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

# NORM-Nav: Zero-Shot Mobile Robot Navigation with Natural Language Behavioral Constraints

Dongjie Huo[1*], Junhui Wang[2,3*‡], Chao Gao[2†], Yan Qiao[3], Dong Zhang[1], Guyue Zhou[2,4†]

***Abstract*— Mobile robots operating in human-centered environments must generate not only collision-free paths but also trajectories that follow local behavioral conventions. Conventional costmap-based navigation emphasizes geometric feasibility and often overlooks such requirements, which can result in socially inappropriate behaviors. This paper presents NORM-Nav, a zero-shot framework that integrates natural language behavioral constraints into costmap-based planning. An LLM parses each instruction into structured constraints and grounds them using real-time vision–LiDAR perception. These constraints are encoded as multi-layer costmaps that represent geometric, semantic, directional, and velocity cues and are directly compatible with standard grid-based planners. Simulation and real-world experiments indicate that NORM-Nav improves task success rates and produces trajectories closer to human references than representative baselines. The project website is available at `https://ei-nav.github.io/NORM-Nav`.**

## I. Introduction

Mobile robots are increasingly deployed in complex and dynamic environments shared with humans. In such settings, effective navigation requires more than collision-free path planning. Robots must also conform to behavioral rules that are implicitly expected by people [1], [2], such as keeping to one side of a corridor or adjusting speed when approaching vulnerable road users. While trajectories that disregard these conventions may remain geometrically feasible, they often appear unnatural, socially inappropriate, or unsafe, thereby limiting the acceptance of robots in human environments. Representative examples of such behavioral expectations are illustrated in Fig. 1.

Costmap-based navigation frameworks continue to serve as the backbone of most deployed systems due to their computational efficiency and robustness [3], [4]. These approaches typically represent the environment as a grid in which all free space is treated as traversable. As a result, they fail to distinguish between semantically different regions, for example, sidewalks and grassy areas, and cannot adapt to context-specific behavioral requirements [5]. Furthermore, reliance on purely geometric perception can result in misinterpretation. Tall grass may be treated as an obstacle, whereas lightweight curtains at a doorway may be mistakenly perceived as rigid barriers. Such limitations reduce efficiency and prevent robots from exhibiting socially compliant behaviors.

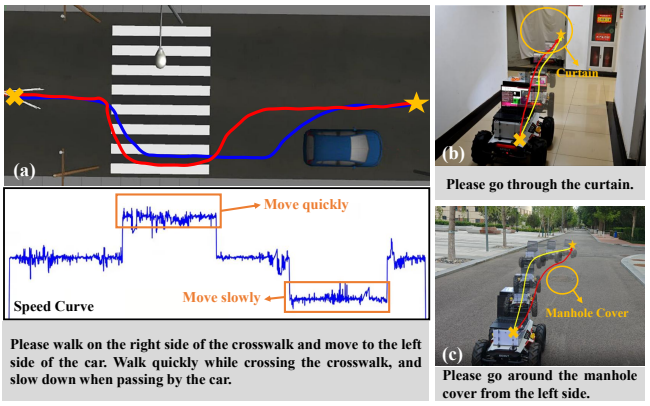


Fig. 1. Examples of robot navigation under natural language behavioral constraints. The blue line indicates the executed trajectory, while the red line shows a human-preferred reference path. The robot is able to (a) adjust its speed while complying with behavioral constraints, (b) traverse perceptually detected but traversable obstacles such as curtains, and (c) comply with side-specific instructions, such as bypassing a manhole cover from the left even though it is physically traversable.

Learning-based approaches have been investigated to alleviate these challenges [6]–[8]. These methods directly map natural language and sensory input to low-level actions, providing flexibility in principle. However, they often depend on large-scale datasets that are costly to collect and seldom capture the diversity of real-world environments. Consequently, they tend to generalize poorly to unseen scenarios and may exhibit brittle or opaque behaviors. Reinforcement and imitation learning strategies offer richer modeling of behaviors [9], [10], but require substantial training effort and remain sensitive to distribution shifts, which constrains their practicality.

Large language models (LLMs) can parse natural language instructions and support rule-related reasoning without task-specific training [11], [12]. Combined with camera–LiDAR perception, they can translate free-form instructions into structured navigation constraints [13]. This enables integrating user intent into planning while retaining the modularity of classical navigation frameworks.

This work proposes *NORM-Nav*, a zero-shot navigation framework that integrates natural language behavioral constraints into costmap-based planning. NORM-Nav converts each instruction into structured constraints (spatial prefer-

* Equal contribution.
‡ Project lead.
† Corresponding authors.

[1]College of Information Science and Technology, Beijing University of Chemical Technology, [2]Institute for AI Industry Research (AIR), Tsinghua University, [3]Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, [4]School of Vehicle and Mobility, Tsinghua University.

Sponsored by Xinchen Qihang Inc. and XIAOMI Fund, and supported by the National Natural Science Foundation of China (No. 52475005), and in part by the Fundamental Research Funds for the Central Universities under Grant PY2505.

ence, velocity adjustment, and traversability) and grounds them through real-time vision–LiDAR fusion. The constraints are encoded as multi-layer costmaps that are directly consumable by standard grid-based planners.

We evaluate NORM-Nav in simulation and real-world experiments. Relative to representative baselines, NORM-Nav achieves higher success rates, follows user-specified behavioral constraints more consistently, and produces trajectories that are closer to human references.

The main contributions are summarized as follows:

- We propose NORM-Nav, a zero-shot navigation framework that translates natural language behavioral constraints into structured planning cues and integrates them into costmap-based navigation.
- We present a modular, plug-in design that augments standard costmap-based stacks without changes to the underlying planner.
- We demonstrate the effectiveness of NORM-Nav through simulation and real-world experiments.

