BenchNav: Simulation Platform for Benchmarking Off-road Navigation Algorithms with Probabilistic Traversability

Masafumi Endo[1], Kohei Honda[2], and Genya Ishigami[1]

***Abstract*— As robotic navigation techniques in perception and planning advance, mobile robots increasingly venture into off-road environments involving complex traversability. However, selecting suitable planning methods remains a challenge due to their algorithmic diversity, as each offers unique benefits. To aid in algorithm design, we introduce BenchNav, an open-source PyTorch-based simulation platform for benchmarking off-road navigation with uncertain traversability. Built upon Gymnasium, BenchNav provides three key features: 1) a data generation pipeline for preparing synthetic natural environments, 2) built-in machine learning models for traversability prediction, and 3) consistent execution of path and motion planning across different algorithms. We show BenchNav's versatility through simulation examples in off-road environments, employing three representative planning algorithms from different domains.**
`https://github.com/masafumiendo/benchnav`

I. Introduction

Mobile robots unchain us from labor-intensive, hazardous activities *in the field*, ranging from environmental monitoring [1], search and rescue [2], to planetary exploration [3]. Autonomous navigation is critical for reliable decision-making, enabling safe and efficient mission operations without consistent human intervention. However, off-road navigation is fraught with risks in unstructured terrains, such as robots getting stuck in deformable terrain, tipping over on steep slopes, or colliding with both positive and negative obstacles. *Traversability* is thus an essential criterion to quantify the level of robot ability to traverse, ranging from safe to hazardous, in unstructured off-road environments.

Off-road navigation has evolved with path and motion planning algorithms that incorporate traversability either through theoretical or machine learning (ML)-based modeling. *Search-based* path planning, valid in discretized graph environments, is notable for its resolution optimality. It can generate global paths for navigation in extreme environments, such as planetary surfaces [3]–[7] and subterranean domains [8], making it suitable for real-world applications, including NASA's AutoNav [4] and Enav [3] systems. Nevertheless, search-based methods often miss robot dynamics, leading to challenges in translating global paths to local motions. *Sampling-based* motion planning has become another common approach, capable of handling high-dimensional spaces through random environmental sampling to construct feasible robot states. Its flexible representation in configuration space allows us to apply complex dynamical models in motion planning, such as closed-loop rapidly-exploring random trees (CL-RRT) [9] and its variants in off-road navigation [10], [11]. Still, they often indirectly include dynamical models and require substantial computational effort to reach asymptotic optimality. *Optimization-based* algorithms, in contrast, enable the direct integration of dynamical models into motion planning, resulting in optimal state and control sequences. Model predictive path integral control (MPPI) [12], a sampling-based optimal control solver, is favored for aggressive off-road navigation [13]–[19], offering rapid trajectory optimization and the ability to handle non-differentiable objectives. These techniques focus on generating sequential, short-horizon solutions while often failing to achieve global optimality.

We here argue that off-road autonomy requires a well-aligned integration of planning algorithms and traversability modeling for robot dynamics. Along with the advances in ML models and growing dataset availability [20]–[23], many studies push forward data-driven traversability prediction and its application to motion planning. However, the lack of standardized benchmarking for planning raises the question: *How can we select the suitable algorithm for designing an off-road navigation system, given their unique differences?* This is often empirically tailored to specific scenarios, though quantitative comparison across different domains is a tedious yet necessary process.

To address the above challenge, we present BenchNav, an open-source, PyTorch-based [24] simulation platform, designed for **Bench**marking off-road **Nav**igation algorithms. Leveraging Gymnasium [25], BenchNav equips three key features to tackle off-road navigation problems as follows:

- Synthetic data generation replicates natural terrain, embedding the challenges of traversability prediction,
- Built-in ML models for probabilistic traversability prediction from given environmental data,
- Path and motion planning algorithm executions unified with robot models factoring in uncertain traversability.

The synthesized process, encompassing data preparation, traversability modeling, and planning, ensures easy and consistent navigation simulations across various algorithms. We perform simulation experiments in deformable off-road environments, focusing on vehicle slip as a typical metric for traversability modeling. The use of different path and motion planning algorithms demonstrates BenchNav's versatility.

