2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Beyond Waypoints: Semantic-Centric Autonomy with Unreliable Maps Through Learned Abstractions

Akila Saravanan[1], Songyuan Zhang[1], Travis Manderson[1], Nicholas Roy[1], Chuchu Fan[1]

***Abstract*— Autonomous navigation that relies on precise metric maps is inherently fragile to environmental changes and mapping inaccuracies. These discrepancies often lead to failures in localization and path planning, as the robot's internal representation of the world no longer matches reality. We propose an alternative navigation approach that instead focuses on how a robot interacts with its surroundings rather than its precise metric position. Our core contribution is a learned behavioral vocabulary conditioned on raw sensor data that can be used to compose plans for navigation. Our system transforms LiDAR data into low-dimensional learned embeddings which are clustered to create a set of abstract, human-interpretable behaviors (e.g., along wall, exiting intersection, on bridge). This representation allows the robot to control its behavior with respect to the embedding rather than controlling its state with respect to a specific metric cost function or waypoint, thereby minimizing the impact of map and position inaccuracies. We define the mission as a topological sequence of behavioral clusters on the overhead map, enabling high-level navigation. This approach provides a robust way to decompose the environment into recognizable and actionable states that can reliably compose a plan, even on stale maps with environmental deformations and world changes. Our method achieves higher navigation success under intentional map distortions, with average mission success rates 53 and 55 percentage points higher for short and long term plans respectively when compared to baselines which rely on accurate metric maps.**

I. Introduction

Navigating large outdoor environments with inaccurate maps remains a significant hurdle for autonomous agents. While most structured path planning and goal-conditioned RL assume metrically accurate environment representations [1], this assumption fails in real-world applications like disaster relief or extraterrestrial exploration, where maps are often stale or unavailable [2]. Traditional coordinate-based navigation fails when overhead maps (e.g., satellite or UAV imagery) contain spatial offsets, noise, or partial information. As shown in Figure 1, an angular misalignment can cause a waypoint-following robot to veer off a bridge and into the water, even if the local path appears physically navigable.

To address these navigation failures, we introduce a novel behavioral-segmentation approach that redefines navigation goals not as precise coordinates, but as semantic, behavioral transitions. While some navigation systems operate without a high-fidelity map, they often rely on simple waypoint sequences or local reactive behaviors. Our work addresses a more complex problem: how to execute a high-level plan when the provided map is imprecise, containing spatial offsets and inaccuracies that render traditional, coordinate-based navigation unreliable. This problem is particularly acute when using overhead maps, like unmanned aerial vehicle and satellite imagery, which can be misaligned and contain significant noise and partial information.

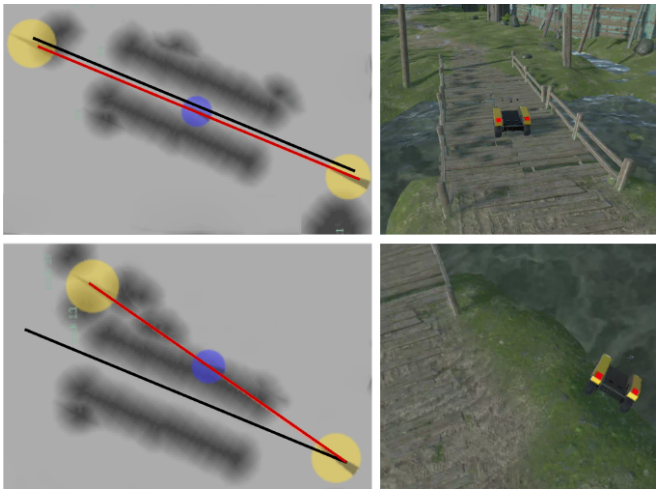

Fig. 1: Trajectories from the overhead map (red) vs ground truth (black). Yellow circles indicate the start and goal. Blue circles are intermediate waypoints. Top: successful "cross bridge" mission with aligned map. Bottom: failure due to the misalignment between overhead map and real world.

Our system's operation involves two key phases. During training, our method transforms sensor data over time into a low-dimensional learned embedding. These embeddings are then clustered to create a vocabulary of concepts that are associated with human-interpretable behaviors. During execution, the robot is given a high-level plan as a topological sequence of behavioral clusters and the bearing between each pair of clusters. It then navigates by transitioning between these segments, using the bearing as a high-level signal to guide its movement. Unlike traditional landmark-based navigation that relies on point-based features, we treat regions as spatially extended, semantic segments fundamental to a task. A bridge, for example, is a region with inherent behavioral phases—entering, on, and exiting—each having a distinct signature in the robot's sensor data. This approach fundamentally changes the problem of navigation from descriptive ("where am I and where should I go next?") to prescriptive ("what kind of place am I in, and what behavior should I execute here?"). Our key contributions are:

1) learning a behavioral vocabulary from LiDAR, enabling a robust representation of the environment, and
2) a new navigation framework that utilizes a high-level plan of behavioral transitions for successful mission completion even with imprecise maps.