## II. Related Work

### A. Semantic Extensions to Classical Navigation

Costmap-based frameworks remain the cornerstone of classical robot navigation, where the environment is discretized into free and occupied cells [14]. While computationally efficient and widely deployed, such representations cannot account for semantic or socially motivated preferences. To address this limitation, researchers have proposed augmenting costmaps with semantic information [15], [16]. By embedding contextual cues, robots are able to differentiate between regions that are geometrically alike but behaviorally distinct. For example, Zhang et al. [17] combined an LLM with a vision–language model (VLM) to reinterpret LiDAR returns, allowing objects such as curtains to be considered traversable. While these techniques demonstrate the benefit of linking semantics with navigation, they are difficult to generalize to broader behavioral norms. Instead of embedding fixed semantic priors into costmaps, we enable flexible interpretation of semantic instructions through a structured integration with classical navigation, thereby retaining robustness while extending behavioral variety.

### B. Language-Driven Navigation

End-to-end vision–language navigation (VLN) methods map user instructions and sensory observations into motion policies through a unified model [18], [19]. These approaches achieve competitive performance in benchmark evaluations [6]–[8], but real-world deployment remains challenging. The main difficulties include dependence on large-scale annotated datasets [20], limited generalization to unseen environments [21], and fragile behavior under perceptual uncertainties [22].

Zero-shot methods attempt to address these limitations by leveraging pretrained LLMs and VLMs to interpret instructions without task-specific supervision [23]–[25]. Natural language is converted into structured representations and aligned with open-vocabulary visual recognition [26]–[28], thereby supporting flexible generalization. Nevertheless, their low decision frequency often hinders reliable deployment in realistic settings. Hybrid methods have therefore been proposed to combine semantic reasoning with traditional planning. BehAV [13] integrates language-specified constraints into classical planning, enhancing interpretability. Our method follows a similar philosophy but adopts a more generalized formulation. Instead of tightly coupling language constraints to planning rules, we introduce a lightweight semantic reasoning module that maps free-form instructions onto structured navigation primitives. This design preserves the efficiency and reliability of costmap-based navigation, while offering broader adaptability to diverse behavioral instructions.

## III. Problem Formulation

This work considers the problem of robot path planning under natural language behavioral constraints with real-time perception. The robot is placed in an environment with a start position $S$ and a goal position $T$, and is provided with a set of behavioral constraints $\mathcal{C}$ stated in natural language. Examples include instructions such as "avoid walking on the grass," "keep to the right side of the road," or "pass vehicles on the left." The objective is to generate a feasible trajectory $\tau$ from $S$ to $T$ that satisfies all constraints in $\mathcal{C}$.

During navigation, the robot continuously acquires multimodal sensory observations $O_t$ from its onboard LiDAR and camera at time step $t$. These data are essential for detecting obstacles, identifying semantic elements, and associating the behavioral constraints with the current environment. As the robot progresses, the planned trajectory must be adaptively updated based on the latest perception to ensure that all specified constraints are respected.

Formally, let $\mathcal{P}$ denote the set of all feasible trajectories from $S$ to $T$ that are consistent with the robot's motion capabilities and the observed environment $\{O_t\}$. The problem is to find an optimal trajectory

$$\tau^* = \arg\min_{\tau \in \mathcal{P}} J(\tau) \quad \text{subject to} \quad \tau \models \mathcal{C}, \tag{1}$$

where $J(\tau)$ is a navigation cost function, and $\tau \models \mathcal{C}$ indicates that $\tau$ fulfills all constraints in $\mathcal{C}$ after they are interpreted and grounded through real-time sensory perception.

In summary, the inputs to the problem consist of the navigation pair $(S, T)$, the sensory stream $\{O_t\}$, and the natural language constraints $\mathcal{C}$, while the output is an executable trajectory $\tau^*$ that minimizes $J(\tau)$ and satisfies $\mathcal{C}$. The key challenge lies in transforming inherently ambiguous natural language instructions into reliable, perception-grounded constraints and in ensuring that the derived trajectory remains both feasible and compliant as the environment evolves.

## IV. Methodology

This work proposes a zero-shot navigation framework that enables robots to interpret and execute behavioral constraints expressed in natural language without relying on additional task-specific training. The core idea is to integrate LLMs with conventional costmap-based navigation so that planned

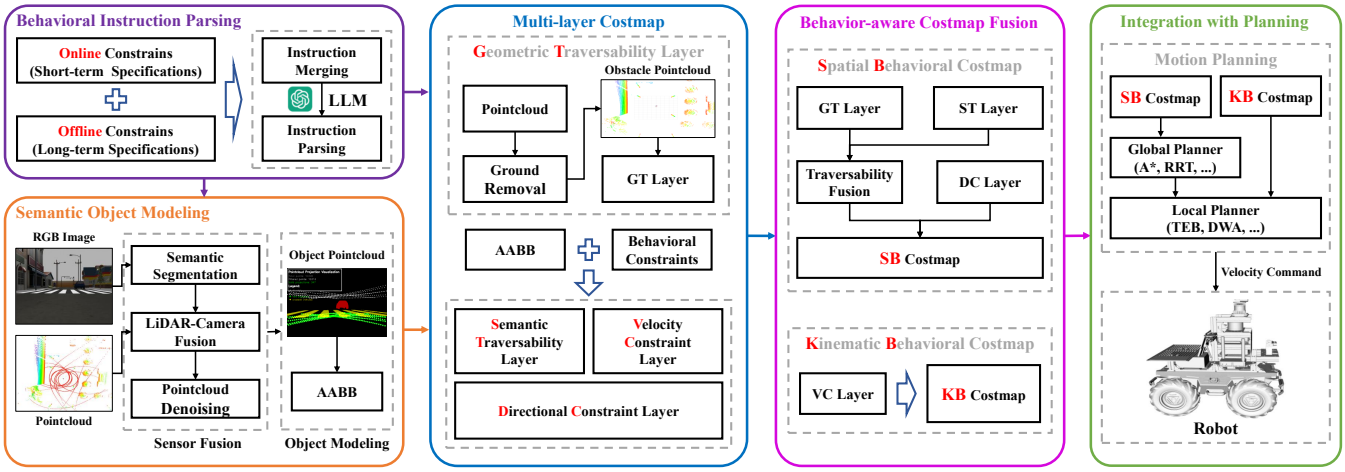


Fig. 2. The architecture of the proposed method for zero-shot navigation under natural language behavioral constraints. The system integrates LLMs with vision–LiDAR perception and encodes parsed behavioral instructions into multi-layer costmaps for motion planning.