*This work was partially supported by JSPS KAKENHI Grant Number JP22J22731.

[1]M. Endo and G. Ishigami are with the Space Robotics Group, Department of Mechanical Engineering, Keio University, Kanagawa 223-8522, Japan `masafumi.endo@keio.jp`, `ishigami@mech.keio.ac.jp`

[2]K. Honda is with the Mobility System Group, Department of Mechanical Systems Engineering, Nagoya University, Aichi 464-8603, Japan `honda.kohei.f4@a.mail.nagoya-u.ac.jp`

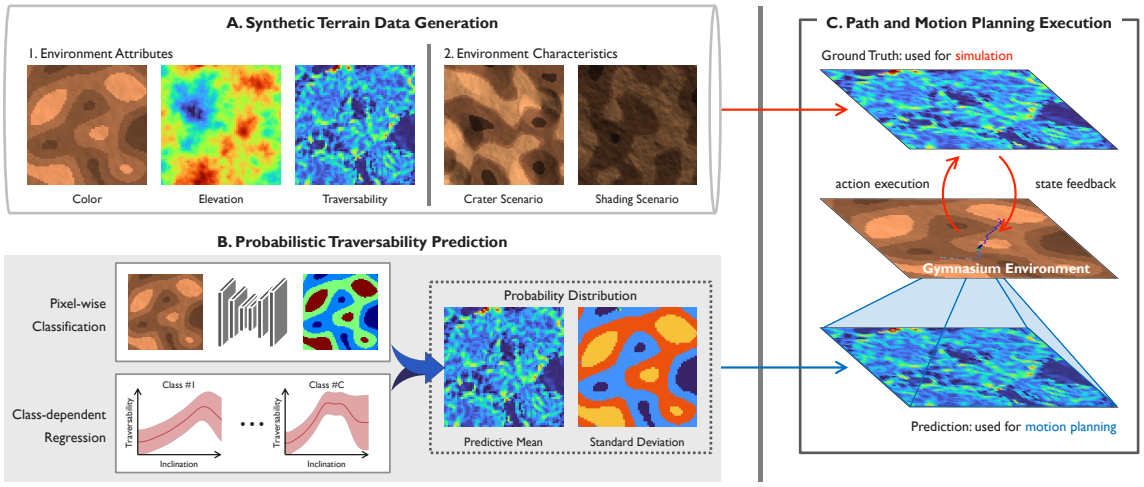


Fig. 1. BenchNav, a platform designed to solve off-road navigation problems, consists of A) synthetic terrain data generation, B) probabilistic traversability prediction based on the classify-then-regress method, and C) path and motion planning execution with given problem formulations and solvers.

II. Problem Statement

We state the off-road navigation problem for mobile robots as one embracing traversability prediction and motion planning. Unstructured environments present complex physical interactions with robots, leading to deviations between commanded and actual velocities. Under significant disturbances, the discrete-time system is given as $\mathbf{s}_{t+1} = F(\mathbf{s}_t, \mathbf{a}_t, \boldsymbol{\lambda})$, where t denotes discrete time steps. Here, $\mathbf{s}_t \in \mathbb{R}^n$ represents the state vector, $\mathbf{a}_t \in \mathbb{R}^m$ the action vector serving as the control input to the robot, and $\boldsymbol{\lambda} \in [0,1]^p$ the traversability coefficient vector, scales $\mathbf{a}_t$. The state transition function F integrates these inputs to update the state vector, determining the system's response accounting for terrain interactions.

The traversability prediction objective is to create a model $\mathcal{M}$, mapping environmental observations $\mathbf{o} \in O$ to traversability coefficients $\boldsymbol{\lambda} \in \Lambda$, where O and Λ are sets of all environmental observations and traversability coefficients, respectively. ML is a powerful tool for discovering latent models by learning general correlations from training data. Recent studies also reveal the value of quantifying uncertainty in traversability learning [26]–[28], arising from inherent observation noise and limited training data. We thus adopt probabilistic ML models as the basis for traversability prediction used across planning algorithms.

