CRESTE: Scalable Mapless Navigation with Internet Scale Priors and Counterfactual Guidance

Arthur Zhang, Harshit Sikchi, Amy Zhang, Joydeep Biswas
The University of Texas at Austin

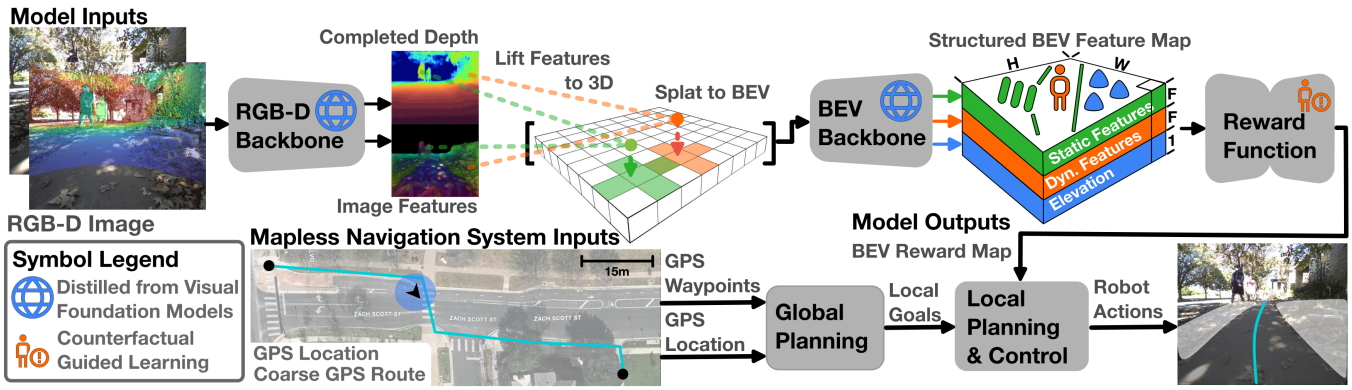

Fig. 1: CRESTE learns a perceptual encoder[1,2] and reward function[3] to predict structured bird's eye view (BEV) feature and reward maps for navigation. Our model inherits generalization and robustness from visual foundation models and learns expert-aligned rewards using our counterfactual-guided learning framework. We integrate CRESTE in a modular navigation system that uses coarse GPS guidance and rewards to reach navigation goals safely.

***Abstract*— We introduce CRESTE, a scalable learning-based mapless navigation framework to address the open-world generalization and robustness challenges of outdoor urban navigation. Key to achieving this is learning perceptual representations that generalize to open-set factors (e.g. novel semantic classes, terrains, dynamic entities) and inferring expert-aligned navigation costs from limited demonstrations. CRESTE addresses both these issues, introducing 1) a visual foundation model (VFM) distillation objective for learning open-set structured bird's-eye-view perceptual representations, and 2) counterfactual inverse reinforcement learning (IRL), a novel active learning formulation that uses counterfactual trajectory demonstrations to reason about the most important cues when inferring navigation costs. We evaluate CRESTE on the task of kilometer-scale mapless navigation in a variety of city, offroad, and residential environments and find that it outperforms all state-of-the-art approaches with 70% fewer human interventions, including a 2-kilometer mission in an unseen environment with just 1 intervention; showcasing its robustness and effectiveness for long-horizon mapless navigation. Videos and additional materials can be found on the project page: https://amrl.cs.utexas.edu/creste.**

I. INTRODUCTION

Mapless navigation is the task of reaching user-specified goals without high-definition (HD) maps and precise navigation waypoints. Mapless approaches plan routes using egocentric sensor observations (e.g. RGB images, point clouds, GPS), coarse waypoints from public routing services, and satellite imagery. These solutions demonstrate promise as a scalable alternative to conventional map-centric approaches: enabling generalization to unforeseen factors (e.g. curb ramps and foliage) and dynamic entities (e.g. pedestrians and strollers) while reducing map maintenance overhead and reliance on pre-defined routes.

Traditional geometric-only mapless solutions [1, 2] exhibit robust generalization when it is only necessary to consider costs for geometric factors like static obstacles. However, open-world navigation demands perception systems that perceive an open set of factors unknown apriori, ranging from terrain preferences (grass vs. concrete) to semantic cues (crosswalks and crossing signs). Furthermore, it requires the ability to identify the most salient features in the scene, and how they influence the choice of paths.

Learning-based approaches are a scalable alternative that considers factors beyond geometry, but must overcome data scarcity, robustness, and generalization challenges to achieve similar reliability. These approaches broadly consist of single-factor perception, hand-curated multi-factor perception, end-to-end learning, and zero-shot pre-trained large language model (LLM)/visual language model (VLM) transfer. While single-factor [3, 4, 2] and multi-factor [5, 6] methods learn representations that consider relevant navigation factors (e.g. geometry, terrain, semantics), they rely on a hand-curated list of semantic classes and terrains that limit generalization to unseen classes. End-to-end methods [7, 8, 9] alleviate this by jointly learning the representation and policy from expert demonstrations, but are prone to overfitting without large-scale robot datasets. While recent works [10, 11] demonstrate that pre-trained LLMs and VLMs can reason about expert-aligned behavior without large-scale robot datasets, we empirically demonstrate that they are poorly attuned to urban navigation, leading to brittle zero-shot transfer in complex scenes.

We address the aforementioned limitations with CRESTE, **Counterfactuals for Reward Enhancement with Struc-**

tured Embeddings, a novel approach that learns open-set representations and navigation costs/rewards for mapless urban navigation. To learn open-set representations, CRESTE combines prior work on bird's eye view (BEV) representation learning [5] and visual foundation models (VFMs) [12, 13] to learn BEV map representations with inherited real-world robustness and open-set semantic knowledge. We unify priors from multiple foundation models by distilling image features from Dinov2 [12] and refining these features in BEV using SAM2 [13] instance labels. CRESTE infers expert-aligned navigation costs without large-scale demonstrations by leveraging counterfactuals to guide learning, where counterfactuals hold all other variables constant except for the path taken from the start to the end goal. Unlike conventional preference learning, our proposed counterfactual inverse reinforcement learning (IRL) objective explicitly minimizes rewards along paths that exhibit undesirable behavior (e.g. veering off crosswalks, driving off curb ramps, etc.), enabling operators to correct robot behavior by providing offline counterfactual feedback. We summarize the main contributions of our work as follows:

- **Representation Learning Through Model Distillation.** A new model architecture and distillation objective for distilling navigation priors from visual foundation models to a lightweight image to BEV map backbone.
- **Counterfactually Aligned Rewards.** An active learning framework and counterfactual IRL formulation for reward alignment using counterfactual and expert demonstrations.

We demonstrate our approach's effectiveness through real-world kilometer-scale navigation experiments in urban environments with off-road terrain, elevated walkways, sidewalks, intersections, and other challenging conditions. In total, we evaluate in **6** distinct seen and unseen geographic areas and significantly outperform existing state-of-the-art imitation learning, inverse reinforcement learning, and heuristic-based methods on the task of mapless navigation.

II. APPROACH

CRESTE is a modular approach with two key components that can be trained end-to-end: 1) A perceptual encoder $\Theta(o_{\text{rgb, t}}, o_{\text{depth, t}})$ that takes the robot's current RGB and sparse depth observation and predicts a completed depth image $y_{\text{depth, t}}$ and structured BEV feature map $y_{\text{bev, t}}$; 2) A reward function $r_\phi(y_{\text{bev, t}})$ that takes $y_{\text{bev, t}}$ and outputs a BEV scalar reward map $y_{\text{reward, t}}$. In the remainder of this section, we describe the following: 1) Sec. II-A - The CRESTE model architecture, 2) Sec. II-B - The CRESTE training procedure, where Sec. II-B.1 presents our VFM distillation objective for Θ and Sec. II-B.2 presents our active reward learning framework for learning r_ϕ from expert and counterfactual demonstrations.