[1]Massachusetts Institute of Technology {akilasar,szhang21,travism,nickroy,chuchu}@mit.edu

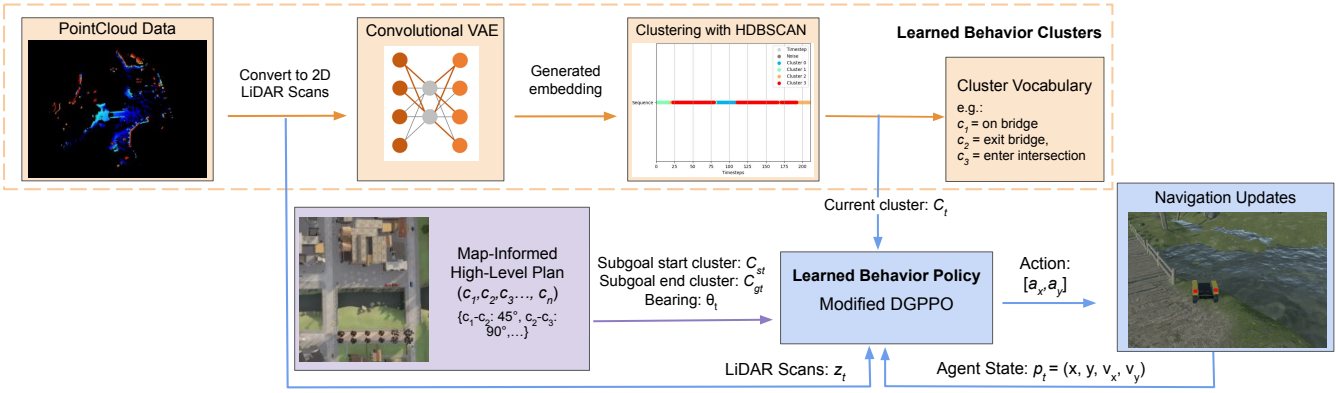

Fig. 2: This system addresses navigation in unstructured environments with two key components. The *Learned Behavior Clusters* component (orange dashed box) is trained once offline to create a cluster-based segmentation of the environment from LiDAR data. During execution, the trained models are used at every time step to provide real-time cluster identity. The *Learned Behavior Policy* (blue box) is trained offline and handles transitions between clusters. It is sequentially invoked at runtime for each cluster pair in the High-level Plan (purple box), eventually guiding the robot to its final goal.

Our system demonstrates improved performance over a waypoint-based baseline, achieving an average mission success rate 53 percentage points higher for short plans and 55 percentage points higher for long plans in highly distorted environments. The generalizability of our approach stems from the fact that its behavioral clusters remain consistent across different environments. Our modified Discrete GCBF Proximal Policy Optimization (DGPPO) [3] policy learns to navigate between pairs of these clusters, rather than specific, static routes. This allows the same policy to be executed on novel missions in unseen environments, where current sensor information is cross-referenced with the learned clusters to determine the robot's current behavioral state.

II. Problem Statement

Our goal is to enable autonomous agents to navigate in environments that are unstructured—lacking a precise global coordinate system or artificial fiducials—yet are semantically structured with features like bridges, buildings and intersections. This approach addresses the challenges of navigation in GPS-denied and poorly-mapped environments. The goal is to find a robust alternative to metric-based planning, which we will show fails under map inaccuracies, and end-to-end policy learning, which lacks generalizability to novel environments [4]. We frame the navigation problem as goal-directed Reinforcement Learning (RL), where subgoals are defined by their semantic grounding in the environment—for instance, navigating to or through regions such as "on a bridge" or "in an intersection"—rather than by specific metric positions. As a result, we must learn to identify and transition between semantic regions in the environment, which we refer to as behavioral clusters: $C_i \in \{C_0, C_1, \ldots, C_n\}$.

Although our clusters are derived from immediate sensor data, their membership can be determined for any point in the environment. This means the behavioral clusters implicitly create a spatial segmentation of the environment, which allows us to define a navigation plan as a sequence of these clusters. This segmentation extends to an overhead map $\mathcal{M}$ of the environment, where each cluster corresponds to a defined segment with its own spatial extent and centroid (G_i).

We assume access to a pre-segmented map of the environment into behavior clusters. This map functions as a source of topological information rather than as a tool for real-time metric localization. The map, which can be as abstract as a hand-drawn sketch or as concrete as a satellite image, provides the relative locations of semantic segments. This behavioral segmentation ultimately provides the basis for the high-level plan given to the robot to execute. The plan is composed of a sequence of m subgoals where each subgoal is a pair of clusters (C_{s_i}, C_{s_j}) and each cluster is obtained from an overhead map. The plan is defined as $\mathcal{P} = ((C_{s_1}, C_{g_1}), (C_{s_2}, C_{g_2}), \ldots, (C_{s_m}, C_{g_m}))$ where for each set of consecutive subgoals p and $p+1$, $C_{g_p} = C_{s_{p+1}}$. The key requirement is that the robot must be able to reliably recognize from sensor data when it is in a cluster C without resorting to a global position estimate, and then signal when it transitions from one semantic place to another. The robot must sequentially transition from each subgoal's start cluster to the corresponding goal cluster as defined in the plan. Finally, the goal is to learn a policy $a_t = \pi(s_t)$ that guides navigation. The policy maps the current state, s_t, to an action that controls the robot's motion, with the critical constraint that the state contains no metric data for localization or navigation. Instead, the policy must rely exclusively on information about behavioral clusters and a general bearing.