Besides, the motion planning objective is to find a sequence of feasible actions $\mathbf{a}_t = \pi(\mathbf{s}_t, \hat{\boldsymbol{\lambda}})$ towards its destination, with a policy π that translates current states and traversability coefficients into the state transition dynamics. The motion planning problem is formulated to optimize user-specified objectives, such as time efficiency, subject to certain constraints. Despite the various solvers available, we emphasize the need to exploit uncertainty in traversability prediction. Following earlier studies [7], [17]–[19], we employ statistical methods for risk inference, such as conditional value at risk (CVaR) [29], to transform probability distributions into deterministic inputs. This approach bridges traversability prediction and motion planning, ensuring consistent execution in solving the off-road navigation problem.

III. BenchNav

This section details BenchNav, a simulation platform tailored for off-road navigation. We developed BenchNav on top of Gymnasium (formerly OpenAI Gym), leveraging its capabilities for sequential decision-making and scalable simulation of diverse environments. BenchNav's implementation in PyTorch further enables GPU acceleration, ensuring efficient computation for ML models and planning algorithms throughout the platform. The overall simulator pipeline, depicted in Fig. 1, unfolds as follows: It first generates controlled datasets containing O and their respective Λ, engineered to catch the difficulties present in traversability prediction. Then, a set of all probability distributions for traversability coefficients, denoted as $\mathbb{P}(\hat{\boldsymbol{\lambda}}|\mathbf{o})$, is formed by initially categorizing symbolic terrain classes from appearance features, followed by predicting class-dependent latent traversability (LT) functions from geometric features. The final stage involves running planning algorithms to find $\mathbf{a}_t = \pi(\mathbf{s}_t, \hat{\boldsymbol{\lambda}})$, with given problem formulations and solvers that incorporate risk inference metrics for uncertain traversability.

A. Synthetic Terrain Data Generation

We synthesize a top-down 2.5D terrain map instance, replicating pixel-wise appearance and geometric features, alongside their latent traversability coefficients. In off-road scenarios, appearance features indicate surface properties, determining distinct traversability trends among terrain classes. Geometric features also dominate the variation in traversability within each class. The data is thereby paired: RGB color data with their corresponding terrain classes $c \in C$, where C is the set of all terrain classes, and elevation data coupled with class-dependent LT functions $f_c(\psi)$, where ψ is the inclination derived by the Horn method [30].

For each map instance, occupancy ratios dictate terrain class distribution; we create such a distribution with Perlin noise to assign color and terrain classes based on the clusters. We then simulate rough terrain elevation with fractal terrain modeling [31] to compute $\lambda = f_c(\psi) + \epsilon_c$, where ϵ represents

additive zero-mean Gaussian noise, and to apply shading to the color data due to elevation changes. We consequently get a map of colors, elevations, and traversability coefficients, depicted in Fig. 1A1. Environmental features are readily adjusted to simulate challenges in traversability prediction, such as ambiguous appearance from intense shading and irregular geometry in crater-like terrains, shown in Fig. 1A2.

B. Probabilistic Traversability Prediction

BenchNav employs two types of pre-trained ML models: 1) a terrain classifier for predicting pixel-wise terrain classes and 2) Gaussian processes (GPs) [32] for estimating class-dependent LT functions with uncertainties. These distinctive ML models are integrated into a single probability distribution via mixtures of GPs [33], representing probabilistic traversability at $\mathbf{s}$ as follows:

$$\mathbb{P}_{\mathbf{s}}(\hat{\lambda}) = \sum_{c \in C} \mathbb{P}_{\mathbf{s}}(c)\, \mathbb{P}_c(\hat{\lambda} \mid \psi_{\mathbf{s}}), \tag{1}$$

where $\mathbb{P}_c(\hat{\lambda} \mid \psi_{\mathbf{s}})$ denotes class-dependent GPs, weighted by a categorical distribution $\mathbb{P}_{\mathbf{s}}(c)$ from the classifier. This probabilistic fusion constructs $\mathbb{P}(\hat{\boldsymbol{\lambda}}|\mathbf{o})$, as shown in Fig. 1B.

While any model capable of predicting pixel-wise categorical distributions is suitable for terrain classification, we opt for the U-Net model [34] with a ResNet-18 backbone [35]. Users also have the option of selecting a single GP that corresponds to the most likely class $c^* = \arg\max_c \mathbb{P}_{\mathbf{s}}(c)$ in eq. (1), rather than summing over multiple GPs.