A. CRESTE *Model Architecture*

Perception Encoder Θ. Our 25.5M parameter perceptual encoder Θ draws inspiration from the TerrainNet [5] backbone, which trains a RGB-D encoder f_{rgbd} to predict a latent feature map z_{rgbd} using an EfficientNet-B0 [15] encoder and a completed depth map y_{depth} using a depth completion head $f_{\text{depth}}(z_{\text{rgbd}})$. Like TerrainNet, we train a lift-splat module $f_{\text{splat}}(z_{\text{rgbd}}, y_{\text{depth}})$ to lift latent features to 3D and "splat" them to an unstructured BEV feature map $z_{\text{bev, splat}}$. Finally, we pass $z_{\text{bev, splat}}$ to a BEV inpainting backbone f_{bev} that uses a shared U-Net [16] encoder and separate decoders to predict a structured feature map consisting of separate semantic and elevation layers.

Building on TerrainNet, we make two key architectural modifications. Our first modification, semantic decoder f_{semantic}, promotes learning semantic and geometric-aware features by regressing image features from Dinov2 [12] using latent image features z_{rgbd}. This design is analogous to model distillation [17] and allows our RGB-D encoder f_{rgbd} to inherit properties from VFMs like robustness to perceptual aliasing and open-set semantic understanding.

Our second modification stems from the observation that Dinov2 features alone lack the entity understanding needed to backproject and inpaint features in heavily occluded urban scenes with noisy depth predictions. Thus, we supplement f_{bev} with two panoptic map decoders $f_{\text{bev,static}}$ and $f_{\text{bev,dynamic}}$ to ensure that predicted BEV feature maps y_{bev} are consistent with BEV instance maps from SegmentAnythingv2 (SAM2) [13]. We use Supervised Contrastive Loss [18] to optimize y_{bev} such that features belonging to the same instance are closer in embedding space than features belonging to different instances. Combined with f_{semantic}, our modifications synergistically unify the strengths of Dinov2 and SAM2 to learn open-set semantic, geometric, and instance-aware representations grounded in the local planning horizon. Altogether, our structured BEV feature map y_{bev} consists of three layers stacked along the channel dimension: 1) Static panoptic feature map $y_{\text{bev, static}}$, 2) Dynamic panoptic feature map $y_{\text{bev, dynamic}}$, and 3) Elevation map $y_{\text{bev, elev}}$.

Reward Function r_ϕ. To ensure our reward function enforces spatial invariance and considers multi-scale features, we implement r_ϕ using a 0.5M parameter Multi-Scale Fully Convolutional Network (MS FCN) first used by Wulfmeier et al. [19]. We supervise r_ϕ using our counterfactual IRL loss, which we describe along with our training procedure for Θ in the next section.

B. CRESTE *Training Procedure*

To train CRESTE, we first optimize Θ and freeze the parameters before training r_ϕ using our active reward learning framework with counterfactuals. Altogether, our full learning objective is:

$$\mathcal{L}_{\text{CRESTE}} = \mathcal{L}_{\text{rgbd}} + \mathcal{L}_{\text{bev}} + \mathcal{L}_{\text{IRL}} \tag{1}$$

1) Training the Perceptual Encoder Θ: We propose a distillation label generation module that leverages VFMs to generate five training labels for supervising Θ. Using sequential SE(3) robot poses and synchronized RGB–point cloud pairs $(o_{\text{rgb, 1:t}}, o_{\text{cloud, 1:t}})$, we generate semantic feature maps $\hat{y}_{\text{semantic}}$, completed depth labels $\hat{y}_{\text{depth}}$, BEV map labels

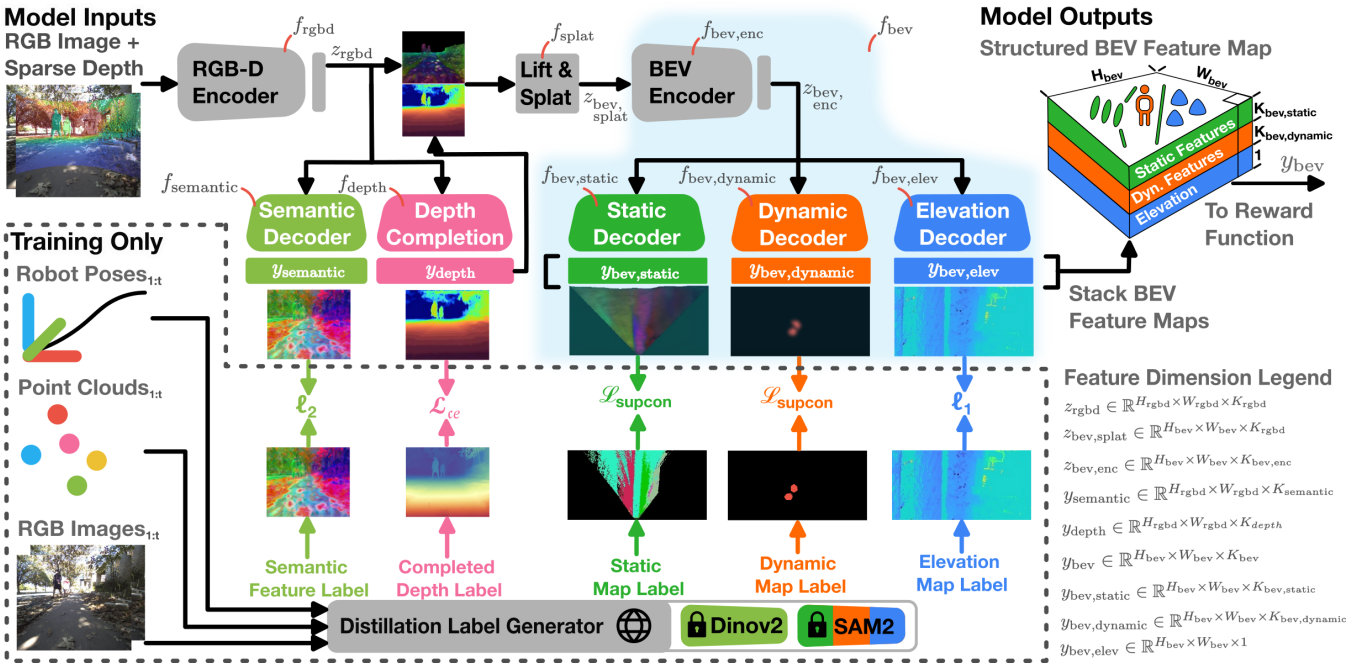

Fig. 2: Model architecture and training procedure for the CRESTE perceptual encoder Θ. Using a RGB and sparse depth image, f_{rgbd} extracts image features z_{rgbd} and performs depth completion. Next, we lift and splat z_{rgbd} to an unstructured BEV feature map before predicting continuous static panoptic, dynamic panoptic, and elevation features. Finally, we stack the predicted features to construct a structured BEV feature map for our learned reward function. We supervise Θ using semantic feature maps, completed depth, and BEV map labels generated by our SegmentAnythingv2 [14] and Dinov2 [12]-powered distillation label generator. We define the dimensions for important feature maps in the Feature Dimension Legend on the bottom right.