III. Technical Approach

A. System Architecture

The system architecture in Figure 2 has two components:

- *Learned Behavior Clusters*: A learned model mapping sensor data to cluster identity. We process Pointcloud data into 2D scans which are then passed to an autoencoder for training. The training of the autoencoder and clustering of the latent space embeddings from the encoder component of the autoencoder is performed once, offline (shown in the orange dashed box). During

online execution, the trained models operate at every time step, converting the robot's real-time sensor data into a cluster identity, which is fed into the next stage.

- *Learned Behavior Policy*: A learned model that encodes how to navigate from one cluster to another. As previously stated we want to learn a mapping of agent states to motion. The state of the robot at time t is defined by a tuple $s_t = (p_t, z_t, C_t, C_{s_t}, C_{g_t}, \theta_t)$, where $p_t = (x, y, v_x, v_y)$ is the current position and velocity of the robot, $z_t \in \mathbb{R}^L$ is the raw LiDAR scan, C_t is the robot's current behavioral cluster, C_{s_t} is the start cluster and C_{g_t} is the next (subgoal) cluster in the high-level plan, and $\theta_t \in [0, 2\pi]$ is the bearing between G_{s_t} and G_{g_t}. This bearing is in a world-aligned frame, but the system's reliance on it is only as a general signal, as navigation is primarily driven by behavioral transitions. The action space, $\mathcal{A}$, is a set of continuous acceleration commands, specifically accelerations along the x and y axes, denoted as a_x and a_y. The policy (shown in the blue box) is trained offline for one-to-one transitions between pairs of behavioral clusters. During execution, this learned policy is sequentially invoked for each cluster pair in the map-informed high-level plan (shown in the purple box), until the last cluster in the plan (representing the final destination) is reached.

Assumptions: We impose constraints on the types of high-level plans that are considered valid. The plan must be composed exclusively of the set of learned behavioral clusters as the system does not have a general policy for navigating between undefined or unknown segments. While we assume a segmented map, generating this is abstracted to a human commander or global planner using coarse overhead data from satellite imagery. Thus, the framework avoids requiring densely annotated prior maps, needing only a high-level topological sequence of expected geometries. The policy assumes forward-only motion to ensure correct behavioral transitions and requires valid sequential cluster pairs (e.g., transitioning directly from "enter bridge" to "exit bridge" is invalid). Furthermore, we do not consider navigation within large, open spaces where semantic cues are limited. Our system is designed to handle transitions into or out of open areas but assumes a greater reliance on the semantic cues available within the environments for successful navigation.

B. Learned Embeddings for Behavioral Clustering

LiDAR is our primary sensor due to its reliability across diverse conditions and its geometric-based decomposition of the environment. We use a Convolutional Variational Autoencoder (CVAE) [5] for dimensionality reduction, by learning a low-dimensional embedding that captures the salient features of the LiDAR scan.

The autoencoder learns an informative and generalizable embedding that encapsulates the key spatial characteristics of the environment. Let a raw LiDAR scan at time t be a vector $\mathbf{x}_t \in \mathbb{R}^L$, where L is the number of LiDAR beams. We map this to a low-dimensional embedding $\mathbf{z}_t \in \mathbb{R}^D$, with $D \ll L$. The encoder, f_ϕ, maps the input scan to a learned embedding: $\mathbf{z}_t = f_\phi(\mathbf{x}_t)$. The decoder, g_θ, attempts to reconstruct the original scan from the embedding: $\hat{\mathbf{x}}_t = g_\theta(\mathbf{z}_t)$. The network minimizes the reconstruction error using Mean Squared Error (MSE) loss. We empirically chose D=16 for the embedding, as we found it to be the smallest size that could accurately reconstruct the original LiDAR data, thereby preserving the essential information for our task.

The learned embeddings, $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$, are clustered using HDBSCAN [6]. This clustering is applied across the robot trajectory, segmenting the trajectory into distinct behavioral regions. A key motivation for this approach is that we want to learn a behaviorally meaningful embedding. While the autoencoder is trained purely for reconstruction, the inherent structure of the data is defined by the robot's interaction with the environment. As a result, the low-dimensional embedding space organizes data points with similar behavioral signatures close to each other. For instance, two different LiDAR scans of a robot approaching an intersection from the same direction, despite having minor variations, will be mapped to nearby points in the embedding space. This learned representation provides the ideal basis for clustering, going beyond simple dimensionality reduction to capture latent, semantically-relevant features. As shown in Figure 3, scans from different behavioral clusters highlight the distinct geometric features that define each region.

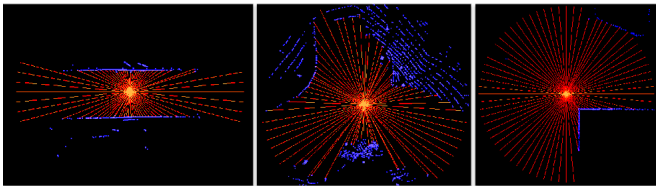

Fig. 3: Sample 2D LiDAR scans from distinct behavioral regions of the environment: "on a bridge" (left), "in intersection" (center), "around a corner" (right). These scans illustrate the unique geometric features that the autoencoder learns to represent in its low-dimensional embedding space.