C. Path and Motion Planning Execution

The last stage simulates off-road robot navigation from a start state $\mathbf{s}_{\text{start}}$ to a goal state $\mathbf{s}_{\text{goal}}$ within the allotted finite positive time budget T. The motion planning problem in off-road environments is given as follows:

$$\text{Find: } \{\mathbf{a}_t\}_{t=0}^{T-1},\ \{\mathbf{s}_t\}_{t=0}^{T} \tag{2a}$$

$$\text{Minimize: } J\left(\{\mathbf{a}_t\}_{t=0}^{T-1},\ \{\mathbf{s}_t\}_{t=0}^{T}\right) \tag{2b}$$

$$\text{subject to: } \mathbf{s}_0 = \mathbf{s}_{\text{start}},\ \mathbf{s}_T = \mathbf{s}_{\text{goal}} \tag{2c}$$

$$\mathbf{s}_t \in S_{\text{free}},\ {}^{\forall}t \in \{0, \dots, T\} \tag{2d}$$

$$\mathbf{s}_{t+1} = F(\mathbf{s}_t, \mathbf{a}_t, \hat{\boldsymbol{\lambda}}),\ {}^{\forall}t \in \{0, \dots, T-1\} \tag{2e}$$

J is the cost function quantifying trajectory optimality to be minimized. S_{free} denotes the set of all free states, indicating navigable space free from hazards. Solving the above problem yields the optimal action vector $\mathbf{a}_t^*$ at each t, controlling the robot to its destination. We design BenchNav to solve the given optimization problem in an iterative fashion, allowing for real-time feedback that compensates for modeling errors in F. Users can define F, J, and S_{free} to formulate the problem according to their respective objectives. BenchNav simulates navigation with deployed planning algorithms as solvers, including both global and local combinations. Here, risk inference metrics convert $\mathbb{P}(\hat{\boldsymbol{\lambda}}|\mathbf{o})$ to $\hat{\boldsymbol{\lambda}}$, incorporating uncertainty in traversability prediction. BenchNav continuously monitors the robot's state, including its experience with traversability, to observe how it safely reaches its destination within T, which is then marked as successful navigation.

IV. Simulation Experiments

A. Implementation Details

We test BenchNav's versatility through simulations aimed at deformable off-road environments. We assume that robot motion is constrained to the terrain surface, thus defining $\mathbf{s}_{\text{start}}$ and $\mathbf{s}_{\text{goal}}$ within a 2.5D spatial space. F is then given as a unicycle model as follows:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \Delta t \cdot \begin{bmatrix} \lambda_{\text{lin}} \cdot v_t \cdot \cos(\theta_t) \\ \lambda_{\text{lin}} \cdot v_t \cdot \sin(\theta_t) \\ \lambda_{\text{ang}} \cdot \omega_t \end{bmatrix}, \tag{3}$$

where $\mathbf{s}_t = [x_t, y_t, \theta_t]^\top$, $\mathbf{a}_t = [v_t, \omega_t]^\top$, and $\boldsymbol{\lambda} = [\lambda_{\text{lin}}, \lambda_{\text{ang}}]^\top$, simplified as $\lambda = \lambda_{\text{lin}} = \lambda_{\text{ang}}$, link linear and angular velocities with changes in position and orientation, capturing the dynamics of robot motion. When formulating the problem, F replaces $\hat{\boldsymbol{\lambda}}$ with $\boldsymbol{\lambda}$ to form eq. (2e). We define J as the function that evaluates states and actions along the trajectory based on their distance to the endpoint for an efficient transition without getting stuck according to the given constraints. We here exploit $\hat{\Lambda}$ to mark S_{free} as states where $\hat{\lambda} > \lambda_{\text{stuck}}$, defining λ_{stuck} as the threshold below which a state becomes stuck due to insufficient traversability. CVaR at level $\alpha = 0.9$ quantifies the distribution to assess uncertainty incorporation in motion planning.

As problem solvers, we deploy three planning methods from distinct domains, each designed for off-road navigation.