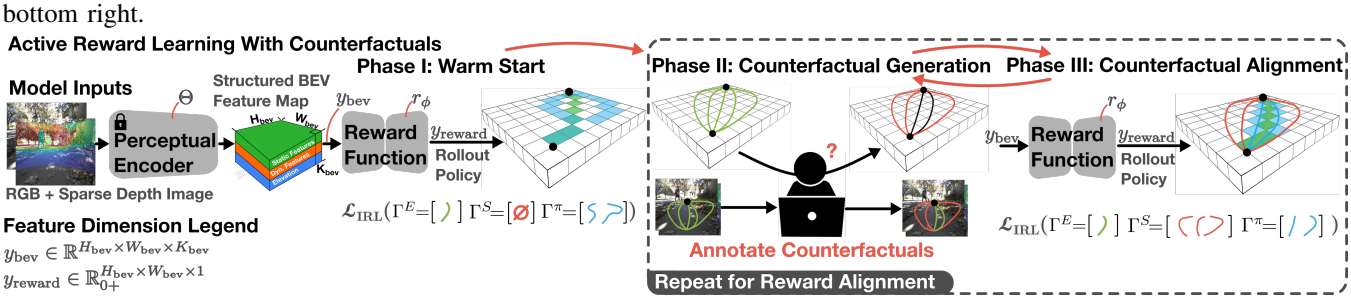

Fig. 3: Framework for Active Reward Learning with Counterfactuals. Before reward learning, we train and freeze the perceptual encoder Θ and use the output BEV feature map y_{bev} with r_ϕ to predict BEV reward maps y_{reward}. In phase I, we train r_ϕ using our counterfactual IRL objective $\mathcal{L}_{\text{IRL}}$ using only expert demonstrations Γ^E. In phase II, we plan goal-reaching paths using y_{reward} and identify samples that align poorly with human preferences. We generate alternate trajectories for these samples and query the operator to select counterfactual demonstrations Γ^S using o_{rgb} for context. In phase III, we retrain r_ϕ using $\mathcal{L}_{\text{IRL}}$, this time with Γ^E and Γ^S. We repeat phases II and III to iteratively improve r_ϕ until it is aligned with human preferences.

for static/dynamic entities ($\hat{y}_{\text{bev, static}}$, $\hat{y}_{\text{bev, dynamic}}$), and BEV elevation map labels $\hat{y}_{\text{bev, elev}}$. For more information on label generation, please see Appendix Sec. VI-A.

RGB-D Encoder f_{rgbd}. We first train f_{rgbd} until convergence via backpropagation from f_{semantic} and f_{depth}. Specifically, we supervise f_{semantic} via an L2/MSE loss and f_{depth} using a cross-entropy classification loss $\mathcal{L}_{\mathcal{CE}}$, where $\hat{y}_{\text{depth}}$ is uniformly discretized into bins. Altogether the training objective for the RGB-D backbone is:

$$\mathcal{L}_{\text{rgbd}} = \alpha_1 l_2(y_{\text{semantic}}, \hat{y}_{\text{semantic}}) + \alpha_2 \mathcal{L}_{\mathcal{CE}}(y_{\text{depth}}, \hat{y}_{\text{depth}}) \tag{2}$$

where α_1 and α_2 are tunable hyperparameters.

BEV Inpainting Backbone f_{bev}. We warm-start f_{bev} by freezing f_{rgbd} and jointly training for a few epochs before unfreezing f_{rgbd} and training end-to-end. We train $f_{\text{bev, static}}$ and $f_{\text{bev, dynamic}}$ with Supervised Contrastive Loss [18] ($\mathcal{L}_{\text{contrastive}}$), optimizing the embedding space such that features of the same entity remain close while repelling those from different entities. This is key to learning continuous feature maps from BEV instance map labels $\hat{y}_{\text{bev, static}}$ and $\hat{y}_{\text{bev, dynamic}}$. We train $f_{\text{bev, elev}}$ to predict the elevation $\hat{y}_{\text{bev, elev}}$ using l_1 regression loss. Our full training objective for f_{bev}

Fig. 4: Satellite image of testing locations for short horizon mapless navigation experiments. We evaluate baselines across 2 seen (green) and 4 unseen (red) urban locations. Our testing locations consist of residential neighborhoods, urban shopping centers, urban parks, and offroad trails. In each location, each baseline must start from an endpoint on the annotated blue trajectory and navigate to the opposite end of the trajectory. We denote each location's ID with a numerical superscript.

is:

$$\mathcal{L}_{\text{bev}} = \beta_1 \mathcal{L}_{\text{contrastive}}(y_{\text{static}}, \hat{y}_{\text{static}}) + \beta_2 \mathcal{L}_{\text{contrastive}}(y_{\text{dynamic}}, \hat{y}_{\text{dynamic}}) + \beta_3 l_1(y_{\text{elev}}, \hat{y}_{\text{elev}}) \quad (3)$$

where β_1, β_2, and β_3 are tunable hyperparameters. For specific hyperparameter settings, please refer to the Appendix.

2) Training the Reward Function r_ϕ: We train r_ϕ using our three-phase active reward learning framework and counterfactual IRL objective. We summarize the three phases next and describe the counterfactual annotation procedure and counterfactual IRL in Appendix Sec. II-B.3.

In *Phase I (Warmstart)*, we train r_ϕ using only expert demonstrations to obtain a base policy. In *Phase II (Counterfactual Generation)*, we use Hybrid A* [20] to plan a path from the same start/goal as the expert using the learned rewards. We compute the Hausdorff distance between the planned path and expert, selecting samples that exceed a distance threshold as candidates for counterfactual labeling. For candidate samples, we replay and accumulate the recent history of expert observations, sample alternate trajectories from the start/end goal of the expert, and ask a human to select alternate trajectories that exhibit undesirable behavior (e.g. veering off crosswalks, driving into curbs) or violate preferences (e.g. driving on sidewalk versus grass). In *Phase III (Reward Alignment)*, we retrain r_ϕ using $\mathcal{L}_{\text{IRL}}$ with expert and counterfactual demonstrations. We repeat phases II and III using the r_ϕ to achieve expert alignment.

3) Counterfactual IRL Objective: We derive our counterfactual IRL objective in Eq. 4 from the Bradley-Terry model of preferences, where $\rho^E(r_\phi)$, $\rho^S(r_\phi)$, and $\rho^\pi(r_\phi)$ are the visitation distributions induced by the expert, counterfactual, and learned policies and r_ϕ is the predicted reward.

$$\mathcal{L}_{\text{IRL}} = \rho^E(s,a) - (\alpha\rho^S(s,a) + (1-\alpha)\mathbb{E}_\pi[\rho^\pi(s,a)])r_\phi \quad (4)$$

By enforcing non-trivial rewards with gradient regularization [21], the above objective learns a reward function such that the difference between the expert's return and agent policy's return is minimized while ensuring suboptimal counterfactuals have a low return. This can be seen as a special case of preference learning [22, 23], as counterfactuals are a more specific type of feedback than general preferences, which can be between any two trajectories. We provide the full derivation and analysis in the Appendix Sec. VI-B.

III. EXPERIMENTS

We investigate the importance of our contributions and overall performance on the task of mapless urban navigation by answering the following questions.

- (Q_1) How well does CRESTE generalize to unseen urban environments for mapless urban navigation?
- (Q_2) How important are structured BEV perceptual representations for downstream policy learning?
- (Q_3) How much do counterfactual demonstrations improve learned policies in challenging urban scenes?
- (Q_4) How well does CRESTE perform long horizon mapless urban navigation compared to other SOTA approaches?

We conduct all experiments using a Clearpath Jackal mobile robot shown in Fig. 7. We outfit the robot with a synchronized RGB camera and LiDAR, and obtain coarse GPS localization and heading from a cellular smartphone. We conduct our experiments in six unique areas shown in Fig. 4 on the task of mapless navigation. We evaluate each approach using the average subgoal completion time (AST), percentage of subgoals reached (%S), and number of interventions per 100m traversed (NIR).