We chose HDBSCAN for our clustering needs after considering common alternatives like K-Means and DBSCAN. K-Means was unsuitable due to its assumption of a predefined number of spherical clusters which is incompatible with the diverse and unknown number of behavioral modes in our data. DBSCAN's reliance on a single density threshold either merged distinct behaviors or failed to find sparse clusters, as our embeddings capture regions of both high density (e.g., traversing a well-defined bridge) and lower density (e.g., approaching an intersection from a distance). HDBSCAN is ideal because its hierarchical approach automatically determines the number of clusters and effectively identifies clusters across varying densities. The resulting clusters are a robust representation of the underlying behavioral states.

C. Semantic Interpretation and Behavioral Phase Extraction

We manually label each cluster with a semantic description, such as "on bridge" or "in intersection." These labels

create a fixed behavioral vocabulary that translates a commander's high-level goals into abstract, human-interpretable concepts like "approach," "traversal," and "exit." The system does not directly interpret natural language commands; instead, a user or a planner must convert a high-level goal (e.g., "cross the bridge") into a sequence of these pre-defined behavioral clusters. Crucially, execution relies on local LiDAR structural equivalence rather than strict semantic mapping. In our setup, outdoor bridges, urban alleyways and indoor hallway generate similar LiDAR signatures and are seamlessly processed under the same "on bridge" policy. Furthermore, manual annotation does not bottleneck scalability; incorporating novel environments simply requires sampling new LiDAR signatures to generate corresponding clusters. Since a finite set of structural geometries covers most navigable spaces, expanding this vocabulary will naturally generalize to unstructured environments. The behavioral clusters are not task-dependent; the same physical location can correspond to "enter intersection" and "exit intersection" cluster depending on the direction of travel as specified by the high-level plan.

A significant benefit of our system's architecture is its ability to relate concepts across different perspectives. Our approach explicitly connects the on-ground, egocentric perception of the robot (derived from LiDAR clusters) with the global, exocentric view from an overhead map. This capability is a crucial bonus that enables intuitive human-robot interaction and more flexible mission planning.

While operating, the robot continuously monitors its current behavioral cluster against the expected cluster prescribed by the high-level plan. At every time step, the trained policy is provided with the current, start, and end behavioral clusters of the ongoing sub-task, allowing the low-level policy to focus on a single adjacent pair of tasks, which makes learning the policy easier and generalizable to other environments that can be composed in the same way. This breakdown allows for the decomposition of a complex mission into phases that can be inferred from the sensor data, enabling navigational corrections when the ground truth is different.

D. Controller Implementation

Our autonomous navigation policy is a graph neural network trained using the Discrete Graph Proximal Policy Optimization (DGPPO) framework [3]. DGPPO provides a structured state representation and a dual-objective training process that has mathematical guarantees on safety. The policy integrates a discrete graph control barrier function (DGCBF) that provides an explicit mechanism for collision avoidance, which is especially important since the use case of this work is environments with inaccurate mapping.

1) Environment and State Representation: The system state is represented as a graph, where nodes encode obstacles derived from LiDAR data and agent features. Agent nodes include the complete set of state information from s_t, described in Section II. The obstacle nodes are composed of the eight closest LiDAR scans, a design choice made to reduce the size of the graph for computational efficiency.

Crucially, we do not provide the policy with explicit (x, y) coordinate information or a direct edge connecting the agent to the geometric goal. Instead, the agent learns to execute a sequence of behaviors to achieve its objective. Since the policy's inputs are decoupled from metric values, it instead relies on relative geometric information (from LiDAR), behavioral context (from the cluster), and a static global bearing. This design choice pushes the agent to learn a high-level behavioral strategy rather than exploit cues local to the environment, enabling generalization across environments. The policy is trained jointly over all transitions per landmark. If new cluster pairs are added, the policy must be retrained.

2) Reward and Cost Function: The controller is trained with a dual reward and cost structure that first prioritizes safety and then optimizes task completion. The reward function, defined as $R = R_\theta + B_{goal} + B_{stay} - P_{wrong} + P_{vel}$, provides a strong signal that guides the agent through the behavioral sequence. The components are listed below:

Term	Details
R_θ	Dense reward proportional to the alignment between agent's velocity vector and target bearing. Guides the agent's movement direction until it reaches the target cluster.
B_{goal}	Large, one-time bonus awarded upon a successful transitions into the target behavioral cluster.
B_{stay}	Smaller, continuous bonus to reinforce the agent's behavior once it has reached and remains within the target region.
P_{wrong}	Significant penalty applied if the agent enters an unauthorized cluster, penalizing deviations from the high-level plan.
P_{vel}	Penalty proportional to the square of the agent's velocity, which discourages aimless wandering once the agent is within the target cluster.

Concurrently, a high-priority cost function provides a continuous signal for collision avoidance. This cost increases as the agent approaches other agents or obstacles and becomes positive when unsafe. During training, the DGPPO framework explicitly minimizes this cost, ensuring the learned policy adheres to a safety-critical constraint. While the cost function is treated as a primary constraint for safety, the reward function serves as a secondary objective, guiding the agent toward efficient and correct task completion. The reader is referred to [3] for full details on training DGPPO. This modified policy trains in $\approx$ 10 hours per landmark.