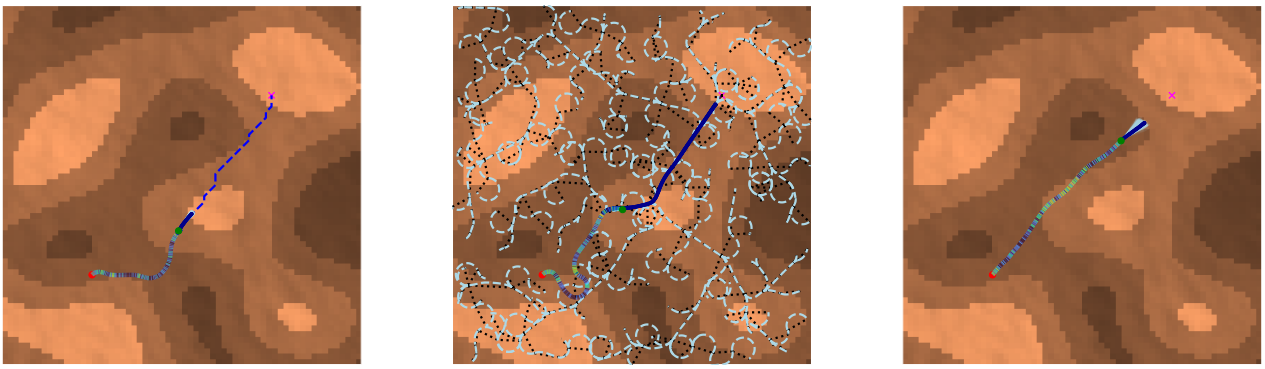
1) Search-based method: Following [8], we implement a hierarchical method divided into global path planning with A* search [36] and local motion planning using the dynamic window approach (DWA) [37]. For each transition, A* finds the shortest path over long horizons in a 2.5D geometric search space, while DWA takes the resulting waypoints as subgoals to plan feasible actions over short horizons.

2) Sampling-based method: We use CL-RRT [9] for global motion planning that incrementally expands a tree via feedback control simulation, allowing the computation of dynamically feasible trajectories. Pure-pursuit and PID controllers manage the steering and velocities of the dynamic model, respectively. Replanning is triggered when substantial deviations occur between the actual and expected states.

3) Optimization-based method: We adopt MPPI [12] for navigation without the use of global path planning. While a reference path remains beneficial for long-horizon decisions, recent studies favor such waypoint-free navigation to take full advantage of planning in control space. Terminal cost is given in J to ensure long-term stability and goal alignment over an infinite horizon.

To quantify the performance of these algorithms on given problem instances, we use three key metrics.

- **Success rate (Succ.) [%]** expresses the ratio of instances for which successful trajectories are found.
- **Total time (T_{total}) [sec]** measures navigation efficiency as the total time taken to traverse a trajectory.
- **Average traversability ($\bar{\lambda}$) [%]** evaluates navigation safety by averaging all the observed traversability coefficients along a trajectory, with higher values indicating a safer traverse.



(a) *Search-based*: A*+DWA (b) *Sampling-based*: CL-RRT (c) *Optimization-based*: MPPI

Fig. 2. Off-road navigation snapshots with different planning algorithms at $t = 25$ [sec]. Red circles, magenta crosses, and green circles mark the start, goal, and current robot positions, respectively. Trajectories are color-coded according to traversability, with cooler colors for safer paths. (a) A* reference paths are blue dashed lines; DWA results are shown with a dark blue solid line and light blue lines. (b) Black dashed lines show edges; light blue dashed lines are for closed-loop simulations, with solid blue lines for planned trajectories. (c) MPPI outcomes are a dark blue solid line with light blue top samples.

B. Experimental Setups

We prepare three environmental scenarios: standard (Std), harsh geometry (HG), and harsh shading (HS). The Std scenario adopts a simple setting mirroring the geometry and appearance trends from ML model training. The HG scenario introduces unique geometric conditions resulting in low traversability, with crater-like terrain in addition to random elevation changes. The HS scenario offers unique appearance conditions complicating traversability prediction due to ambiguous visual cues from variable shading. Each problem instance takes a $32 \times 32\,\mathrm{m}$ map with a $0.5\,\mathrm{m}$ resolution. Off-road navigation starts at $\mathbf{s}_{\text{start}} = (8\,\mathrm{m}, 8\,\mathrm{m})$ and continues until $\mathbf{s}_{\text{goal}} = (24\,\mathrm{m}, 24\,\mathrm{m})$.