Baselines. We compare CRESTE against 4 baselines, which we describe in detail in Appendix Sec. VI-C and summarize next: 1) Geometric Only, 2) (PACER+G [3]) - a multi-factor terrain and geometric aware model, 3) ViNT [8] - a robot navigation foundation model, and 4) PIVOT [11] - a pre-trained VLM-based navigation model. We train/finetune methods 2 and 3 on the same dataset as CRESTE, which consists of 3 hours of expert demonstrations. Additionally, we conduct ablations to understand the importance of each contribution: 1) CRESTE-cfs-st, our model trained without counterfactuals and structured representations, 2) CRESTE-cfs, our model trained without counterfactuals, and 3) CRESTE-st, our model trained without structured representations. We describe additional model implementation and training details in Appendix Sec. VI-C.

Analysis. Across all experiments, CRESTE outperforms existing methods, achieving significantly fewer interventions in seen and unseen environments, including 1.9km mission traversal with just a single intervention. We provide qualitative analysis of each method's failures in Appendix Sec. VI-D. Quantitatively, PACER+G, the next best approach, requires two times more interventions than CRESTE in seen

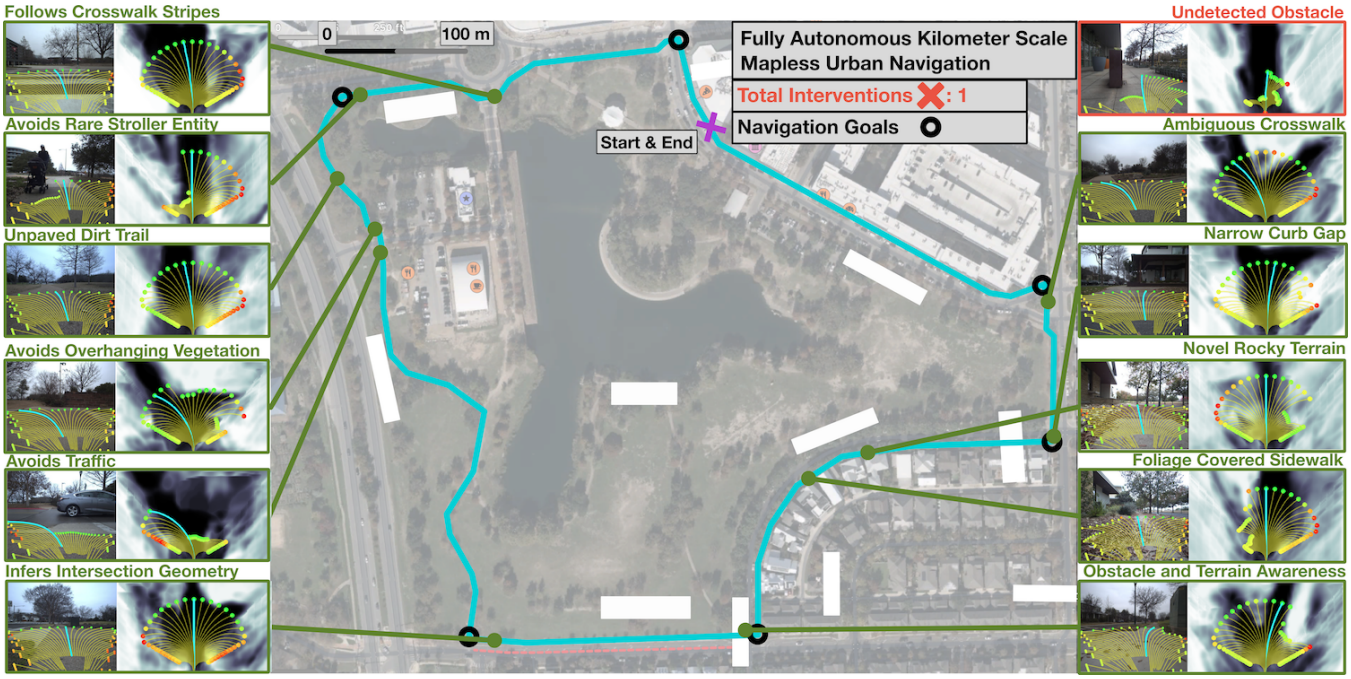

Fig. 5: Satellite image of our 2 kilometer long-horizon testing area, with examples with front view RGB image observations and CRESTE's predicted BEV costmap (converted from the predicted BEV reward map). We annotate successful examples in green with a brief description of the situation. We annotate unsuccessful examples in red, present the observation right before the intervention, and provide a brief description of the cause of failure.

	In Distribution						**Out of Distribution**								
Location	Sherlock North			Woolridge Square			Sherlock South			Trader Joes			Hemphill Park		
Method	AST ↓	%S↑	NIR ↓	AST ↓	%S↑	NIR ↓	AST ↓	%S↑	NIR ↓	AST ↓	%S↑	NIR ↓	AST ↓	%S↑	NIR ↓
Geometric Only	17.64	61.60	11.21	18.56	86.36	9.50	15.56	92.50	7.80	15.22	94.74	12.05	14.21	95.00	13.21
PACER+G [3]	25.79	96.15	9.42	26.51	90.90	8.83	22.11	95.00	7.67	24.38	87.14	17.93	23.94	92.85	9.47
ViNT [8]	20.38	65.51	10.36	17.85	90.36	7.94	17.06	92.30	7.19	22.10	84.21	18.36	23.92	95.00	10.96
PIVOT [11]	46.56	83.33	26.03	57.36	84.11	20.22	45.41	75.00	21.91	41.77	74.44	40.67	33.85	85.00	28.36
CRESTE - cfs - st	21.26	96.15	12.99	20.53	91.66	9.79	21.13	92.80	5.65	23.17	93.75	14.87	26.85	94.11	11.11
CRESTE - cfs	25.86	96.29	9.20	19.84	90.90	6.12	13.27	92.30	2.41	25.39	91.66	10.49	28.43	83.33	8.57
CRESTE - st	19.09	96.29	6.10	17.80	92.90	3.03	12.88	94.65	2.04	22.01	92.85	7.83	16.32	93.42	2.16
CRESTE (ours)	**14.01**	**96.60**	**4.60**	**14.70**	**100.0**	**0.00**	**12.60**	**95.23**	**0.86**	**15.28**	**95.65**	**3.73**	**15.42**	**100.0**	**0.00**

TABLE I: Quantitative evaluation comparing CRESTE against existing navigation baselines for short horizon mapless navigation experiments. We bold the best performing method for each metric in each location, and annotate each metric with an up or down arrow to indicate if higher or lower numbers are better.

Location	Mueller Loop				
Method	AST ↓	%S↑	NIR ↓	Dist. (m) ↑	Total Int. ↓
PACER+G [3]	15.8	61.53	3.56	1345.07	48
CRESTE (ours)	**12.94**	**99.45**	**0.052**	**1919.44**	**1**

TABLE II: Quantitative evaluation for 1.9km long horizon mapless navigation experiments. We bold the best performing method for each metric in each location, and annotate each metric with an up or down arrow to indicate if higher or lower numbers are better.

environments and five times more interventions in unseen environments. Furthermore, we find that PIVOT, our VLM-based navigation baseline, performs poorly relative to other methods, corroborating our claim that VLMs and LLMs are not well attuned for urban navigation despite containing internet-scale priors. We hypothesize this is because navigation requires identifying which priors are most important for navigation, a task seen rarely by VLMs and LLMs during pre-training. Notably, we achieve an intervention-free traversal in one unseen environment (Hemphill Park), a residential park with diverse terrains, narrow curb gaps, and crosswalk markings. From these findings, we conclude that CRESTE is remarkably more generalizable for mapless urban navigation than existing approaches.

