CoReLIN: Constraint-based Reasoning for Zero-shot Lifelong Interactive Navigation

Apoorva Vashisth[1*], Manav Kulshrestha[1], Pranav Bakshi[2], Damon Conover[3], Guillaume Sartoretti[4†], and Aniket Bera[1†]

[1] Purdue University, USA
[2] Indian Institute of Technology, Kharagpur, India
[3] DEVCOM Army Research Lab, USA
[4] National University of Singapore, Singapore
vashista@purdue.edu

Abstract. Robot navigation typically assumes an obstacle-free path exists between start and goal. In real environments, however, clutter may block all routes. We introduce **Lifelong Interactive Navigation**, where a mobile robot with manipulation capabilities must move objects to forge paths and complete sequential object-placement tasks. Because environment modifications persist, decisions impact future navigability and task difficulty. We propose **CoReLIN**, an LLM-driven constraint-based reasoning framework with active perception. CoReLIN reasons over a structured scene graph to decide which objects to relocate, where to place them, and where to explore next. A standard motion planner executes reliable navigation and manipulation primitives. To evaluate long-horizon behavior, we introduce 2 new metrics - **Long-term Efficiency Score (LES)**, a unified metric capturing success, execution efficiency, environment optimality, captured by **Price of Clutter**. In ProcTHOR-10k, CoReLIN outperforms best baseline by 16% under standard metrics and LES, and transfers to real-world hardware[5].

Keywords: Interactive Navigation · Task and Motion Planning · Embodied AI

1 Introduction

Visual navigation has achieved remarkable progress in recent years via advances in embodied AI [1, 2, 5, 14, 15, 17], generative models [3, 46], and deep learning [23, 30]. However, most navigation systems [1, 15] assume the existence of at least one obstacle-free path between the start and goal locations. This assumption holds in idealized cases, but may not hold in real-world scenarios such as homes and warehouses, where clutter, furniture, and everyday objects can obstruct all routes to the goal. In such settings, navigation alone is insufficient

* Corresponding Author
† These authors contributed equally and share senior authorship.
[5] Our code and dataset are available at project webpage.

Fig. 1: Demonstration of our approach, CoReLIN, deployed on the Boston Dynamics Spot robot. CoReLIN first observes the initial scene and determines the presence of task-relevant objects (e.g., Bottle and Desk) alongside environmental clutter. Based on the observations, current task progress, and cost-benefit analysis, our approach determines the best course of action ("Move PaperTowelRoll to BlackBox", "Move RedBottle to Desk2"). CoReLIN first optimizes the environment by moving the clutter (the paper towel roll) to a carefully chosen out-of-the-way spot (the black box). It can then place the red bottle on the desk.

and robots must take agency over their environment rather than merely reacting to it. With modern mobile platforms increasingly equipped with arms, grippers, and perception modules, they are no longer just sensors on wheels but embodied agents capable of reshaping their surroundings to achieve their goals. In these cases, an agent must do more than navigate: it must reconfigure its environment to make progress.

Existing visual navigation frameworks are typically brittle under such conditions. Reactive planners [9, 21] often detour around occlusions, while learned policies [24, 36] fail to generalize to new environments. Interactive navigation methods [8, 38, 39, 45, 50] have begun to address this gap, but are commonly defined for a single task at a time, often pre-arranged such that interaction is the only solution, and assume full observability of the environment. In contrast, real robots face sequential streams of tasks in the same dynamic space, where each decision – to move or not to move an object – carries lasting consequences.

In this work, we introduce the *Lifelong Interactive Navigation* problem, where a mobile manipulator receives a sequence of navigation-and-manipulation objectives in an unknown environment, each requiring placement of a given object (eg. Basketball, Alarm Clock) onto a target object (eg. Bed, Arm Chair). This requires the agent to continuously reason about where to explore, which obstacles to remove, which to bypass, and future impact of these choices. Crucially, planning and perception are tightly coupled: the agent not only plans about how to act but also where to look next to reveal task-relevant objects. While moving obstacles could open up previously unavailable routes at a nontrivial manipulation cost, detouring may save effort while restricting future mobility. Success thus depends not only on short-term feasibility but also on long-horizon, perception-aware reasoning over the evolving environment.