C. Results

We highlight representative simulation examples in the Std scenario by displaying snapshots of ongoing navigation at $t = 25$ [sec] using distinct planning methods, as illustrated in Fig. 2. The nature of these planning algorithms is evident in their trajectories: Fig. 2a shows a safe and fairly efficient trajectory under A* guidance; Fig. 2b a suboptimal trajectory due to its sampling nature; and Fig. 2c an efficient but slightly hazardous trajectory, a consequence of its short-horizon solutions. We emphasize BenchNav's ability to apply various planning algorithms, focusing on synthetic data generation and probabilistic traversability prediction with built-in ML models, as fundamental steps prior to motion planning. This feature allows users to easily simulate off-road navigation, facilitating problem formulation and implementation of planning algorithms while accounting for uncertain traversability.

We also present a quantitative summary of MPPI planning to see how different environmental scenarios impact off-road navigation performance. Table I summarizes the results from simulation experiments across 20 problem instances for each of the Std, HG, and HS scenarios, with $T = 100$ [sec]. MPPI performs best in the Std scenario, where ML models provide accurate traversability predictions due to training on in-domain data. The HG and HS scenarios introduce challenges resulting in lower success rates and longer traverse times as ML models struggle with out-of-domain features such as steep inclines and shaded visuals. In off-road environments, robots often face novel situations that lead to *epistemic uncertainty*, stemming from insufficient training data. Thus, incorporating uncertainty is critical in assessing the risk inherent in erroneous traversability prediction. BenchNav allows us to implement various planning algorithms and explore risk inference metrics for resilient off-road navigation. It equips well-controlled environmental feature generation to replicate challenging scenarios, providing insight into the factors contributing to uncertain traversability predictions.

TABLE I

MPPI PLANNING RESULTS ON DIFFERENT SCENARIOS

Scenario	Succ.	T_{total}	$\bar{\lambda}$
Std	90	52.7 ± 22.5	76.3 ± 8.8
HG	80	60.2 ± 26.1	76.5 ± 4.5
HS	85	58.6 ± 25.7	74.2 ± 11.4

D. Limitations and Possible Extensions

We clarify BenchNav's limitations as 1) the lack of local observability using onboard sensors and 2) insufficient 3D representations in both perception for handling overhanging objects and motion for representing dynamic robot behaviors. We will implement these capabilities to simulate navigation algorithms designed for more dynamic, aggressive maneuvers in unknown environments [16], [38].

V. Conclusion

We have developed BenchNav, a simulator that tests planning algorithms alongside ML-based traversability modeling for off-road navigation. Simulation experiments verified that BenchNav consistently executes different off-road navigation algorithms and provides well-controlled datasets replicating difficulties present in traversability prediction. Future work will address the limitations described in IV-D to make the simulator more realistic. We also aim to extend navigation components by adopting end-to-end learning [17]–[19], [39] and reinforcement learning techniques [28], [40], [41].

REFERENCES

[1] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robot. Autom. Mag.*, vol. 19, no. 1, pp. 24–39, 2012.

[2] K. Nagatani *et al.*, "Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots," *J. Field Robot.*, vol. 30, no. 1, pp. 44–63, 2013.

[3] V. Verma *et al.*, "Autonomous robotics is driving perseverance rover's progress on mars," *Sci. Robot.*, vol. 8, no. 80, p. eadi3099, 2023.

[4] S. Goldberg, M. Maimone, and L. Matthies, "Stereo vision and rover navigation software for planetary exploration," in *Proc. IEEE Aerosp. Conf.*, vol. 5, 2002, pp. 5–5.

[5] G. Ishigami, K. Nagatani, and K. Yoshida, "Path planning for planetary exploration rovers and its evaluation based on wheel slip dynamics," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 2361–2366.

[6] S. Daftry *et al.*, "MLNav: Learning to safely navigate on martian terrains," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5461–5468, 2022.