IV. Experimental Results

To validate the robustness of our approach, we conduct a series of experiments in this section that are designed to test its core capabilities - specifically resilience to various map-environment discrepancies and its ability to generalize.

A. Environments and Data Acquisition

We initially validated our method in a simplified 2D simulation environment shown in Figure 4a, which was built to model bridges, buildings, intersections, and obstacles. This environment was designed to ensure diversity in the size, shape, and orientation of the semantic regions. This simulation provides 2D LiDAR scan data, represented as a 1D array of 36 elements (corresponding to 36 scans at 10-degree increments), with each element indicating the distance to the nearest object within the sensor's range.

The navigation policy was trained exclusively in the 2D simulation. Since the policy learned to navigate between abstract concepts rather than metric or environment-specific features, it demonstrated remarkable transferability to qualitatively different agents and environments. We successfully deployed the learned policy in photo-realistic environments using CARLA, an open-source simulator for autonomous driving research [7], without retraining. This direct transfer is a key validation of our approach.

All three environments—2D simulation, CARLA, and the real-world—are depicted with a robot navigating a bridge in Figure 4. To complete the task of "cross the bridge" the high level plan would be: "Approach Bridge → On Bridge; On Bridge → Exit Bridge; Exit Bridge → Open Space".

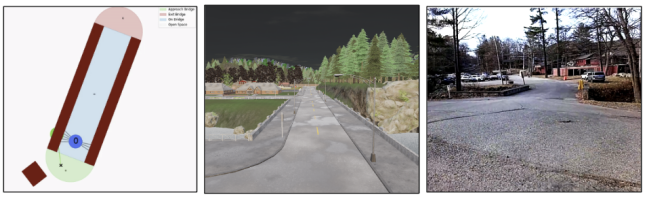

Fig. 4: Our experiments used three environments: simplified 2D simulation (Left), photorealistic CARLA simulator (Center), and real-world (Right)

B. Landmark Cluster Stability and Robustness

In this work, we collect data from a single environment containing all relevant semantic regions, in this case bridges, buildings, and intersections. We isolated 11 behavioral clusters to compose into plans: Bridge {approach, on, exit}, Building {along, around, past}, Intersection {enter, in, exit}, Open Space, Road. We evaluated the performance of our clustering by applying it to a test set with unseen parts of the training scene, new CARLA maps, and novel environments from external datasets like SemanticKITTI [8][9]. The confusion matrix generated with this test set (Figure 5) confirms the strong performance and high level of generalization. The model maintains high recall for the most distinct and easily separable categories, such as "around corner," "along wall," "on bridge," and "open space."

For the remaining behaviors, the most common misclassifications often occur with categories that are adjacent in a sequence. For example, "In Intersection" is most frequently mislabeled as "Enter Intersection" or "Exit Intersection." Since our system relies on navigating between adjacent behaviors, a delayed classification during these transitionary periods is not a significant issue. Misclassifications outside of adjacent regions are the most concerning, and our model does a good job of avoiding these.

In addition to the defined clusters, the matrix includes an "unlabeled" column representing scans that do not fall under any of the existing categories. The majority of these "unlabeled" points are related to intersections. Additionally, the false negatives for these intersections are classified as the "Road" category. This occurs because real-world intersections are far more diverse than the examples encountered

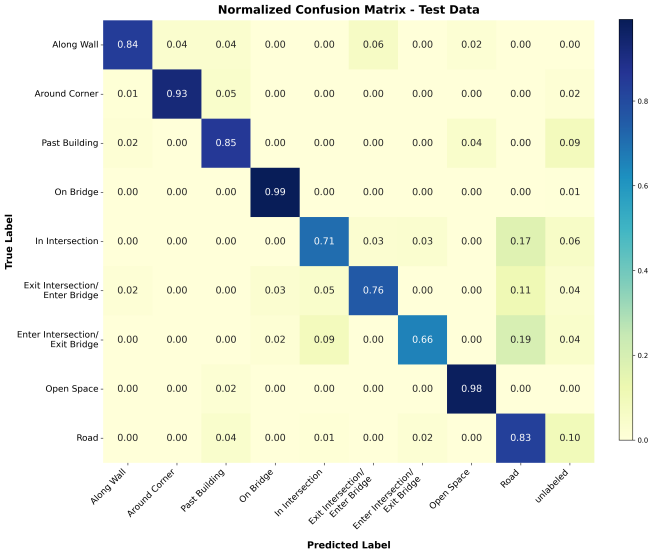

Fig. 5: Normalized confusion matrix for category-wise clustering performance in the test data.

in simulation, causing their embeddings to be further away from their designated cluster centroid. Consequently, these points are misclassified as "Road" rather than being correctly identified or labeled as noise.