Large language models (LLMs) can reason over structured and semantic inputs [1, 15], with their strengths demonstrated in abstract reasoning rather than in producing fine-grained control sequences. Our key insight is to invert the typical interface between LLMs and embodied agents: we recast the LLM's role from

action sequence generator to a constraint reasoner, operating over the structure of the environment. A structured scene graph, continuously updated with perceptual feedback to include new observations, serves as the LLM's input representation. The frozen LLM then treats each blocking object as a decision point and determines whether to intervene – i.e., whether moving the obstacle will substantially expand navigable space given the movement cost – or if it can be safely detoured around without significant impact on long-term performance.

By re-framing planning as constraint resolution, we shift long-horizon reasoning from chaining low-level actions into making a small number of strategic, semantically grounded decisions. This enables zero-shot LLM planning, without task-specific fine-tuning, while tightly coupling perception and reasoning.

Building on this constraint-driven formulation, we evaluate our framework in the ProcTHOR-10k [7] simulator, which offers visually rich, physics-enabled environments. Our task suite is augmented with cluttered object configurations across environments consisting of up to 10 rooms. Each episode involves sequential tasks (e.g., bringing object X to receptacle Y), where the robot receives each new task only after completing the previous one. The robot has no prior knowledge of future tasks – if it fails to complete a task, the episode terminates immediately, and it does not receive any subsequent tasks. Hence, later in the episode, success depends not only on efficient exploration but also on strategic environment modification, as earlier decisions (such as whether to move or detour around obstacles) can significantly influence accessibility to regions containing the target objects, and later progress. By evaluating our system in this online task sequence framework, we test the real-world applicability of our approach, which must adapt in real time without knowledge of future goals. The key contributions of our work include:

- Lifelong Interactive Navigation: Extending prior interactive navigation problem to long-horizon, sequential tasks in unknown environments.
- Constraint-based planning framework: Shifting the decision space from low-level actions to high-level environmental constraints, enabling zero-shot long-horizon reasoning by large language models.
- Our method achieves the highest Long-term Efficiency Score (LES), improving over the strongest non-learned baseline by 16%, and outperforming prior interactive navigation methods by $3-6\times$ in complex environments.

2 Related Work

Embodied and Visual Navigation. Visual navigation has been extensively studied as mapping observations to actions under the implicit assumption of existence of at least one obstacle-free path between start and goal. Early works and benchmarks such as Cognitive Mapping and Planning [11], Target-driven Navigation [51], AI2-THOR [17], Gibson [43], Habitat [27, 34], and ProcTHOR [7] have driven progress toward robust policies in visually rich, partially observed environments. Subsequent methods improve exploration and representation via generative models, memory, and structured scene reasoning [3, 18, 20, 29, 41,

46, 47, 48], and tackle image-goal, language-driven, and multi-object navigation [10, 22, 33, 37, 44, 49]. However, these methods treat the environment as static and ultimately traversable: agents search more efficiently, predict unseen regions, exploit priors over layouts, or decompose the task, but if all geometric routes to the goal are blocked by clutter, they can only attempt detours or fail. In contrast, our problem setting explicitly targets cluttered, potentially non-traversable environments, enabling a mobile manipulator to reason about "whether, which, and where" to move obstacles, reconfiguring the environment to restore or improve connectivity for long-horizon task sequences.

Interactive Navigation & **Navigation Among Movable Obstacles (NAMO)**. A parallel line of work studies Navigation Among Movable Obstacles, where an agent may relocate objects to reach a blocked goal. Classical NAMO and rearrangement planners search over joint spaces of robot motion and obstacle configurations [4, 6, 19, 25, 26, 31, 32, 42], but assume full knowledge of geometry and dynamics, small number of obstacles, and short horizons, while optimizing reaching a single goal rather than reasoning about persistent environmental changes or future tasks. Interactive navigation approaches bring these ideas into embodied setting: Interactive Gibson [43] provides benchmarks for interaction in clutter; InterNav [45] and CaMP [39] learn how to move blocking objects based on visual input. Interactive-FAR [12], IN-Sight [28], ADIN [38], SPIN [35], Flax [8], NAMO-LLM [50] and related systems for legged manipulators or human-in-the-loop assistance explore fast heuristics, affordances, or uncertainty-aware querying for interactive routing. RoboEXP [16] and CuriousBot [40] interact to reveal unknown hidden spaces and discover object relations.