Ablating our proposed contributions, we observe an intervention reduction of 41% with structured representations and 70% reduction with counterfactuals. We hypothesize that structured representations improve performance by encoding higher-level features that are more generalizable and easier for policies to reason about compared to lower-level features that capture less generalizable high-frequency information. Our counterfactual ablations distill the same VFM features, thus vindicating our claim that even with sufficiently informative perceptual representations, it is important to leverage our counterfactual-based objective to reason about the most salient features for navigation. We conclude that our contributions in question are highly effective, enabling CRESTE to perform mapless urban navigation and robustly generalize to diverse environments beyond the training data.

IV. Limitations and Future Work

While CRESTE inherits some viewpoint-invariance from VFMs, it does not allow reward maps to be conditioned on embodiment-specific constraints, such as the traversability differences between quadrupeds and wheeled robots. An interesting future direction is applying existing research on cross-embodiment conditioning [8] to our architecture. Furthermore, CRESTE infers reward maps using observations from a single timestep, limiting multi-step horizon reasoning tasks, such as recovering from dead ends or negotiating paths in constricted environments. Extending CRESTE with memory and counterfactual IRL for reasoning about object dynamics are promising directions for addressing these issues. In addition, reward learning with counterfactuals is a potential bottleneck, as it requires humans for counterfactual labeling. Automated counterfactual generation with VLMs/LLMs is another promising direction to further improve scalability. Lastly, our counterfactual IRL objective can further be extended to incorporate preferences following the same derivation from the ranking perspective. This would not only allow the reward to reason about what paths are best but allow is to learn how to discriminate between two suboptimal paths should the need arise.

V. Conclusion

In this paper, we introduce Counterfactuals for Reward Enhancement with Structured Embeddings (CRESTE), a scalable framework for learning representations and policies that address the open-set generalization and robustness challenges of open-world mapless navigation. CRESTE learns robust, generalizable perceptual representations with internet-scale semantic, geometric, and entity priors by distilling features from multiple visual foundation models. We demonstrate that our perceptual representation encodes a sufficient set of factors for urban navigation, and introduce a novel counterfactual-based loss and active learning framework that teaches policies to hone in on the most important factors and infer how they influence fine-grained navigation behavior. These contributions culminate to form CRESTE, a local path planning module that significantly outperforms state-of-the-art alternatives on the task of long-horizon mapless urban navigation. Through kilometer-scale real-world robot experiments, we demonstrate our approach's effectiveness, gracefully navigating an unseen 2 kilometer urban environment with only a handful of interventions.

Acknowledgment

This work has taken place in the Autonomous Mobile Robotics Laboratory (AMRL) and Machine Decision-making through Interaction Laboratory (MIDI) at UT Austin. AMRL research is supported in part by NSF (CAREER-2046955, PARTNER-2402650) and ARO (W911NF-24-2-0025). MIDI research is supported in part by NSF (CAREER-2340651, PARTNER-2402650), DARPA (HR00112490431), and ARO (W911NF-24-1-0193). Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] J. Biswas and M. Veloso, "The 1,000-km challenge: Insights and quantitative and qualitative results," *IEEE Intelligent Systems*, vol. 31, no. 3, pp. 86–96, 2016.

[2] S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte *et al.*, "Minerva: A second-generation museum tour-guide robot," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 3. IEEE, 1999.

[3] L. Mao, G. Warnell, P. Stone, and J. Biswas, "Pacer: Preference-conditioned all-terrain costmap generation," *arXiv preprint arXiv:2410.23488*, 2024.

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

[5] X. Meng, N. Hatch, A. Lambert, A. Li, N. Wagener, M. Schmittle, J. Lee, W. Yuan, Z. Chen, S. Deng *et al.*, "Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation," *arXiv preprint arXiv:2303.15771*, 2023.

[6] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, "Viplanner: Visual semantic imperative learning for local navigation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5243–5249.

[7] D. Shah and S. Levine, "Viking: Vision-based kilometer-scale navigation with geographic hints," *arXiv preprint arXiv:2202.11271*, 2022.

[8] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "Vint: A foundation model for visual navigation," *arXiv preprint arXiv:2306.14846*, 2023.

[9] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "Ving: Learning open-world navigation with visual goals," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 215–13 222.

[10] K. Weerakoon, M. Elnoor, G. Seneviratne, V. Rajagopal, S. H. Arul, J. Liang, M. K. M. Jaffar, and D. Manocha, "Behav: Behavioral rule guided autonomy using vlms for robot navigation in outdoor scenes," *arXiv preprint arXiv:2409.16484*, 2024.

[11] S. Nasiriany, F. Xia, W. Yu, T. Xiao, J. Liang, I. Dasgupta, A. Xie, D. Driess, A. Wahid, Z. Xu *et al.*, "Pivot: Iterative visual prompting elicits actionable knowledge for vlms," *arXiv preprint arXiv:2402.07872*, 2024.

[12] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[13] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson *et al.*,

"Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[14] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," *arXiv:2304.02643*, 2023.

[15] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[16] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

[17] M. Ranzinger, G. Heinrich, J. Kautz, and P. Molchanov, "Am-radio: Agglomerative vision foundation model reduce all domains into one," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 12 490–12 500.

[18] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Advances in neural information processing systems*, vol. 33, pp. 18 661–18 673, 2020.

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

[20] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.

[21] Y. J. Ma, A. Shen, D. Jayaraman, and O. Bastani, "Smodice: Versatile offline imitation learning via state occupancy matching," *arXiv preprint arXiv:2202.02433*, vol. 1, no. 2, p. 3, 2022.

[22] B. et al., "Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences," *The International Journal of Robotics Research*, vol. 41, no. 1, pp. 45–67, 2022.

[23] D. Brown, W. Goo, P. Nagarajan, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International conference on machine learning*. PMLR, 2019, pp. 783–792.

[24] C. Premebida, L. Garrote, A. Asvadi, A. P. Ribeiro, and U. Nunes, "High-resolution lidar-based depth mapping using bilateral filter," in *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*. IEEE, 2016, pp. 2469–2474.

[25] A. Ošep, T. Meinhardt, F. Ferroni, N. Peri, D. Ramanan, and L. Leal-Taixé, "Better call sal: Towards learning to segment anything in lidar," in *European Conference on Computer Vision*. Springer, 2025, pp. 71–90.

[26] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[27] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[28] T. Ni, H. Sikchi, Y. Wang, T. Gupta, L. Lee, and B. Eysenbach, "f-irl: Inverse reinforcement learning via state marginal matching," in *Conference on Robot Learning*. PMLR, 2021, pp. 529–551.

[29] H. Sikchi, A. Saran, W. Goo, and S. Niekum, "A ranking game for imitation learning," *arXiv preprint arXiv:2202.03481*, 2022.

[30] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[32] A. Tamar, Y. Wu, G. Thomas, S. Levine, and P. Abbeel, "Value iteration networks," *Advances in neural information processing systems*, vol. 29, 2016.

[33] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," *arXiv preprint arXiv:1507.04888*, 2015.

[34] I. Loshchilov, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

VI. APPENDIX

In this section, we provide additional implementation details on our perceptual encoder Θ and label generation procedure in Sec. VI-A, Counterfactual IRL and counterfactual annotation tool in Sec. VI-B, model training details Sec. VI-C, and qualitative analysis comparing each baseline in Sec. VI-D.

