellow, IEEE) received the B.A.Sc. degree from the University of Toronto, Toronto, ON, Canada, in 1987, and the S.M. and Ph.D. degrees in aeronautics and astronautics from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 1990 and 1993, respectively.

He is currently the Richard C. Maclaurin Professor of aeronautics and astronautics with the MIT. Prior to joining MIT in 2000, he was an Assistant Professor with the Department of Aeronautics and Astronautics, Stanford University.

Dr. How is a Fellow of AIAA and was elected to the National Academy of Engineering in 2021.