However, these methods are largely reactive, aiming to locally bypass or momentarily displace obstacles rather than plan how modifications shape future accessibility. NAMO-LLM [50] does not scale to our highly cluttered scenes (hundreds of obstacles) due to LLM context limits, and FLAX [8] targets simple gridworlds with few obstacle instances/types. Moreover, many approaches [12, 28, 35, 38, 39, 45] are episodic and reset after a single task, avoiding long-horizon manipulation consequences; in contrast, our Lifelong Interactive Navigation runs in a sequential no-reset setting, requiring decisions of which obstacles to move, where to place them, and when to abstain to optimize long-term connectivity and efficiency under uncertainty.

3 Our Approach

Overview. We target the Lifelong Interactive Navigation setting introduced in Sec. 1, where a mobile manipulator receives a sequence of navigation-and-manipulation tasks in a partially unknown, cluttered environment. The central challenge is that not all tasks are reachable by pure navigation: some obstacles must be moved, some rooms must be explored, and each intervention can affect future tasks. To handle this, we decouple strategic long-horizon choices from tactical low-level control.

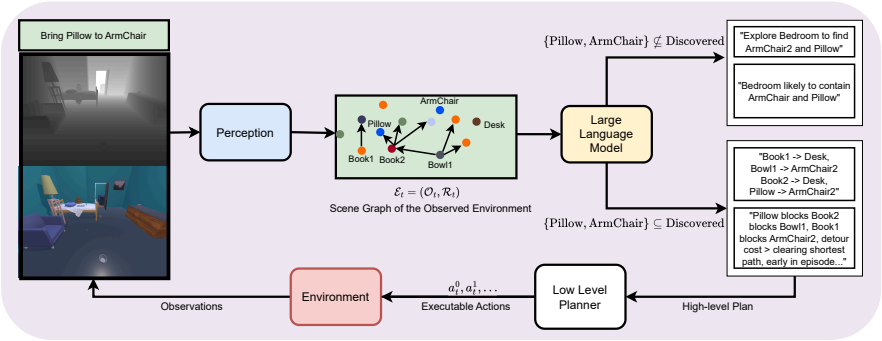

Fig. 2: Overview of our constraint-based planning framework for interactive navigation. At each timestep, our agent receives observations from the environment, which are utilized by our perception module to update the scene graph. Our scene graph contains the observed objects of the environment as nodes and the blocking relation among them as edges. Each node has its own attributes, which provide additional navigability and manipulability context to the LLM. The LLM then decides whether to explore the environment further to collect additional information or to attempt to complete the given task based on the scene graph content and the environment constraints.

At time t, the agent maintains a partial environment state $\mathcal{E}_t = (\mathcal{O}_t, \mathcal{R}_t)$, where $\mathcal{O}_t$ is the set of discovered objects/rooms and $\mathcal{R}_t$ encodes their blocking and reachability relations. Objects are persistent scene-graph nodes with unique IDs assigned at detection. Our large language model (LLM) operates on a structured textual serialization of $\mathcal{E}_t$ and reasons about how to resolve the current environmental constraints – deciding, for example, which obstacle to relocate, where to place it, or which room to investigate next. These high-level constraint resolutions are then converted by a low-level planner into concrete navigation and manipulation actions. CoReLIN operates in a closed-loop *perceive–plan–act* cycle, where each executed action produces new observations that update the scene graph and trigger replanning over the evolving environment. Hence, we recast the LLM as a constraint reasoner over the environment: it determines "what to change in the world" rather than "how to move the robot step by step".

Perception and structured scene construction. In this work, we assume that the general floor plan is known, with room contents unknown at first. We define a grid-graph $G_t = (N_t, E_t)$ denoting the discovered and traversable region of the world at time t. Here, nodes N_t correspond to the nodes that the robot has discovered to be free and can physically occupy, and edges E_t connect adjacent free nodes. The graph is constructed incrementally: nodes and edges are added only when free space is observed through the onboard perception system. All path planning and connectivity reasonings are performed over this partial graph.