[7] M. Endo *et al.*, "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. 11 852–11 858.

[8] D. D. Fan *et al.*, "STEP: Stochastic traversability evaluation and planning for safe off-road navigation," in *Proc. Robot.: Sci. Syst.*, 2021.

[9] Y. Kuwata *et al.*, "Real-time motion planning with applications to autonomous urban driving," *IEEE Trans. Control Syst. Technol.*, vol. 17, no. 5, pp. 1105–1118, 2009.

[10] R. Takemura and G. Ishigami, "Traversability-based trajectory planning with quasi-dynamic vehicle model in loose soil," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 8411–8417.

[11] ——, "Uncertainty-aware trajectory planning: Using uncertainty quantification and propagation in traversability prediction of planetary rovers," *IEEE Robot. Autom. Mag.*, pp. 2–12, 2023.

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

[13] M. V. Gasparino *et al.*, "WayFAST: Navigation with predictive traversability in the field," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10 651–10 658, 2022.

[14] M. V. Gasparino, A. N. Sivakumar, and G. Chowdhary, "WayFASTER: a self-supervised traversability prediction for increased navigation awareness," *arXiv:2402.00683*, 2024.

[15] 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.

[16] T. Han *et al.*, "Model predictive control for aggressive driving over uneven terrain," *arXiv:2311.12284*, 2023.

[17] X. Cai *et al.*, "Risk-aware off-road navigation via a learned speed distribution map," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 2931–2937.

[18] ——, "Probabilistic traversability model for risk-aware motion planning in off-road environments," *arXiv:2210.00153*, 2022.

[19] ——, "Evora: Deep evidential traversability learning for risk-aware off-road autonomy," *arXiv:2311.06234*, 2023.

[20] S. Triest *et al.*, "TartanDrive: A large-scale dataset for learning off-road dynamics models," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2022, pp. 2546–2552.

[21] S. Sharma *et al.*, "CaT: Cavs traversability dataset for off-road autonomous driving," *IEEE Access*, vol. 10, pp. 24 759–24 768, 2022.

[22] M. Sivaprakasam *et al.*, "TartanDrive 1.5: Improving large multimodal robotics dataset collection and distribution," in *Int. Conf. Robot. Autom. Workshop Pretrain. Robot.*, 2023.

[23] ——, "TartanDrive 2.0: More modalities and better infrastructure to further self-supervised learning research in off-road driving tasks," *arXiv:2402.01913*, 2024.

[24] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[25] G. Brockman *et al.*, "Openai gym," *arXiv:1606.01540*, 2016.

[26] 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. 11 855–11 862, 2022.

[27] 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, 2022.

[28] S. Triest *et al.*, "Learning risk-aware costmaps via inverse reinforcement learning for off-road navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 924–930.

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

[30] M. Ono *et al.*, "Mars 2020 site-specific mission performance analysis: Part 2. surface traversability," in *Proc. AIAA SPACE*, 2018.

[31] Y. Yokokohji, S. Chaen, and T. Yoshikawa, "Evaluation of traversability of wheeled mobile robots on uneven terrains by fractal terrain model," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 3, 2004, pp. 2183–2188 Vol.3.

[32] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. Cambridge, MA, USA: MIT press, 2006.

[33] V. Tresp, "Mixtures of gaussian processes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 13, 2000.

[34] O. Ronneberger *et al.*, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Med. Image Comput. Comput.-Assist. Interv.*, 2015, pp. 234–241.

[35] K. He *et al.*, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[36] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968.

[37] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robot. Autom. Mag.*, vol. 4, no. 1, pp. 23–33, 1997.

[38] X. Meng *et al.*, "Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation," in *Proc. Robot.: Sci. Syst.*, 2023.

[39] M. G. Castro *et al.*, "How does it feel? self-supervised costmap learning for off-road vehicle traversability," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2023, pp. 931–938.

[40] L. Gan *et al.*, "Energy-based legged robots terrain traversability modeling via deep inverse reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 8807–8814, 2022.

[41] G. B. Margolis *et al.*, "Learning to see physical properties with active sensing motor policies," *arXiv:2311.01405*, 2023.