C. Navigation Performance with Distorted Prior Maps

To quantitatively evaluate our system's resilience against systematic map inaccuracies, we conducted a series of controlled experiments in the SUBURBAN CARLA map. We progressively introduced four distinct types of map distortion to random sections of the map: rotational offset ($15° - 45°$ in $15°$-degree increments), scaling changes (0.5x and 2x multipliers on semantic region), translational offset (lateral distance from the original centroid, tested at 1, 5, 7 and 10 meters), and perspective (overhead map pitch) distortion ($15° - 45°$ degrees in $15°$-degree increments) which introduced various combinations of the previous three distortions. Figure 6 below illustrates an example of these distortions. The figure on the left, with the red outline, shows the original bridge representation, which corresponds to the ground-truth environment in CARLA. The orange outline in the figure on the right shows the new, distorted bridge representation, with the dotted red outline for comparison to the original.

We measured mission success, defined as the collision-free completion of the entire task, under various distortions for both short and long plans. For each condition, we conducted a total of 12 trials of short plans (4 trials each for building, bridge, and intersection tasks) and 10 trials of long plans (random combinations of 2-3 landmarks' semantic regions).

Our method consistently outperformed a traditional navigation baseline that uses a map-derived cost function to generate a reference trajectory. This baseline implemented a pure pursuit controller to follow a sequence of metric waypoints. As shown in Table I, the baseline's performance degraded significantly after a map rotation of just 15 degrees or a translation offset of 5 meters. This highlights the

TABLE I: Mission success rate for 12 (short) and 10 (long) random trials per test across various map distortion types for our method and a waypoint navigation baseline in the SUBURBAN map.

Distortion Type	Our Method (Short Plan)	Baseline (Short Plan)	Our Method (Long Plan)	Baseline (Long Plan)
Rotation (15°)	1.00	0.75	1.00	0.50
Rotation (30°)	**0.83**	0.25	**0.80**	0.10
Rotation (45°)	**0.58**	0.08	**0.40**	0.00
Scale (0.5x)	**1.00**	0.00	**1.00**	0.00
Scale (2x)	**1.00**	0.42	**1.00**	0.20
Translation (1m)	1.00	1.00	1.00	1.00
Translation (5m)	0.92	0.67	0.80	0.60
Translation (7m)	**0.75**	0.25	**0.70**	0.10
Translation (10m)	0.33	0.00	0.10	0.00
Perspective Skew (15°)	1.00	0.92	1.00	0.90
Perspective Skew (30°)	**0.92**	0.50	**0.70**	0.40
Perspective Skew (45°)	0.75	0.42	0.60	0.10

Fig. 6: Ground-truth overhead map (left) shows the original bridge (red box). Right image depicts the same map with perspective skew where the orange box shows the new, distorted bridge compared to the original (dashed red box).

superior robustness of our semantic-centric approach, which was still successful with up to 30 degrees of rotation and 7 meters of translation. This demonstrates that our system can successfully operate in environments with large discrepancies between the map and reality.

The system's adaptability varied by the type of distortion, with its performance against scale and perspective distortions being particularly robust. Our method demonstrated a strong capability of handling scale distortion, maintaining a perfect 1.00 success rate for both 0.5x and 2x scale changes. This is due to the fact that scaling does not impact our approach because bearings remain constant and the stop condition is defined by a semantic cluster rather than a metric location, which is why the waypoint baseline performed poorly.

The results show that a pure rotation is more difficult for both methods than a perspective skew of the same magnitude. For instance, our method's success rate at a 45° rotational offset (0.58) is lower than its performance under a 45° perspective skew (0.75). This suggests that the uniform, widespread error of a pure rotation is more difficult to manage than the non-uniform, localized errors produced by a pitched overhead map. Ultimately, our model is better equipped to handle less severe distortions, even when compounded, than a single, high-magnitude distortion.

Finally, large translations posed the greatest challenge. The most significant errors came from a mismatch between the agent's actual starting location and the semantic cluster designated by the high-level plan. As the offset increases, the robot could be placed in a completely different behavioral cluster from what the overhead map predicts (e.g., the robot is on a road while the map indicates it is on a bridge). This discrepancy between the on-ground and overhead perspectives violates a core assumption of the approach—that the agent begins in a correct starting behavioral region—and leads to some level of exploration informed by the policy, but successful high-level plan execution remains unlikely.

Longer plans proved more challenging across all types of distortion. This is because navigation errors accumulate over time. This challenge is even greater with more significant distortion, as the initial error is already substantial.

D. Generalization to New Environments

We conducted tests on two new CARLA maps to demonstrate our method's ability to generalize. We specifically chose an environment (URBAN), which has structural similarities to the training map, and a more challenging environment (RURAL), which features greater ground deformation.

TABLE II: Generalization performance (mission success rate) of our semantic-centric method in new CARLA maps. 10 trials per test with 30° of perspective skew applied.

Plan	SUBURB	URBAN	RURAL
Short	0.90	0.80	0.70
Long	0.70	0.70	0.60

Our method maintains a consistently high mission success rate across both new environments as shown in Table II. The minor drop in performance in the RURAL map, compared to the more structurally similar URBAN map, is expected. This can be attributed to the subtle differences in ground structure and environmental features of the rural environment, which occasionally make the clustering more challenging.

The strong performance on unseen maps provides compelling evidence that our system learns abstract behavioral and semantic concepts rather than memorizing environment-specific features. The policy's success in new environments confirms that the learned behavioral clusters are robust and transfer effectively to new, structurally different maps.