After each observation, the perception module detects visible objects and free space, updates the grid-graph G_t with newly discovered traversable nodes,

and augments a structured scene representation that maintains both the set of discovered objects and the set of known but unexplored rooms. Although the total number of rooms and their coarse topological adjacencies are provided by the underlying floorplan, no geometric routes are assumed known a priori. The actual navigable passages between rooms – including whether doorways, corridors, or connecting regions are free, blocked, or partially occluded – must be discovered through exploration. As a result, the robot progressively expands the partial graph G_t and discovers the feasible routes between task-relevant objects throughout the task.

We represent the world state at time t as a directed graph $\mathcal{E}_t = (\mathcal{O}_t, \mathcal{R}_t)$, where nodes correspond to discovered objects or rooms, and edges encode blocking relations between them (e.g., an object obstructs the shortest path to another). Formally, for $o_i, o_j \in \mathcal{O}_t$,

$$C_{ij} = \begin{cases} 1, & \text{if } o_i \text{ blocks the shortest path to } o_j, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

We utilize the current grid graph G_t to describe the impact of an obstacle object on the navigability of the region. Let $n(o_i) \in N_t$ denote the node occupied by o_i. We compute the (normalized) betweenness centrality of $n(o_i)$:

$$\mathrm{bc}(n(o_i)) = \frac{1}{Z} \sum_{\substack{s \neq t \in N_t \\ s \neq n(o_i) \neq t}} \frac{|\sigma_{st}(n(o_i))|}{|\sigma_{st}|},$$

where $|\sigma_{st}|$ is the cardinality of the set of shortest paths σ_{st} between any node s and any node t where $s \neq t$, $|\sigma_{st}(n(o_i))|$ is the cardinality of the subset of the shortest paths that pass through $n(o_i)$, $\sigma_{st}(n(o_i))$, and Z is a normalization constant. This measures how many shortest paths depend on the node occupied by o_i, and thus how many paths would potentially become available if o_i were removed.

In our representation $\mathcal{E}_t$, each object node $o_i \in \mathcal{O}_t$ is augmented with attributes that capture both geometric and topological context for decision-making: (i) the cost of traversing the shortest path to the object, if such a path exists; (ii) the set of discovered objects that block this path, represented via $C_{ij} = 1$; (iii) the betweenness centrality $\mathrm{bc}(n(o_i))$ of the grid cell occupied by the object in the grid graph $G_t = (N_t, E_t)$, measuring its influence on global connectivity at time t; (iv) the cost of an alternative path that avoids all current blockers, if such a path exists. All path computations are performed on G_t, where only nodes and edges corresponding to currently observed free space are treated as traversable; unknown regions are never assumed passable until they are discovered. Unseen rooms are maintained as frontier nodes, serving as exploration candidates. After each executed action, $\mathcal{E}_t$ is updated to incorporate newly discovered objects, newly navigable rooms, revised exploration status, and obstacles that have been successfully removed.

LLM as a constraint-based planner. Given this text description of the scene graph, we use a frozen LLM to select the next high-level step. Intuitively, every blocking object is a decision point: should this object be relocated now, and if so, where, or can it be bypassed without significantly harming long-term performance? We formalize this as a cost-benefit selection over blocking objects.

For a candidate obstacle o_i, we define a removal cost that accounts for both the effort of manipulating the object and the effort of relocating it to an available location. To ensure they can be appropriately compared, both navigation and manipulation costs are estimated in terms of time:

$$\text{cost}(o_i, r_t, z_j) = d(r_t, o_i) + 2 \times e(o_i) + d(o_i, z_j),$$

where $d : N \times N \to \mathbb{R}^+$ describes the time taken to traverse the geodesic distance between two nodes. Hence, $d(r_t, o_i)$ is the cost of navigating from the robot's current position to the obstacle, $e(o_i)$ summarizes the manipulation effort for object o_i (time required to pick and place an object due to its physical attributes such as size and weight), and $d(o_i, z_j)$ is the cost of navigating the path from the obstacle's position to an available location z_j. The values of $d(r_t, o_i)$ and $d(o_i, z_j)$ are estimated from the paths on the known partial grid graph G_t. In our experiments, we simplify the manipulation cost estimation using a static value of $e(o_i) = 5.00$ (seconds) (counted twice for pick-and-place actions). This setting reflects our empirical observation that a pick/place cycle – including approach, alignment, grasp execution, verification, and safety pauses – dominates base travel time, and therefore the cost function should generally favor clearing tasks via short navigation movements over initiating unnecessary manipulations. The LLM reasons jointly over candidate pairs (o_i, z_j), evaluating both the effort required to clear the object and the suitability of each placement (whether the placement location is accessible, obstacles on the path, their attributes and their valid placement locations, etc.). We also present an ablation that varies the value of e to measure the sensitivity of our approach to this hyperparameter.

We intentionally simplify the manipulation aspect of our pipeline: the primary contribution of this work lies in high-level planning and task sequencing and not low-level affordance estimation. Under this assumption, a fixed effort term provides a reasonable approximation of the pick–place cycle duration. Conceptually, the decision faced by the planner at time t can be viewed as evaluating the trade-off

$$o^\star, z^\star = \underset{o_i \in \mathcal{O}_t,\, z_j \in \mathcal{Z}_t}{\arg\min} \big(\text{cost}(o_i, r_t, z_j) - \text{bc}(n(o_i))\big), \tag{2}$$

Importantly, Eq. (2) is not explicitly optimized by our system. Rather, it serves as an analytical abstraction to illustrate the structure and scale of the underlying decision space. In cluttered environments, the planner must implicitly reason

over all candidate obstacle–drop-zone pairs (o_i, z_j), a set that grows combinatorially with the number of movable objects and feasible placements. [6]

When no obstacle offers a sufficient gain, the planner first queries the current grid graph G_t to determine whether an alternate, obstacle-free path to the goal exists. If such a detour exists, the agent proceeds with a standard navigation sequence using the low-level planner, completing the task without manipulating any obstacles. If a detour does not exist, the planner attempts to move the minimum number of obstacles that would restore connectivity to achieve the task. If the goal objects have not been discovered yet, the planner initiates a task-directed exploration. Exploration is not a hard-coded perception routine: based on the current task and observed scene graph, the planner explicitly selects a room to explore to gather missing information from, then replans from the updated scene graph. During exploration, the agent ranks known but unexplored (or partially explored) rooms by semantic relevance to the task. For example, for the task "bring vase to dining table", rooms such as Living Room or Kitchen are prioritized over Bathroom or Bedroom. Semantic relevance serves only as a soft ordering, not a hard constraint. If the target objects are not found in higher-ranked rooms, the planner continues exploring remaining rooms in decreasing order of relevance, effectively performing exhaustive search over all reachable rooms. Once the relevant objects are discovered, the planner evaluates connectivity, identifies blocking obstacles, and decides which object to move and where based on obstacle attributes, drop-zone feasibility, and manipulation cost. If all reachable rooms are explored without finding the required objects, the episode ends with failure.

Deciding whether to move an obstacle depends on several factors such as remaining future tasks, long-term cost of detouring around the obstacle (if not moved), immediate cost of moving, and contextual constraints such as available locations for placement and unexplored regions. Hence, in our framework, the LLM operates as a constraint reasoner, combining these quantitative estimates with commonsense and task-level reasoning to generate interpretable, semantically grounded plans that generalize beyond the scope of a fixed heuristic.

Low-level planning and closed-loop execution. The execution of the LLM's output plan is delegated to a standard motion stack that outputs a sequence of executable actions $[a_t^0,\ a_t^1,\ a_t^2, \cdots]$. Prior to querying the LLM, the environment configuration $\mathcal{E}_t$ is filtered through the low-level planners to ensure that only reachable objects, feasible pick-and-place actions, and collision-free navigation targets are exposed in the decision space. This guarantees that the LLM reasons only over executable actions. For navigation, we run a Dijkstra-based planner over the currently discovered and reachable subgraph to obtain collision-free paths to the target objects, obstacle objects, rooms, or placement locations. For manipulation, we treat pick-place as a reliable primitive provided by the

[6] This formulation is merely intended to highlight the competing factors involved in the task – manipulation cost, travel distance, and long-term navigability – and is not utilized as an explicit heuristic during planning.

underlying robot control stack, and assume that designated objects have sufficient free space for placement. This separation of concerns lets the LLM focus on long-horizon, semantically meaningful choices about which constraints to resolve (which object to move, where to relocate it, and when to explore), while the low-level planner ensures reliable navigation and manipulation control in cluttered, physics-enabled environments under our assumptions.