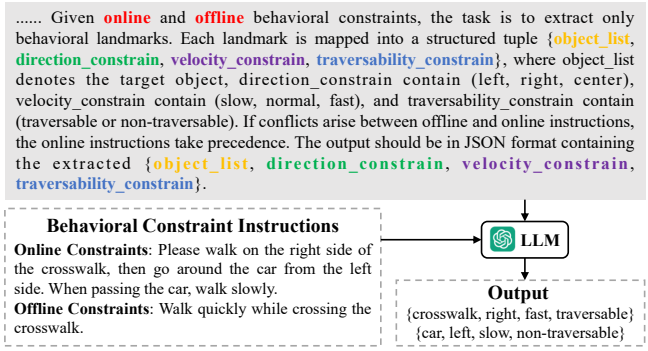


Fig. 3. Example of parsing online and offline behavioral constraints into structured representations.

trajectories remain both geometrically feasible and behaviorally appropriate. Instead of relying solely on geometric representations, the framework parses long-term rules and short-term situational instructions into structured constraints, grounds them through real-time fusion of visual and LiDAR perception, and encodes them within the cost space. These constraints are organized into multi-layer costmaps that incorporate geometric, semantic, spatial, and kinematic information, which are subsequently fused to guide planning. The overall system architecture is illustrated in Fig. 2.

### A. Behavioral Instruction Parsing

Natural language provides a direct and flexible interface for humans to specify navigation behaviors. To accommodate instructions with different temporal scopes, behavioral constraints are categorized into two groups.

*1) Offline constraints:* The set of offline behavioral constraints is defined as

$$\mathcal{L}_{off} = \{L_1^{off}, L_2^{off}, \ldots, L_m^{off}\}, \tag{2}$$

where each $L_i^{off}$ encodes global requirements that remain valid for the entire task, such as "*keep right when walking on the road*" or "*avoid entering the grass*."

*2) Online constraints:* The set of online behavioral constraints is defined as

$$\mathcal{L}_{on} = \{L_1^{on}, L_2^{on}, \ldots, L_n^{on}\}, \tag{3}$$

where each $L_j^{on}$ specifies short-term requirements triggered by particular situations, such as "*slow down near the car*" or "*pass through the curtain*."

*3) Unified constraints:* For subsequent processing, the two sets are merged into a unified specification:

$$\mathcal{L} = \mathcal{L}_{off} \cup \mathcal{L}_{on}. \tag{4}$$

This specification is then parsed by an LLM into a structured representation, as illustrated in Fig. 3.

$$\mathcal{C} = \{(O_i, d_i, v_i, t_i) \mid i = 1, \ldots, r\}, \tag{5}$$

where $O_i$ denotes the referenced object, $d_i$ encodes spatial preference, $v_i$ indicates velocity requirements, and $t_i$ represents semantic traversability. Through this process, both long-term norms and short-term situational instructions are cast into a unified symbolic form that can guide perception and planning.

Table I summarizes the definition and representative values of each element. The LLM interprets free-form natural language instructions and maps them into these representative values, thereby producing consistent structured tuples.

### B. Semantic Object Modeling

To interpret behavioral constraints within the observed environment, semantic object models are constructed via camera–LiDAR fusion in two stages: open-vocabulary semantic grounding and multi-frame aggregation.

*1) Open-vocabulary semantic grounding:* Given an RGB image $I_t \in \mathbb{R}^{H \times W \times 3}$ at time step $t$ and the textual description of an object $O_i$, GSAM2 [29] generates a binary mask $M_i$. The pixel set corresponding to $O_i$ is

$$S_i = \{(u, v) \mid M_i(u, v) = 1\} \tag{6}$$

The LiDAR point cloud at time $t$ is denoted as $P_t = \{p_j = (x_j, y_j, z_j)\}_{j=1}^{N}$. Each point $p_j$ is projected to the image

TABLE I

DEFINITION AND REPRESENTATIVE VALUES FOR EACH ELEMENT IN STRUCTURED BEHAVIORAL CONSTRAINTS

| Symbol | Meaning | Representative Values | Example Instruction |
|---|---|---|---|
| $O_i$ | Referenced object | pedestrian, vehicle, grass, crosswalk, ... | "Walk on the crosswalk", "Avoid the grass" |
| $d_i$ | Directional preference | {∅, left, right, middle} | "Pass the car on the left", "Keep to the right side of the road" |
| $v_i$ | Velocity requirement | {∅, slow, normal, fast} | "Walk slowly near the car", "Move quickly when crossing the street" |
| $t_i$ | Semantic traversability | {∅, traversable, non-traversable} | "Go through the curtain", "Lawn is non-traversable" |

TABLE II

ASSIGNMENT OF $(c_1, c_2, c_3)$ UNDER DIFFERENT DIRECTIONAL CONSTRAINTS

| Constraint | $c_1$ | $c_2$ | $c_3$ |
|---|---|---|---|
| left | $c_{\min}$ | $\frac{1}{2}(c_{\min} + c_{\max})$ | $c_{\max}$ |
| right | $c_{\max}$ | $\frac{1}{2}(c_{\min} + c_{\max})$ | $c_{\min}$ |
| middle | $c_{\max}$ | $c_{\min}$ | $c_{\max}$ |

plane via the projection matrix and is associated with $O_i$ if its projection lies in $S_i$, yielding the object-specific subset $P_i^t \subseteq P_t$.