E. Static and Dynamic Obstacles in the Environment

We investigated the effects of visual clutter on our system's performance by conducting experiments across three distinct environments created from the SUBURBAN map: the original map, a version with added static clutter, and a version with both static and dynamic obstacles (see Figure 7 below).

Fig. 7: Addition of static (left) and dynamic (right) obstacles (red boxes) to the SUBURBAN environment.

TABLE III: Mission Success Rate with Varying Clutter. 10 trials per test with 30° of perspective skew applied to map.

Plan Length	Original	Static Clutter	Dynamic Clutter
Short	0.90	0.80	0.60
Long	0.70	0.70	0.40

As Table III shows, the system performed robustly with static visual distractors. Our clustering algorithm, which identifies key structural components, remained effective as these obstacles did not interfere with the learned embeddings. The DGPPO controller successfully navigated these novel elements, having been trained on similar static obstacles in simulation. However, the inclusion of dynamic obstacles led to a significant performance drop. The clustering remained relatively robust so for future work, retraining the controller with exposure to moving agents during training should resolve issues with erratic behavior and high collision rates.

F. Real-Robot Experiments

To validate our approach in real-world settings, we deployed our live clustering algorithm on a robot test configuration with a Livox MID360 LiDAR. The current bearing comes from DLIO [10] odometry. The behavioral clusters were then compared against the DLIO map to confirm that it correctly identified the robot's behavior in each region.

The results of this validation corroborate our findings from simulation - each of the 30 cluster segments, as shown in Figure 8, were correctly identified by the system. The robot's trajectory, when analyzed with our live clustering, effectively shows transitions at key semantic regions and remains consistent within a given behavioral segment. This ability to detect and respond to real-world environmental features on-the-fly validates that our live clustering and high-level planning components are robust in real-world scenarios. The real-time performance of our approach demonstrates its feasibility for online behavioral navigation. The full execution of a cluster plan will be shown in the video submission.

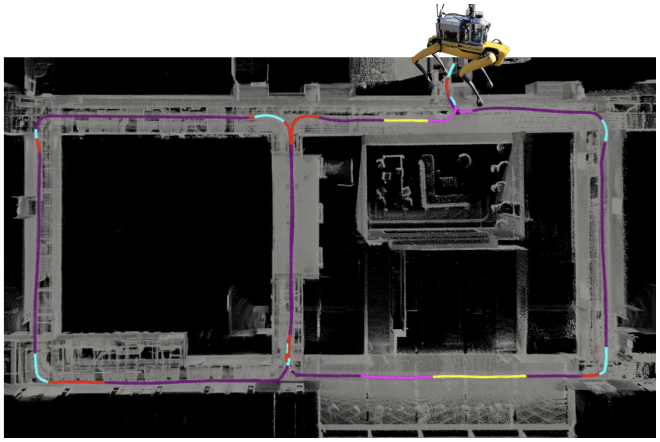

Fig. 8: The robot's trajectory is overlaid on a reconstruction of the environment. Colors indicate the behavioral cluster.

V. DISCUSSION

The experiments above validate our system's robustness to map discrepancies and its ability to generalize to novel environments. Our clustering algorithm successfully learned a meaningful, abstract representation of the environment, which is the foundation of our method's resilience. By consistently outperforming a traditional waypoint-based baseline, our system proved its superior ability to navigate with significant map distortions, including rotational, scaling, and translational errors. The system's ability to handle static visual clutter further demonstrates this robustness; this scenario is analogous to operating with a stale map, where the robot's sensors detect new, fixed geometry not present in its prior map. Our experiments showed that the system had minimal performance degradation with these visual distractors, as it successfully adapted to these new static obstacles.

VI. RELATED WORKS

Traditional navigation systems often rely on *a priori* cost maps which fuse global map information with real-time sensor data [11]. However, they are susceptible to errors in the initial global map and struggle to adapt to changed environments. In contrast, our work uses a behavioral-centric framework that associates high-level concepts with a robot's current perception, moving away from relying on static maps.

Alternative strategies, like landmark-based navigation, use place recognition to identify key locations and bypass the need for detailed maps. However, these methods often reduce landmarks to point-based representations, which may not capture the complexity of how a robot interacts with semantically meaningful regions [12]. In contrast, our representation provides a richer understanding of the environment.

Another body of work, specifically end-to-end policy learning, seeks to directly learn navigation policies from sensor data using deep reinforcement learning (RL) [13] and particularly goal-conditioned RL. Navbot_ppo is an example of this approach, using Proximal Policy Optimization (PPO) to learn a mapless navigation policy directly from LiDAR and target coordinates [14]. However, these techniques still require a metric position estimate and goal. In contrast, our approach can be seen as a form of goal-conditioned RL with a semantically specified goal.

Our framework offers a crucial balance of interpretability and computational efficiency. Unlike LLM-based approaches which are opaque and computationally expensive for real-time robot control [15], our framework is explicitly transparent since the robot's high-level plan is a human-interpretable sequence of behavioral clusters. Additionally, while a hand-crafted set of rules could be designed for a specific environment, such a system would be brittle and would not generalize to new or distorted surroundings [16]. Our learned embeddings and HDBSCAN clustering process help identify the most salient behavioral regions, allowing the system to robustly generalize without manual feature engineering.