4 Experiments

4.1 Setup

Dataset. We construct a dataset of cluttered indoor floors by instantiating ProcTHOR-10k simulator with procedurally generated clutter. We generate 10k episodes with 20 tasks each, following the setup described in Sec. 1. For our test set, we randomly draw 100 environments equally distributed among the number of rooms $1-10$. Following previous works [39], we group the environments by floorplan complexity and report the results in three categories: $1-3$, $4-6$, and $7-10$. To generate realistic clutter in the environment, we occlude a percentage of traversable nodes (based on floorplan area) in the grid graph $G = (N, E)$ of an environment without obstacles, where the probability of a node $n \in N$ being selected for occlusion is proportional to its betweenness centrality $\mathrm{bc}(n)$. This biases obstacle placement toward structurally critical regions such as hallways, bottlenecks, and intersections, without deterministically blocking all high-centrality nodes. We provide dataset visualization in Appendix A.

Evaluation Metrics. Our framework jointly optimizes three objectives: (i) the success rate $\mathrm{SR} = \frac{1}{N_{\text{episodes}}} \sum_{k=1}^{N_{\text{episodes}}} \frac{N_{\text{success}}^{(k)}}{N_{\text{goals}}^{(k)}}$, reflecting the agent's ability to complete assigned tasks (where $N_{\text{success}}^{(k)}$ and $N_{\text{goals}}^{(k)}$ denote the number of successfully completed and total assigned goals in episode k, respectively.); (ii) the time efficiency, measured by the number of timesteps required to finish each episode $\mathrm{TS} = \frac{1}{N_{\text{episodes}}} \sum_{k=1}^{N_{\text{episodes}}} T_k$, (where T_k is the total number of timesteps in episode k); and (iii) the environmental efficiency for navigation, which quantifies how the agent's actions improve or degrade the navigability of the environment. As explained in Appendix B, we quantify this via the ratio of the cumulative shortest-path lengths in the current cluttered environment to those in an obstacle-free configuration, termed as **Price of Clutter** - $\mathrm{PoC} = \frac{\sum_{s,t \in V} d_{\text{obs}}(s,t)}{\sum_{s,t \in V} d_{\text{free}}(s,t)}$, where $d_{\text{obs}}(s,t)$ is the shortest-path distance between nodes s and t in the cluttered environment, and $d_{\text{free}}(s,t)$ is the corresponding distance in the obstacle-free floorplan, both evaluated over the complete graph G.

In our problem setting, SR captures the task completion, TS measures execution speed, and PoC measuring lifelong environment that remains navigable for future goals. To consolidate these competing objectives into a single primary metric, we introduce the globally normalized **Long-term Efficiency Score (LES)**, which models the trade-off between short-term success (SR), temporal

efficiency (TS), and sustainable environment restructuring (PoC). Concretely, we first convert TS and PoC into higher-is-better utilities using global min-max normalization computed over the evaluation set:

$$u_{\mathrm{TS}} = 1 - \frac{\mathrm{TS} - \mathrm{TS}_{\min}}{\mathrm{TS}_{\max} - \mathrm{TS}_{\min}}, \quad u_{\mathrm{PoC}} = 1 - \frac{\mathrm{PoC} - \mathrm{PoC}_{\min}}{\mathrm{PoC}_{\max} - \mathrm{PoC}_{\min}}, \tag{3}$$

where $\mathrm{TS}_{\min}, \mathrm{TS}_{\max}, \mathrm{PoC}_{\min}, \mathrm{PoC}_{\max}$ are global extrema. We then aggregate the three objectives using a weighted geometric mean:

$$\mathrm{LES} = 100 \cdot \mathrm{SR}^{w_{\mathrm{SR}}} \cdot (u_{\mathrm{TS}} + \epsilon)^{w_{\mathrm{TS}}} \cdot (u_{\mathrm{PoC}} + \epsilon)^{w_{\mathrm{PoC}}}, \qquad w_{\mathrm{SR}} + w_{\mathrm{TS}} + w_{\mathrm{PoC}} = 1, \tag{4}$$