*2) Multi-frame aggregation:* To enhance robustness, object models are aggregated over a sliding temporal window. At each time step $t$, $P_i^t$ is refined using DBSCAN [30] to suppress noise, resulting in $\hat{P}_i^t$. The temporally aggregated model is

$$P_i(t) = \bigcup_{\tau=t-T+1}^{t} \hat{P}_i^{\tau} \tag{7}$$

where $T$ is the aggregation horizon. On the bird's-eye view (BEV) plane, object $O_i$ is approximated by an axis-aligned bounding box (AABB) given by

$$B_i = \left( \min_{p \in P_i}(x, y),\ \max_{p \in P_i}(x, y) \right) \tag{8}$$

This BEV-projected bounding box provides a compact representation for encoding behavioral constraints in costmap construction.

### C. Multi-layer Costmap with Semantic Guidance

Trajectories consistent with natural language constraints are generated by constructing a multi-layer costmap. Four complementary layers are considered: geometric traversability, semantic traversability, directional constraints, and velocity constraints.

*1) Geometric Traversability Layer:* This layer encodes free space and obstacle occupancy. From $P_t$, ground points and points above the maximum robot height $h_{\max}$ are removed. The remaining obstacles form

$$P_t^{obs} = \{p_j \mid z_{\min} < z_j < h_{\max},\ p_j \notin P_t^{ground}\} \tag{9}$$

where $P_t^{ground}$ denotes estimated ground. Each obstacle is projected onto a BEV grid to yield

$$C_{geo}(u, v) = \begin{cases} 255, & \exists p_j \in P_t^{obs} \text{ s.t. } \pi_{\text{BEV}}(p_j) = (u, v) \\ 100, & \text{otherwise} \end{cases} \tag{10}$$

with $\pi_{\text{BEV}}(\cdot)$ the projection function.

*2) Semantic Traversability Layer:* Semantic information from $\mathcal{C}$ is incorporated. For object $B_i$ with semantic traversability $t_i$, the average geometric cost is

$$\bar{C}_{geo}(B_i) = \frac{1}{|B_i|} \sum_{(u,v) \in B_i} C_{geo}(u, v) \tag{11}$$

If $t_i = \emptyset$, it is inferred as

$$t_i = \begin{cases} \text{non-traversable}, & \bar{C}_{geo}(B_i) > \tau \\ \text{traversable}, & \text{otherwise} \end{cases} \tag{12}$$

The semantic traversability cost is then defined as

$$C_{sem}(u, v) = \begin{cases} 1, & (u, v) \in B_i,\ t_i = \text{traversable} \\ 2, & (u, v) \in B_i,\ t_i = \text{non-traversable} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

This step allows dynamic regulation of traversable regions based on semantic priors.

*3) Directional Constraint Layer:* Directional constraints are employed to encode side-specific navigation preferences with respect to semantic objects. When an instruction specifies $d_i \neq \emptyset$, the bounding box $B_i$ of the corresponding object is first enlarged by a margin $d$ if $t_i =$ non-traversable, so that the constraint also applies around the obstacle boundary for safe planning.

For cost assignment, three boundary values $(c_1, c_2, c_3)$ are placed along the lateral axis of $B_i$, with their values determined by the directional rule (see Table II). As illustrated in Fig. 4, the bounding box is divided into two consecutive regions. The cost distribution is obtained through piecewise interpolation:

- from $c_1$ to $c_2$ across the first region,
- from $c_2$ to $c_3$ across the second region.

Both regions are interpolated using the same function. Let $u$ denote the horizontal coordinate of a grid cell. The interpolation function is expressed as

$$C(u) = \begin{cases} \left\lfloor c_1 + (c_2 - c_1) \left( \dfrac{u - u_1}{u_2 - u_1} \right)^{\alpha} \right\rfloor, & u \in [u_1, u_2], \\ \left\lfloor c_2 + (c_3 - c_2) \left( \dfrac{u - u_2}{u_3 - u_2} \right)^{\alpha} \right\rfloor, & u \in [u_2, u_3], \end{cases} \tag{14}$$

where $(u_1, u_2, u_3)$ denote the left boundary, midpoint, and right boundary of $B_i$, and the parameter $\alpha > 0$ controls the sharpness of the gradient. When $\alpha = 1$, the interpolation is linear. The operator $\lfloor \cdot \rfloor$ ensures that the interpolated costs are discretized to integer values for consistency with the grid-based costmap. The preferred side specified by

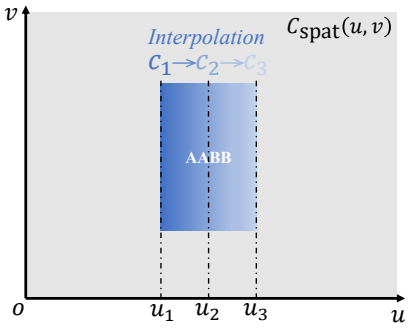


Fig. 4. Illustration of the directional constraint layer.

the instruction corresponds to the low-cost region, whereas the non-preferred side is penalized by higher costs. The resulting cost field allows standard planners to naturally generate trajectories that adhere to side-specific navigation preferences.