While our use of a simplified overhead map for high-level guidance is conceptually similar to work on navigating from hand-drawn "napkin maps" [17], these implementations often rely on Vision-Language Models (VLMs). VLMs use large-scale, pre-trained models to interpret maps and sensor input, solving navigation at a higher semantic level [18]. For example, LM-Nav [19] performs visual navigation by reasoning about high-level, human-like instructions (e.g., "drive to the building"). In contrast, our system generates and uses simple, composable behavioral categories at a geometric level. Our approach is complementary to VLM-based methods and can learn the association between abstract map regions and real-world behaviors without relying on massive training datasets and visual cues inherent in VLM-based approaches.

VII. Conclusion

This work presents a novel navigation system that enables robots to operate with imprecise maps by leveraging a semantic-centric framework. Instead of relying on a fragile geometric representation, our approach abstracts the environment into a set of human-interpretable behavioral concepts, which we obtain by clustering learned embeddings from sensor data. A key contribution is our learned controller, which navigates between these abstract concepts rather than precise coordinates, allowing for real-time adaptation to an environment that deviates from its map. Experimental results demonstrate improved performance over traditional waypoint-based baselines in scenarios with intentionally distorted maps, and confirm the successful transfer of learned behaviors from simulation to a real-world robot.

A natural extension of this work would be handling path variations, like prioritizing "around" over "on" a bridge when the path is obstructed, ensuring the robot can navigate alternative routes based on real-time conditions. Future work could also incorporate multi-modal inputs, leveraging the synergy of LiDAR and vision. By processing image data at key transitions - like at the start of a new semantic region - the system can enrich its semantic understanding without the computational overhead. A key advantage of our framework is that this abstraction renders the downstream navigation policy inherently sensor-agnostic. Any combination of sensors capable of identifying the behavioral vocabulary can be integrated without retraining the policy.

Acknowledgements

We gratefully acknowledge the support of the NSF GFRP program, ARL under the DCIST CTA and ONR Uncertainty-based Active Learning N00014-22-1-2677.

References

[1] J. Liang, Z. Wang, Y. Cao, J. Chiun, M. Zhang, and G. A. Sartoretti, "Context-aware deep rl for autonomous robotic navigation in unknown area," in *7th Annual Conference on Robot Learning*, 2023.

[2] T. Ort, L. Paull, and D. Rus, "Autonomous vehicle navigation in rural environments without detailed prior maps," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[3] S. Zhang, O. So, M. Black, and C. Fan, "Discrete GCBF proximal policy optimization for multi-agent safe optimal control," in *The 13th International Conference on Learning Representations*, 2025.

[4] B. Eysenbach, T. Zhang, S. Levine, and R. R. Salakhutdinov, "Contrastive learning as goal-conditioned reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 35, 2022.

[5] I. A. Nellas, S. K. Tasoulis, and V. P. Plagianakos, "Convolutional variational autoencoders for image clustering," in *2021 International Conference on Data Mining Workshops (ICDMW)*, 2021, pp. 695–702.

[6] R. J. G. B. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Advances in Knowledge Discovery and Data Mining*, J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, Eds., 2013.

[7] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[10] K. Chen, R. Nemiroff, and B. T. Lopez, "Direct lidar-inertial odometry: Lightweight lio with continuous-time motion correction," 2023. [Online]. Available: https://arxiv.org/abs/2203.03749

[11] J. Jiao, R. Geng, Y. Li, R. Xin, B. Yang, J. Wu, L. Wang, M. Liu, R. Fan, and D. Kanoulas, "Real-time metric-semantic mapping for autonomous navigation in outdoor environments," *IEEE Transactions on Automation Science and Engineering*, 2024.

[12] J. Crespo, J. C. Castillo, O. M. Mozos, and R. Barber, "Semantic information for robot navigation," *Applied Sciences*, no. 2, 2020.

[13] A. Ghafourian, Z. CuiZhu, D. Shi, I. Chuang, F. Charette, R. Sachdeva, and I. Soltani, "Hierarchical end-to-end autonomous navigation through few-shot waypoint detection," *IEEE Robotics and Automation Letters*, vol. 9, no. 4, pp. 3211–3218, 2024.

[14] H. Taheri, S. R. Hosseini, and M. A. Nekoui, "Deep reinforcement learning with enhanced ppo for safe mobile robot navigation," 2024.

[15] B. Liang, Y. Wang, and C. Tong, "Ai reasoning in deep learning era: From symbolic ai to neural–symbolic ai," *Mathematics*, vol. 13, 2025.

[16] A. A. Golroudbari and M. H. Sabour, "Recent advancements in deep learning applications and methods for autonomous navigation: A comprehensive review," 2023.

[17] A. H. Tan, A. Fung, H. Wang, and G. Nejat, "Mobile robot navigation using hand-drawn maps: A vision language model approach," 2025.

[18] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023.

[19] D. Shah, B. Osinski, B. Ichter, and S. Levine, "Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action," 2022. [Online]. Available: https://arxiv.org/abs/2207.04429