A. Perceptual Encoder Implementation Details

Distillation Label Generator. Fig. 2 visually depicts the inputs and outputs for our label generator. Using sequential SE(3) robot poses and synchronized RGB–point cloud pairs ($o_{\text{rgb, 1:t}}$, $o_{\text{cloud, 1:t}}$), our Distillation Label Generator produces five training labels for supervising Θ. Below, we detail the label generation procedure for a single timestep t, separating it into two parts: (i) labels supervising f_{rgbd}, and (ii) labels supervising f_{bev}.

i) Generating Training Labels for the RGB-D Encoder f_{rgbd}. We generate $\hat{y}_{\text{semantic}}$ by passing $o_{\text{rgb, t}}$ through a frozen Dinov2 encoder and bilinearly interpolating the spatial dimension of our output feature map to match the spatial resolution of y_{semantic}. We generate $\hat{y}_{\text{depth}}$ by projecting $o_{\text{cloud, t}}$

to the image and applying bilateral filtering [24] for edge-aware inpainting, producing a dense depth map that respects object boundaries.

ii) Generating Training Labels for the BEV Inpainting Backbone f_{bev}. To generate $\hat{y}_{\text{bev, dynamic}}$, we prompt SAM2 with bounding box detections from a set of commonly encountered dynamic categories (e.g. vehicles, pedestrians) to obtain dynamic entity labels. We follow the approach in Osep. et al. [25], backprojecting dynamic labels to 3D using the corresponding point cloud $o_{\text{cloud, t}}$ and clustering the points at multiple density thresholds using DBSCAN [26]. We retain the DBSCAN clusters that exceed a minimum intersection-over-union (IoU) overlap with the entity-labeled clusters and project the matched points to the dynamic entity BEV map.

To generate $\hat{y}_{\text{bev, static}}$, we first obtain static entity labels by prompting SAM2 with a grid of query points (generating dynamic entity labels via bounding box queries as before), mask out overlapping regions between grid-queried and dynamic labels to isolate the static masks in each frame, and then apply an iterative greedy merging strategy that fuses overlapping masks (above an IoU threshold) across consecutive frames — treating unmatched IDs as new entities. Finally, we project and accumulate these entity-consistent multi-frame labels in BEV using known robot poses to produce the final static entity BEV map.

To generate $\hat{y}_{\text{bev, elev}}$, we accumulate static entity-labeled 3D points across sequential frames using robot poses (generating static entity labels as before), assign each 3D point to a grid cell, and compute each cell's minimum elevation by averaging the N lowest 3D points that fall into that cell.

B. Counterfactual IRL Details

Generating Counterfactual Annotations. To obtain counterfactual annotations for a training sample t in our dataset, we first require the expert trajectory Γ_t^E, $o_{\text{rgb, t}}$, and $o_{\text{depth, t}}$ for the sample. We uniformly sample a handful of "control" states along Γ_t^E, excluding the start and goal states. Then, we randomly perturb these control states before planning a kinematically feasible path from the start and goal states such that it reaches all of the perturbed control states. Practically, we implement this using Hybrid A* [20] - however, any kinematic planner will suffice. We use the counterfactual annotation tool shown in Fig. 6 to generate a handful of alternate trajectories in this manner and annotate the suboptimal trajectories given $o_{\text{rgb, t}}$ and $o_{\text{depth, t}}$. We use these counterfactuals to retrain r_ϕ in our active reward learning framework. In total, we train CRESTE using 600 annotated samples with 2-3 counterfactuals per sample, which is roughly 3% of all frames in the training dataset.

Counterfactual IRL Derivation $\mathcal{L}_{\text{IRL}}$. IRL methods [27, 28] allow for learning reward functions r_ϕ, parameterized by ϕ, given expert demonstrations. However, they provide no mechanism to incorporate suboptimal trajectories. Suboptimal trajectories are easy to obtain and enable a data flywheel for navigation; the BEV observations obtained from expert runs can simply be relabelled with suboptimal of

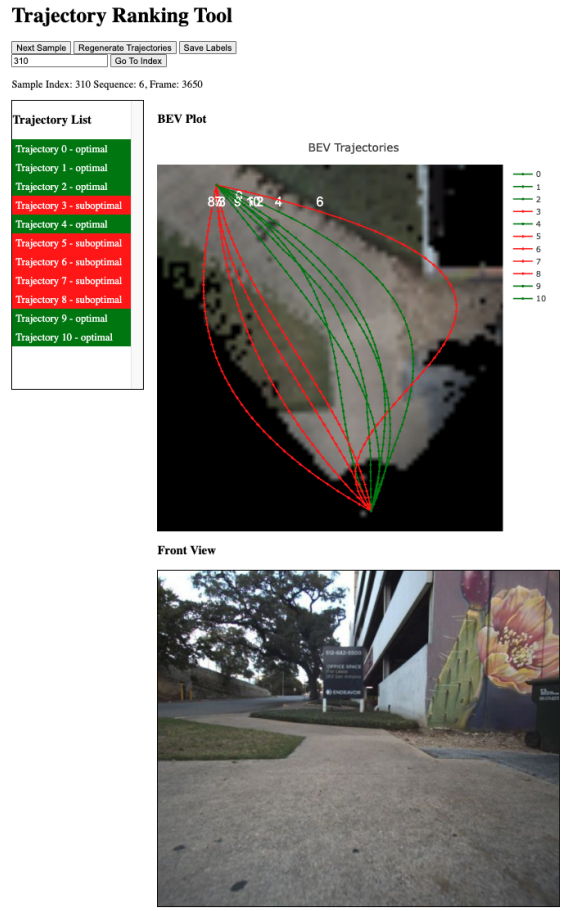

Fig. 6: Counterfactual annotation tool used for training the reward function r_ϕ. We provide the front-view RGB image and BEV RGB image to the human annotator for context. We highlight the counterfactuals in red and acceptable trajectories in green.

unsafe trajectories offline without any more environmental interactions. Motivated by this idea, we derive $\mathcal{L}_{\text{IRL}}$, a general and principled way to learn from suboptimal and expert trajectories jointly under a single objective.

Our approach builds on the ranking perspective of imitation learning [29] which uses visitation distributions to denote long-term behavior of an agent. We denote $\rho^\pi(s,a)$, $\rho^E(s,a)$, $\rho^S(s,a)$ to be the agent, expert, and suboptimal state-action visitation distributions respectively. We assume the reward function is conditioned on y_{bev} as before, but drop it from the derivation for conciseness. Under this notation the problem of return maximization becomes finding a visitation induced by a policy π that maximizes the expected return given by:

$$\max_\pi J^\pi(r_\phi) = \max_\pi \mathbb{E}_{\rho^\pi(s,a)}[r_\phi(s,a)]. \tag{5}$$

The reward function of the expert should satisfy the ranking $\rho^\pi(s,a) \preceq \rho^E(s,a)$, which implies that the expert's visitation distribution obtains a return that is greater or equal to any other policy's visitation distribution in the environment:

$$\rho^\pi(s,a) \preceq \rho^E(s,a) \implies \mathbb{E}_{\rho^\pi(s,a)}[r_\phi(s,a)] \leq \mathbb{E}_{\rho^E(s,a)}[r_\phi(s,a)]. \tag{6}$$

This property extends to any suboptimal visitations, and as a consequence of linearity of expectations, to any convex combination of the current policy's visitation and any other suboptimal visitation distribution. Mathematically,

$$\alpha\rho^{\pi}(s,a) + (1-\alpha)\rho^{S}(s,a) \preceq \rho^{E}(s,a) \ \forall \alpha \in [0,1]. \quad (7)$$

Thus, given suboptimal visitation distributions, we can create a number of pairwise preferences by choosing a suboptimal visitation and a particular α. We turn to the Bradley-Terry model of preferences to satisfy these pairwise preferences which assumes that preferences are noisy-rational and that the probability of a preference can be expressed as:

$$\begin{aligned} &P(\rho^{E}(s,a) \succeq \alpha\rho^{\pi}(s,a) + (1-\alpha)\rho^{S}(s,a)) = \\ &\frac{e^{J^{E}(r_\phi)}}{e^{J^{E}(r_\phi)} + e^{\alpha J^{\pi}(r_\phi)+(1-\alpha)J^{S}(r_\phi)}} \\ &= \frac{1}{1 + e^{\alpha(J^{S}(r_\phi)-J^{E}(r_\phi))+(1-\alpha)(J^{\pi}(r_\phi)-J^{E}(r_\phi))}}. \end{aligned} \quad (8)$$