*4) Velocity Constraint Layer:* Velocity rules are encoded when $v_i \neq \emptyset$. The resulting cost field is

$$C_{vel}(u,v) = \begin{cases} c_{\min}, & (u,v) \in B_i,\ v_i = \text{slow} \\ \frac{1}{2}(c_{\min} + c_{\max}), & (u,v) \in B_i,\ v_i = \text{normal} \\ c_{\max}, & (u,v) \in B_i,\ v_i = \text{fast} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

### D. Behavior-aware Costmap Fusion

The four layers are combined into two categories of behavioral costmaps: a spatial behavioral costmap $C_{spat}$, derived from geometric, semantic, and directional constraints; and a kinematic behavioral costmap $C_{vel}$, derived from velocity preferences.

*1) Spatial Behavioral Costmap:* Geometric and semantic traversability layers are first fused:

$$\begin{aligned} I(u,v) = {} & \tfrac{1}{2} C_{geo} \odot (\mathbf{1} - C_{sem})(\mathbf{2} - C_{sem}) \\ & + 100 \cdot C_{sem} \odot (\mathbf{2} - C_{sem}) \\ & + \tfrac{1}{2} \cdot 255 \cdot C_{sem} \odot (C_{sem} - \mathbf{1}) \end{aligned} \tag{16}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{1}, \mathbf{2}$ are matrices of all ones and all twos. This step corrects LiDAR perception results using semantic input. The result $I$ is further combined with $C_{dir}$:

$$\begin{aligned} C_{spat}(u,v) = {} & \mathbf{1}(I = 255) \odot 255 \\ & + \mathbf{1}(I \neq 255) \odot \mathbf{1}(C_{dir} > 0) \odot C_{dir} \\ & + \mathbf{1}(I \neq 255) \odot \mathbf{1}(C_{dir} = 0) \odot I \end{aligned} \tag{17}$$

where $\mathbf{1}(\cdot)$ is an indicator function.

*2) Kinematic Behavioral Costmap:* The kinematic behavioral costmap $C_{kin}$ is directly constructed from $C_{vel}$ (Sec. IV-C.4) and encodes localized velocity modulation without further fusion.

Together, the spatial behavioral costmap $C_{\text{spat}}$ and the kinematic costmap $C_{\text{kin}}$ provide complementary guidance. The spatial term enforces geometric and semantic traversability as well as directional preferences, whereas the kinematic term regulates local velocity. The planner then leverages these fused costmaps to generate trajectories that are collision-free and aligned with natural language behavioral specifications.

### E. Integration with Planning

The fused behavioral costmaps are subsequently utilized by the motion planning to generate executable trajectories. The spatial behavioral costmap and the kinematic behavioral costmap play complementary roles. Each contributes a distinct function, and together they ensure that the robot follows paths that are both geometrically feasible and consistent with behavioral specifications provided in natural language.

*1) Spatial Behavioral Costmap for Geometric Optimization:* The spatial behavioral costmap $C_{\text{spat}}$ can be directly applied to planners that operate on a grid-based representation of the environment. In this configuration, $C_{\text{spat}}$ augments the standard geometric costmap so that behavioral requirements, including keeping to a specific side of a road or avoiding restricted regions, are incorporated into the optimization process. Because $C_{\text{spat}}$ maintains compatibility with existing costmap-based frameworks, no modification to the underlying planning algorithms is required. This modular integration preserves generality while embedding behavior awareness into the trajectory generation process.

*2) Kinematic Behavioral Costmap for Velocity Regulation:* The kinematic behavioral costmap $C_{\text{kin}}$ provides a complementary function by regulating the velocity of the robot during execution. Instead of influencing the spatial structure of the trajectory, $C_{\text{kin}}$ specifies speed constraints for the cell currently occupied by the robot. The cost value at the corresponding grid location is mapped into an admissible limit on translational velocity, which is forwarded to the local planner. In this way, the robot is able to adjust its speed dynamically according to user instructions, for example, slowing down near vehicles or increasing velocity when crossing a crosswalk.

In combination, the two costmaps provide a unified mechanism for integrating behavioral constraints with conventional planning algorithms. The spatial behavioral costmap guides the generation of paths that adhere to traversability and directional preferences, while the kinematic behavioral costmap ensures that execution speed respects context-dependent motion requirements. Together, these mechanisms enable conventional planning algorithms to produce trajectories that are both socially acceptable and context aware.

## V. Experimental Results

### A. Experimental Setup

We conduct experiments in simulation and real-world environments to evaluate the effectiveness of the proposed method.

*1) Simulation Environment:* The MEDIUM environment from OpenBench [3] is used to examine navigation performance under various behavioral constraints. The simulated platform is a four-wheeled differential-drive robot equipped with a Livox MID-360 LiDAR and an RGB-D camera.

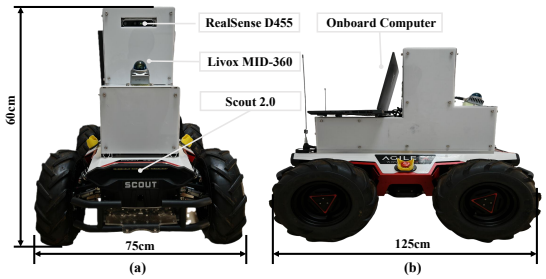


Fig. 5. The experimental robot platform: (a) front view and (b) side view.

*2) Real-world Platform:* For physical validation, we employ a SCOUT 2.0 differential-drive mobile robot (Fig. 5). The platform is equipped with a Livox MID-360 LiDAR for 3D perception, an Intel RealSense D455 depth camera for RGB-D sensing, and an onboard computer (AMD R9-7945HX CPU, NVIDIA RTX 4060 GPU) for real-time processing. The LiDAR is tilted at an angle of 20 degrees to scan the ground.

*3) Evaluation Metrics:* We assess navigation performance using four adopted metrics:

- *Success Rate (SR)*: the proportion of successful trials in which the robot reaches the goal while fully respecting all behavioral constraints.
- *Success weighted by Path Length (SPL) [31]*: a combined measure of efficiency that reflects both success rate and path optimality relative to the shortest feasible trajectory.
- *Fréchet Distance (FD) [32]*: the similarity between the executed trajectory and a human teleoperated reference path, where smaller values indicate closer alignment to human-preferred navigation.
- *Behavioral Following Accuracy (BFA) [13]*: the fraction of the executed trajectory length that is consistent with the specified behavioral constraints.