with $w_{\mathrm{SR}} = 0.5$, $w_{\mathrm{PoC}} = 0.25$, $w_{\mathrm{TS}} = 0.25$, and $\epsilon = 10^{-8}$ for numerical stability. A high LES indicates that a policy not only succeeds, but does so efficiently while preserving long-term navigability. By construction, higher SR increases LES, whereas larger TS or PoC decreases LES (via lower utilities), and the multiplicative form prevents a method from appearing strong by excelling in only one dimension (e.g., being fast but failing often, or succeeding while severely degrading the environment). We extensively describe our metric LES in Appendix C.

4.2 Baseline Comparison

We compare our method against 4 heuristic methods and 2 learning baselines, using identical goal sequences and obstacle placements across 3 goal horizons - $g \in \{10, 15, 20\}$. Since the 6 baselines assume ground-truth knowledge of the environment, we expose this information to one of our variants and report metrics for two versions of our approach - Ours(unk), which has no a priori knowledge of the world and must explore the environment, and Ours (known), which has ground-truth knowledge of the world. Our baselines are: (a) **ADIN** (learning-based) [38] (b) **InterNav (learning-based)** [45] (c) **FastDownward** (plans on the complete PDDL problem instance) [13] (d) **Always Detour** (pure navigation) (e) **Always Interact** (relocate obstacles on shortest path to the goal) (f) **Clean + S/P** (First relocate all obstacles in the environment, then perform pure navigation). We adapt and re-train the learning baselines on our dataset.

Across medium and large environments ($4-6$ and $7-10$ rooms), our approach achieves the best LES for every goal horizon, and the margin grows as scenes and horizons become complex. This is where lifelong structure matters: Always Interact myopically clears each current path, accumulating unnecessary relocations and future traversal cost; Always Detour avoids manipulation but preserves bottlenecks and fails when no free route exists; finally, Clean+S/P restores global connectivity only by paying a large, task-agnostic clearing cost. In contrast, our method explicitly balances task completion, time, and environmental impact, producing plans that remain efficient over long horizons without accumulating unnecessary clutter. In small environments ($1-3$ rooms), the best heuristics can marginally edge out our LES because navigation distances are inherently short and clutter has limited long-term impact. Simple strategies that