Finding a reward function implies maximizing the likelihood of observed preferences while the policy optimizes the learned reward function. Since, the convex combination holds for all values of $\alpha \in [0,1]$, we consider optimizing against the worst-case to obtain the following two-player counterfactual IRL objective, where the reward player learns to satisfy rankings against the worst-possible α:

$$\max_{\phi} \min_{\alpha} P\left(\rho^{E}(s,a) \succeq \alpha\rho^{\pi}(s,a) + (1-\alpha)\rho^{S}(s,a)\right) \quad (9)$$

and the policy player maximizes expected return:

$$\max_{\pi} J^{\pi}(r_\phi). \quad (10)$$

In practice, optimizing for worst-case α for each BEV scene y_{bev} can quickly make solving the optimization objective challenging due to the large number of scenes we train the reward function on. We make two mild approximations that we observed to make learning more efficient and tractable: First, we replace the worst-case α with a fixed α, and second, we consider maximizing a pointwise monotonic transformation to the Bradley Terry loss function that directly maximizes $\alpha(J^{S}(r_\phi)-J^{E}(r_\phi))+(1-\alpha)(J^{\pi}(r_\phi)-J^{E}(r_\phi))$ instead of its sigmoid transformation. With these changes, we can rewrite our practical counterfactual IRL objective as:

$$\min_{\pi} \max_{\phi} \left(J^{E}(r_\phi) - (\alpha J^{S}(r_\phi) + (1-\alpha)J^{\pi}(r_\phi))\right). \quad (11)$$

This objective reveals a deeper connection between apprenticeship learning (Eq 6 [30]) obtained by setting α to 0 and learning from preferences [23] obtained by setting α to 1. The loss function goes beyond the apprenticeship learning objective that only learns from expert by incorporating suboptimal demonstrations. Second, it goes beyond the offline nature of prior algorithms that learn from preferences alone by instead learning a policy that attempts to match expert visitation making use of the suboptimal demonstrations.

Fig. 7: Mobile robot, Clearpath Jackal, used for real-world experiments with sensors for deployment annotated.

C. Training Details

In this section, we supplement the implementation details for CRESTE and each baseline described in Sec. III.

1) CRESTE: We warm-start the RGB-D backbone for 50 epochs or until convergence. We set α_1 and α_2, the loss weights for semantic feature regression and completion, to 1 and 0.5, respectively, and keep this fixed for the rest of training. After this, we freeze the RGB-D backbone and train the f_{bev} using the BEV inpainting backbone loss $\mathcal{L}_{\text{bev}}$ for five epochs before unfreezing the RGB-D backbone and training end-to-end for another 45 epochs or until convergence. This enables faster convergence by stabilizing f_{bev} before joint training. We set β_1, β_2, and β_3 to 1, 2, and 3 for $\mathcal{L}_{\text{bev}}$, but find that training remains stable for any reasonable combination of values. Finally, we empirically find that replacing the Supervised Contrastive loss [18] with Cross Entropy loss while training the dynamic panoptic head $f_{\text{bev, dynamic}}$ empirically improves overall performance. Thus, we use this model for our long-horizon navigation experiments presented in Table II. We summarize additional model architecture details in Table III and training hyperparameters in Table IV.

As mentioned in Sec. II-B.2, we train our reward function r_ϕ in two phases. During both phases, we freeze Θ, thus only training r_ϕ. During initialization (Phase I), we train for 25 epochs, setting $\alpha = 0$ only to use expert demonstrations. During finetuning (Phase III), we initialize r_ϕ from scratch, train for 50 epochs, setting $\alpha = 0.5$ and a reward regularization term [21] $\alpha_{reg} = 1.0$. Empirically, we find that our objective is stable for a range of α values and benefits from a larger reward regularization penalty to encourage more discriminative rewards. In total, we train the perceptual encoder Θ and reward function r_ϕ for a combined 150 epochs.

2) ViNT [8]: To modify ViNT to follow XY goal guidance instead of image goal guidance, we follow the architecture design details in the ViNT paper for adapting the backbone to GPS goal guidance. We only train the XY goal encoder and freeze the remaining model parameters. We sample XY goals from future odometry between 8 to 10 seconds to generate training data.

3) PIVOT [11]: We condition PIVOT using an annotated satellite view image, an annotated front view image with numbers indicating goal candidates, and text instructions. We annotate a satellite image containing the robot's current position, heading, and next goal GPS coordinates. Finally, we provide the text prompt below:

> I am a wheeled robot that cannot go over objects. This is the image I'm seeing right now. I have