*4) Implementation Details:* Behavioral instructions are parsed using *Qwen-VL-Max*. In costmap construction, both the spatial and kinematic constraint layers employ $c_{\min} = 0$ and $c_{\max} = 255$, with the interpolation parameter in Eq. 14 fixed at $\alpha = 3.0$. The costmap resolution is set to $0.05\,\mathrm{m}$ over a horizon of $10\,\mathrm{m} \times 10\,\mathrm{m}$. For path generation, the global planner adopts the A* algorithm, while local trajectory optimization is performed using the Timed Elastic Band (TEB) planner.

### B. Evaluation of Navigation Performance

We evaluate the proposed method against representative language-driven navigation approaches, including BehAV [13], InstructNav [28], and the interactive navigation framework (INF) [17]. All task instructions are expressed in natural language and are divided into four categories:

- *Region-Following Task:* Robots are required to approach or traverse designated semantic regions while following specified behavioral constraints.
- *Region-Avoidance Task:* Robots are required to actively avoid certain semantic regions or obstacles while respecting the given behavioral constraints.

TABLE III
EVALUATION OF NAVIGATION PERFORMANCE

| Task | Method | SR (↑) | SPL (↑) | FD (↓) | BFA (↑) |
|---|---|---|---|---|---|
| Region Following Task | BehAV | 60% | 19.63% | 4.21 | 62% |
| | InstructNav | 50% | 15.14% | 5.36 | 58% |
| | INF | – | – | – | – |
| | NORM-Nav | **90%** | **65.77%** | **2.14** | **89%** |
| Region Avoidance Task | BehAV | 60% | 38.34% | 2.82 | 64% |
| | InstructNav | 40% | 17.68% | 5.10 | 61% |
| | INF | – | – | – | – |
| | NORM-Nav | **90%** | **58.92%** | **2.16** | **87%** |
| Traversable Obstacle Task | BehAV | 0% | – | – | – |
| | InstructNav | 0% | – | – | – |
| | INF | 70% | 52.14% | 2.23 | 67% |
| | NORM-Nav | **80%** | **67.26%** | **1.35** | **84%** |
| Combined Tasks | BehAV | 30% | 18.62% | 7.02 | 39% |
| | InstructNav | 20% | 12.21% | 6.45 | 36% |
| | INF | – | – | – | – |
| | NORM-Nav | **90%** | **54.87%** | **3.01** | **85%** |

- *Traversable-Obstacle Task:* Robots are required to navigate through regions that conventional LiDAR-based systems detect as obstacles but that are in fact traversable.
- *Combined Task:* Robots are required to perform all of the above simultaneously, following designated regions and avoiding restricted areas while satisfying behavioral constraints.

In all tasks, success is defined only when the robot reaches the goal while fully complying with the given instructions. The results are summarized in Table III, and the performance of each method is analyzed across the four task types.

In the region-following task, NORM-Nav achieves the best performance across all metrics. It follows instructions such as moving along a specified region or keeping to one side with high accuracy. The relatively small FD indicates close alignment with human-operated trajectories, while the high SPL shows improved efficiency. BehAV, relying only on VLM-based reasoning, completes some tasks but with inefficient paths. InstructNav depends on landmark-based navigation and frequently fails with large landmarks such as crosswalks. Both baselines deviate substantially from human-like trajectories when side-specific instructions are required.

In the region-avoidance task, BehAV shows higher trajectory similarity in successful cases due to its cost-map design for non-traversable regions. InstructNav produces unstable paths because of its reliance on landmarks. NORM-Nav completes avoidance tasks consistently and achieves the best performance across all metrics.

In the traversable-obstacle task, BehAV and InstructNav fail to traverse regions misclassified as obstacles. INF successfully handles traversable obstacles but does not account for behavioral preferences, leading to degraded performance in combined scenarios. NORM-Nav overcomes these limitations by traversing misclassified obstacles while simultaneously following user-specified constraints, giving it a clear advantage.

In the combined task, users provide instructions that inte-

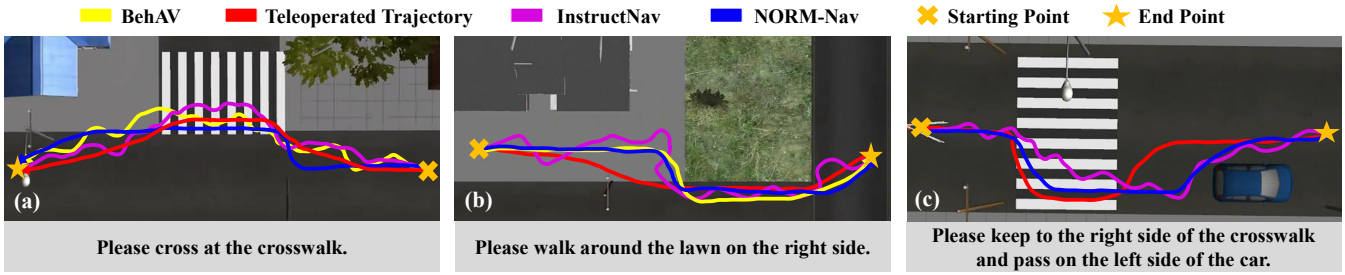


Fig. 6. Simulation results on three representative navigation tasks. The proposed method produces stable trajectories that closely follow human-operated reference paths, outperforming baseline approaches.

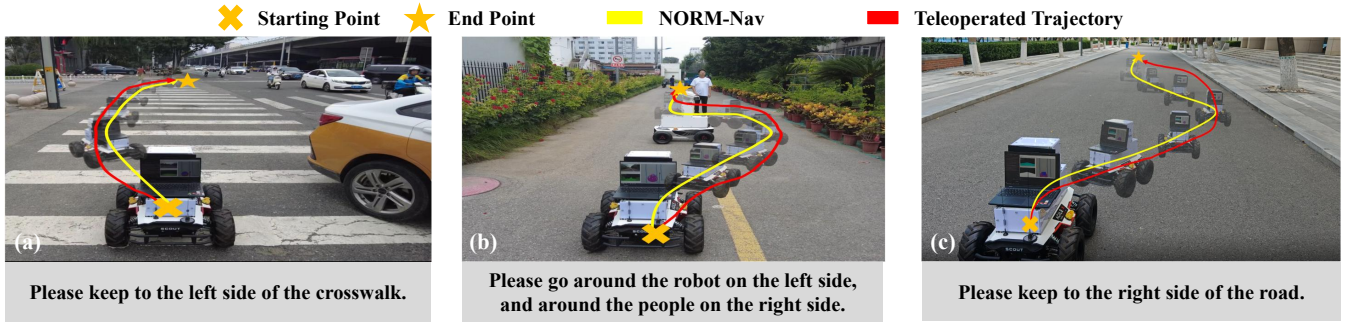


Fig. 7. Real-world demonstrations of behavior-constrained navigation. The proposed method successfully follows natural language instructions without collisions, generating stable trajectories that remain close to human-operated reference paths.

grate all subtasks. NORM-Nav successfully executes these instructions and achieves superior results in all metrics. The generated trajectories are more efficient, and closer to human-operated trajectories compared with the baselines.

### C. Evaluation of Cost Distribution Shaping

To evaluate the influence of the interpolation parameter $\alpha$ on navigation performance, we configured different $\alpha$ values in NORM-Nav. As defined in Eq. 14, $\alpha$ determines the non-linear interpolation of cost values, thereby shaping the distribution of preference enforcement around objects. The experimental results are illustrated in Fig. 8.

When $\alpha < 1.0$, the robot trajectories exhibit weak adherence to the directional preferences. When $\alpha = 1.0$, the generated paths moderately reflect the intended constraints. When $\alpha > 1.0$, the robot follows the instructions more strictly. In summary, Fig. 8 reveals that smaller $\alpha$ values favor conservative trajectories with greater obstacle clearance, while larger values enhance behavioral fidelity. Empirical observations suggest that settings with $\alpha > 1.0$ provide the most appropriate balance between safety and compliance with user-specified constraints.

### D. Evaluation of Speed Constraints

The ability of the proposed method to enforce speed-related behavioral constraints is evaluated through instructions specifying three velocity modes: quickly, slowly, and normal (default), as illustrated in Fig. 1. The results show that the robot adapts its translational velocity in real time according to the given commands: traversal speed increases in regions, decreases when instructed to move slowly near objects such as vehicles, and remains at a default pace when

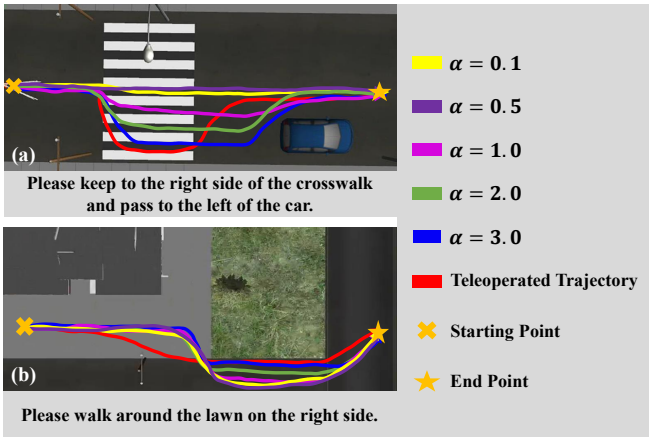


Fig. 8. Effect of interpolation parameter $\alpha$ on trajectory generation. Larger $\alpha$ values lead to stronger compliance with directional preferences, while smaller values produce more conservative paths.

no explicit constraints are imposed. These observations indicate that natural language instructions are reliably translated into consistent low-level velocity control within the proposed method.

### E. Real-World Task Demonstrations

We conduct representative real-world tests covering region following, region avoidance, traversable obstacle handling, and combined behaviors. Goal points are manually set, and behavioral constraints are given in natural language.

As shown in Fig. 7, the robot follows side-specific rules, such as keeping to the right of a road or bypassing pedestrians from the instructed side, and it reaches the goals without collisions. It also passes through a curtain that LiDAR

misclassifies as non-traversable and avoids a manhole cover from the left, even though the cover is not detected as an obstacle.

These demonstrations confirm that the method reliably executes natural language instructions and generates safe, efficient, and human-like trajectories. In particular, the method correctly handles obstacles with ambiguous traversability semantics, such as curtains or manhole covers, which are prone to misclassification in conventional LiDAR-based navigation.

## VI. Conclusion

In this work, we present NORM-Nav, a zero-shot navigation framework that integrates natural language behavioral constraints into costmap-based planning. The framework parses free-form instructions with a language model, grounds them through vision–LiDAR perception, and encodes them into multi-layer costmaps so that planned trajectories remain both geometrically feasible and socially compliant. Experiments in simulation and real-world environments show that NORM-Nav achieves higher success rates, greater efficiency, closer adherence to behavioral rules, and stronger alignment with human-preferred paths than representative baselines. The results indicate that NORM-Nav provides a practical and generalizable solution for behavior-aware navigation.

## References


[1] L. Yue, D. Zhou, L. Xie, F. Zhang, Y. Yan, and E. Yin, "Safe-vln: Collision avoidance for vision-and-language navigation of autonomous robots operating in continuous environments," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 4918–4925, 2024.