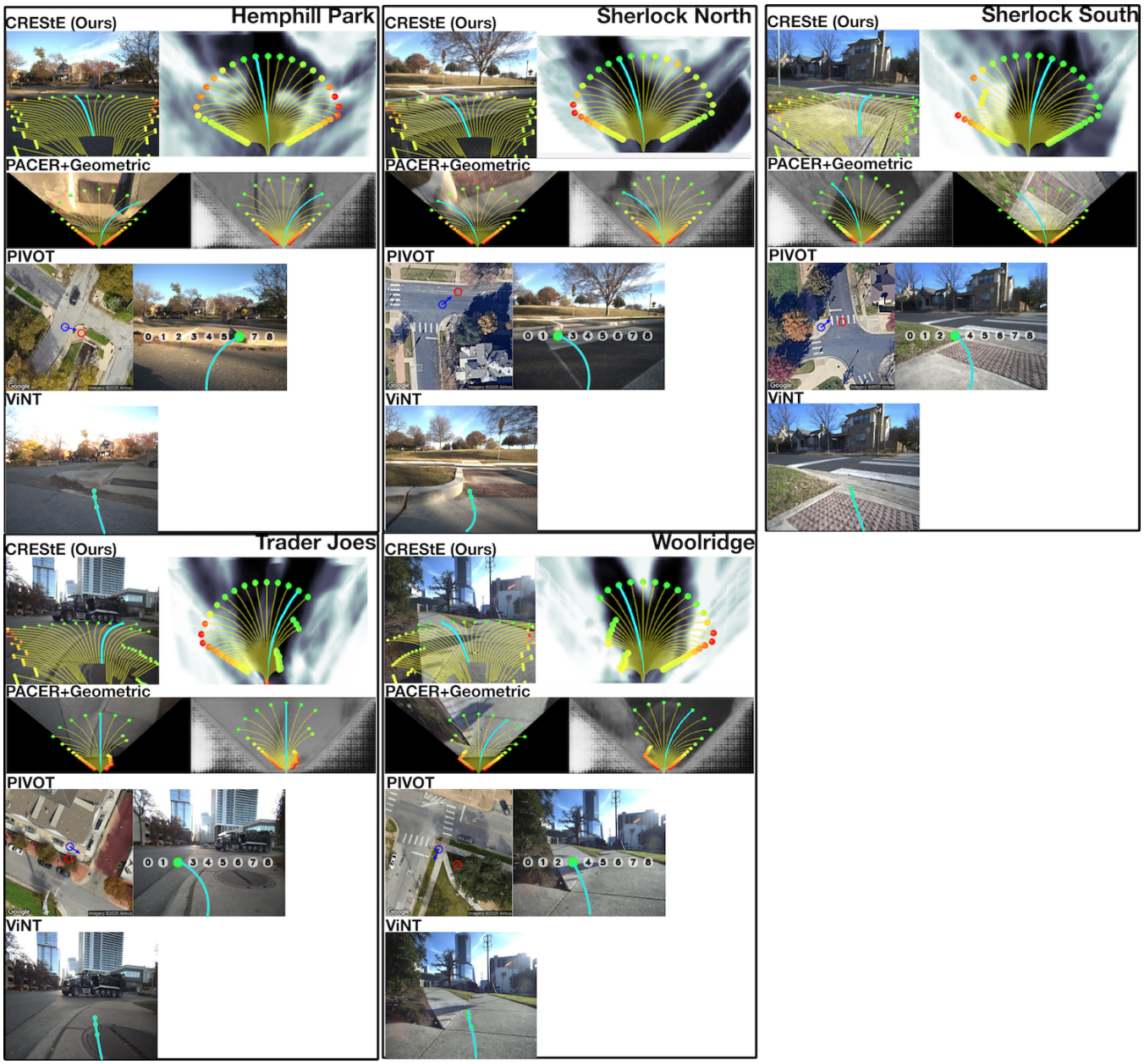

Fig. 8: Qualitative comparison of local paths planned by different learned baselines in different geoegraphic locations: Hemphill Park, Sherlock North, Sherlock South, Trader joes, Woolridge. We visualize each chosen path either in bird's eye view (BEV) or on the front view RGB image in aqua blue. For PACER+Geometric [3], we provide the BEV image used by the model. For PIVOT [11], we show the annotated satellite image and front view image used to prompt the VLM. For ViNT [8], we front view RGB image given as input. For more information regarding each baseline, we refer readers to Appendix Sec. VI-D.1.

> annotated it with numbered circles. Each number represents a general direction I can follow. I have annotated a satellite image with my current location and direction as a red circle and arrow and the goal location as a blue circle. Now you are a five-time world champion navigation agent and your task is to tell me which circle I should pick for the task of: going forward **X** degrees to the **Y**? Choose the best candidate number. Do NOT choose routes that go through objects AND STAY AS FAR AS POSSIBLE AWAY from untraversable terrains and regions. Skip analysis and provide your answer at the end in a json file of this form: "points": []

where X is replaced by the degrees from the goal computed using the magnetometer heading and goal GPS location, and Y is either left or right depending on the direction of the goal. During testing, we use our geometric obstacle avoidance module to safely reach the current goal selected by PIVOT.

4) PACER [3]: PACER is a terrain-aware model that predicts BEV costmaps from BEV images. It requires pre-enumerating an image context that specifies the preference ordering for terrains. In all environments, we would like the

TABLE III: Architecture Hyperparameters for CRESTE.

Hyperparameter	Value
RGB-D Encoder (f_{rgbd})	
Base Architecture	EfficientNet-B0 [15]
Number of Input Channels	4
Input Spatial Resolution	512×612
Output Spatial Resolution	128×153
Output Channel Dimension	256
Semantic Decoder Head (f_{semantic})	
Input Spatial Resolution	128×153
Input Channel Dimension	256
Output Spatial Resolution	128×153
Output Channel Dimension	128
Number of Hidden Layers	4
Depth Completion Head (f_{depth})	
Input Spatial Resolution	128×153
Input Channel Dimension	256
Output Spatial Resolution	128×153
Number of Depth Bins	128
Number of Hidden Layer	2
Depth Discretization Method	Uniform
Number of Intermediate Layers	2
Lift Splat Module (f_{splat})	
Total Input Channel Dimension	288
Input Semantic Channel Dimension	256
Input Depth Channel Dimension	32
Map Cell Resolution	$0.1m \times 0.1m \times 3$
Output Channel Dimension	96
Output Spatial Resolution	256×256
BEV Inpainting Backbone (f_{bev})	
Base Architecture	ResNet18 [31]
Input Channel Dimension	96
Input Spatial Resolution	256×256
Output Static Panoptic Dimension	32
Output Dynamic Panoptic Dimension	32
Output Elevation Dimension	1
Reward Function (r_ϕ)	
Base Policy Architecture	Value Iteration Network [32]
Base Reward Architecture	MultiScale-FCN [33]
Input Spatial Resolution	256×256
Input Channel Dimension	65
Prepool Channel Dimensions	[64, 32]
Skip Connection Channel Dimensions	[32, 16]
Trunk Channel Dimensions	[32, 32]
Postpool Channel Dimensions	[48]
# Future Actions	50
# Actions Per State	8
Discount Factor	0.99

TABLE IV: Training Hyperparameters for CRESTE.

Hyperparameter	Value
RGB-D Backbone Training (f_{rgbd})	
Semantic Loss Weight (α_1)	2.0
Depth Completion Loss Weight (α_2)	0.5
# Training Epochs	50
Batch Size	12
Optimizer	AdamW [34]
Adam β_1	0.9
Adam β_2	0.999
Learning Rate	5×10^{-3}
Learning Rate Scheduler	Exponential
Learning Rate Gamma Decay γ	0.98
GPU Training Hours	10 Hours
GPUS Used	3 Nvidia H100 GPUs
BEV Inpainting Backbone Training (f_{bev})	
Static Panoptic Loss Weight (β_1)	1.0
Dynamic Panoptic Loss Weight (β_2)	2.0
Elevation Loss Weight (β_3)	3.0
Warmup Epochs	5
# Training Epochs	50
Batch Size	24
Optimizer	AdamW [34]
Adam β_1	0.9
Adam β_2	0.999
Learning Rate	5×10^{-4}
Learning Rate Scheduler	Exponential
Learning Rate Gamma Decay γ	0.98
GPU Training Hours	24 Hours
GPUS Used	3 Nvidia H100 GPUs
Reward Function Training (r_ϕ)	
Batch Size	30
Optimizer	AdamW [34]
Adam β_1	0.9
Adam β_2	0.999
Learning Rate	5×10^{-4}
Learning Rate Scheduler	Exponential
Learning Rate Gamma Decay γ	0.96
Reward Learning Weight $\mathcal{L}_{\text{IRL}}$	1.0
Reward Smoothness Penalty Weight	4.0
GPU Training Hours	2 Hours
GPUS Used	3 Nvidia H100 GPUs

robot to prefer traversing terrains in the following preference order: sidewalk, dirt, grass, rocks. Thus, we condition PACER on this preference order for all environments.

D. Qualitative Analysis

In this section, we present a qualitative analysis of the short-horizon experiments for each location. For each experiment area illustrated in Fig. 4, we compare the planned path for each learned baseline in approximately the same location.

1) Comparing CRESTE *Against Existing Baselines:* Fig. 8 provides additional context to understand the inputs given to each baseline and the planned path, which we describe as follows: 1) CRESTE - The input RGB and sparse depth image (not shown) and predicted BEV cost map where darker regions correspond to low cost, 2) PACER+Geometric (PACER+G) [3] - The input BEV image and predicted BEV cost map where darker regions correspond to lower cost, 3) PIVOT [11] - The annotated satellite image with the robot's current location (blue circle), heading (blue arrow), and goal location (red circle). The front view RGB image annotated with numbered circles for prompting the VLM along with the chosen circle highlighted in green, 4) ViNT [8] - The front view image annotated with the local path waypoints predicted by ViNT.

Analyzing each model, we find that PACER+G struggles to infer the cost for terrains that are underrepresented in the training dataset, such as dome mats. Furthermore, when two terrains look visually similar, such as the sidewalk and road pavement for the Trader Joes location, PACER infers the costs incorrectly. PIVOT can select the correct local waypoint in scenes with a large margin for error, but struggles at dealing with curb cuts (Hemphill Park, Sherlock North) and narrow sidewalks (Trader Joes). Long latency between VLM queries negatively impacts the model's ability to compensate for noisy odometry. This effect is particularly apparent in narrow corridors or sidewalks where even minor deviation from the straight line path results in failure. ViNT can successfully maintain course in straight corridors and sidewalks, but struggles to consider factors like curb cuts and terrains. We hypothesize this is because our dataset is

not sufficiently large for learning generalizable features from expert demonstrations alone. Furthermore, demonstrations with straight line paths dominate our dataset, making it difficult for behavior cloning methods like ViNT to learn which factors influence the expert to diverge from the straight line path.