[2] D. Zhang, D. Huo, M. Zhou, and Z. Cao, "MPC-DS: A safe path tracking method for agvs in dynamic environments with dense obstacles," *IEEE Trans. Intell. Transp. Syst.*, 2025.

[3] J. Wang *et al.*, "Openbench: A new benchmark and baseline for semantic navigation in smart logistics," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2025, pp. 16 202–16 208.

[4] J. Wang, D. Huo, Y. Shi, C. Gao, Y. Qiao, and G. Zhou, "Open: Lightweight map-based semantic navigation for gps-free last-mile delivery," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 20 713–20 725, 2025.

[5] S. Macenski, F. Martín, R. White, and J. G. Clavero, "The marathon 2: A navigation system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2020, pp. 2718–2725.

[6] J. Lin *et al.*, "Advances in embodied navigation using large language models: A survey," *arXiv preprint arXiv:2311.00530*, 2025.

[7] W. Wu, T. Chang, X. Li, Q. Yin, and Y. Hu, "Vision-language navigation: A survey and taxonomy," *Neural Comput. Appl.*, vol. 36, no. 7, pp. 3291–3316, 2023.

[8] Y. Zhang *et al.*, "Vision-and-language navigation today and tomorrow: A survey in the era of foundation models," *Trans. Mach. Learn. Res.*, 2024.

[9] K. Weerakoon, A. J. Sathyamoorthy, U. Patel, and D. Manocha, "TERP: Reliable planning in uneven outdoor environments using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 9447–9453.

[10] J. Jiang, Y. Yang, Y. Deng, C. Ma, and J. Zhang, "BEVNav: Robot autonomous navigation via spatial-temporal contrastive learning in bird's-eye view," *IEEE Robot. Autom. Lett.*, vol. 9, no. 12, pp. 10 796–10 802, 2024.

[11] S. Minaee *et al.*, "Large language models: A survey," *arXiv preprint arXiv:2402.06196*, 2025.

[12] D. Shah, B. Osiński, B. Ichter, and S. Levine, "LM-Nav: Robotic navigation with large pre-trained models of language, vision, and action," in *Proc. Conf. Robot Learn. (CoRL)*. PMLR, 2023, pp. 492–504.

[13] K. Weerakoon *et al.*, "Behav: Behavioral rule guided autonomy using vlms for robot navigation in outdoor scenes," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2025, pp. 7044–7051.

[14] K. Zheng, "ROS navigation tuning guide," in *Robot Operating System (ROS): The Complete Reference (Vol. 6)*, A. Koubaa, Ed. Cham: Springer, 2021, pp. 197–226.

[15] E. Sani, A. Sgorbissa, and S. Carpin, "Improving the ros 2 navigation stack with real-time local costmap updates for agricultural applications," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2024, pp. 17 701–17 707.

[16] L. Mao, G. Warnell, P. Stone, and J. Biswas, "Pacer: Preference-conditioned all-terrain costmap generation," *IEEE Robot. Autom. Lett.*, 2025.

[17] Z. Zhang, A. Lin, C. W. Wong, X. Chu, Q. Dou, and K. S. Au, "Interactive navigation in environments with traversable obstacles using large language and vision-language models," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2024, pp. 7867–7873.

[18] R. Liu, W. Wang, and Y. Yang, "Volumetric environment representation for vision-language navigation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2024, pp. 16 317–16 328.

[19] J. Zhang *et al.*, "Navid: Video-based vlm plans the next step for vision-and-language navigation," *Robot. Sci. Syst.*, 2024.

[20] Z. Wang *et al.*, "Scaling data generation in vision-and-language navigation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 12 009–12 020.

[21] N. Hirose, C. Glossop, A. Sridhar, D. Shah, O. Mees, and S. Levine, "Lelan: Learning a language-conditioned navigation policy from in-the-wild videos," *arXiv preprint arXiv:2410.03603*, 2024.

[22] C. Glossop, W. Chen, A. Bhorkar, D. Shah, and S. Levine, "CAST: Counterfactual labels improve instruction following in vision-language-action models," *arXiv preprint arXiv:2508.13446*, 2025.

[23] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "Vlfm: Vision-language frontier maps for zero-shot semantic navigation," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2024, pp. 42–48.

[24] V. S. Dorbala, G. Sigurdsson, R. Piramuthu, J. Thomason, and G. S. Sukhatme, "Clip-nav: Using clip for zero-shot vision-and-language navigation," *arXiv preprint arXiv:2211.16649*, 2022.

[25] M. Ahn *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[26] G. Zhou, Y. Hong, Z. Wang, X. E. Wang, and Q. Wu, "Navgpt-2: Unleashing navigational reasoning capability for large vision-language models," in *Eur. Conf. Comput. Vis.* Springer, 2024, pp. 260–278.

[27] Y. Kuang, H. Lin, and M. Jiang, "Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models," *arXiv preprint arXiv:2402.10670*, 2024.

[28] Y. Long, W. Cai, H. Wang, G. Zhan, and H. Dong, "Instructnav: Zero-shot system for generic instruction navigation in unexplored environment," *arXiv preprint arXiv:2406.04882*, 2024.

[29] T. Ren *et al.*, "Grounded SAM: Assembling open-world models for diverse visual tasks," *arXiv preprint arXiv:2401.14159*, 2024.

[30] D. Deng, "DBSCAN clustering algorithm based on density," in *2020 7th Int. Forum on Elect. Eng. Automat. (IFEEA)*. IEEE, 2020, pp. 949–953.

[31] P. Anderson *et al.*, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[32] B. Aronov, S. Har-Peled, C. Knauer, Y. Wang, and C. Wenk, "Fréchet distance for curves, revisited," in *Eur. Symp. algorithmss*. Springer, 2006, pp. 52–63.