Episode Horizon	**1 – 3 rooms**				**4 – 6 rooms**				**7 – 10 rooms**			
	SR	PoC	TS	LES	SR	PoC	TS	LES	SR	PoC	TS	LES
g = 10												
CoReLIN (known)	98.79	1.29	578.88	<u>94.52</u>	100.00	1.28	1040.42	**91.60**	90.61	1.54	1594.36	**81.62**
CoReLIN (unk)	93.33	1.85	675.52	86.30	72.42	2.12	2099.79	65.64	65.45	3.00	2270.42	54.14
ADIN	8.57	1.81	419.64	26.90	3.93	1.44	955.25	17.91	3.21	2.90	1462.14	13.30
InterNav	1.64	1.90	496.09	37.30	0.88	1.49	2155.88	28.44	0.52	1.43	4511.78	21.71
FastDownward	38.12	2.47	87.50	53.28	17.50	2.41	67.94	36.43	7.74	3.14	30.42	21.45
Always Interact	100.00	1.19	681.79	**94.60**	100.00	1.37	1521.58	<u>87.57</u>	100.00	1.16	2565.52	<u>79.67</u>
Always Detour	87.88	2.34	364.48	81.00	86.36	2.10	613.52	81.24	66.06	3.23	758.76	59.39
Clean + S/P	100.00	1.00	835.72	94.48	100.00	1.00	1909.91	86.60	100.00	1.00	3865.10	57.91
g = 15												
CoReLIN (known)	100.00	1.22	814.48	<u>94.35</u>	97.78	1.20	1403.76	**90.18**	94.14	1.57	2356.94	**79.90**
CoReLIN (unk)	94.14	1.53	857.79	89.02	79.41	1.46	2049.79	76.17	64.85	2.88	2843.94	54.29
ADIN	13.81	1.81	602.86	8.77	4.76	1.43	1453.61	5.11	2.86	2.90	2178.04	3.12
InterNav	17.75	1.81	688.76	39.91	12.62	1.51	2950.67	34.36	4.17	1.43	6545.78	19.52
FastDownward	31.67	2.33	104.41	49.37	12.53	2.38	69.24	31.01	5.42	3.28	34.00	17.57
Always Interact	100.00	1.15	905.39	**94.43**	100.00	1.34	1983.30	<u>86.56</u>	100.00	1.15	3321.25	<u>77.16</u>
Always Detour	84.04	2.19	505.30	80.37	84.44	2.09	878.97	79.80	66.26	3.21	1094.18	59.16
Clean + S/P	100.00	1.00	1075.30	92.45	100.00	1.00	2383.58	85.78	100.00	1.00	4693.99	56.49
g = 20												
CoReLIN (known)	94.55	1.50	1044.55	89.20	97.73	1.25	1911.97	**88.37**	100.00	1.23	2435.48	**86.56**
CoReLIN (unk)	88.94	1.48	1339.36	83.87	81.67	1.81	3211.20	67.25	62.27	2.79	4497.80	49.99
ADIN	13.81	1.81	602.86	37.42	4.76	1.43	1453.61	20.03	2.86	2.90	2178.04	12.21
InterNav	16.67	1.96	842.00	33.78	6.00	2.13	4700.70	14.53	3.00	3.34	6089.91	0.20
FastDownward	27.66	2.32	123.19	46.14	9.85	2.38	73.12	27.50	4.06	3.28	35.81	15.21
Always Interact	100.00	1.14	1108.88	<u>94.18</u>	100.0	1.13	2416.67	<u>87.19</u>	100.00	1.36	3974.88	<u>74.73</u>
Always Detour	90.45	1.96	665.64	85.12	83.18	2.15	1115.27	78.26	64.09	3.43	1381.61	54.99
Clean + S/P	100.00	1.00	1200.48	**94.38**	100.00	1.00	2622.94	86.65	100.00	1.00	5169.58	61.86

Table 1: Baseline comparison of our approach CoReLIN with other methods across different floorplans (rooms 1 through 10) and varying episode horizons $g \in \{10, 15, 20\}$ (where g is the total number of tasks per episode). Bold indicates best performing method and underline indicates second-best performing approach.

aggressively manipulate obstacles can be "good enough" and occasionally slightly faster.

The learning-based baselines ADIN [38] and InterNav [45] underperform primarily due to a mismatch between their design assumptions and our lifelong setting. Both methods are largely optimized for single-goal interactive navigation, where interactions are chosen to unblock the current path to improve immediate reachability. In our problem, however, object placements persist and affect future tasks, causing myopic interactions to accumulate clutter and degrade long-term navigability (reflected in PoC), sharply lowering LES. As environments and horizons grow, this lack of explicit long-horizon reasoning causes their performance to degrade, whereas our method optimizes decisions for sustained efficiency across the entire sequence. These approaches were not designed to optimize a global, horizon-spanning objective that explicitly penalizes accumulated clutter, hence they struggle as soon as the sequential goals make "interaction for the current step" disadvantageous. Notably, our approach Ours (unk) remains

Manipulation Effort	**1 – 3 rooms**				**4 – 6 rooms**				**7 – 10 rooms**			
CoReLIN (unk)	SR↑	PoC↓	IE	PL (m)↓	SR↑	PoC↓	IE	PL (m)↓	SR↑	PoC↓	IE	PL (m)↓
$e = 1$	92.58	**1.44**	42.21	302.73	**83.79**	**1.46**	41.09	**646.35**	65.45	**2.65**	36.09	**999.48**
$e = 5$	85.30	1.48	36.31	295.04	79.55	1.81	38.48	825.78	**69.09**	2.79	32.09	1057.96
$e = 10$	92.42	1.53	23.27	301.07	80.45	1.62	39.76	809.92	60.61	2.94	25.48	1211.33
$e = 15$	91.21	1.59	15.33	287.38	80.91	1.87	21.88	822.90	63.39	3.00	20.89	1037.37
$e = 20$	**92.88**	1.62	9.58	**259.63**	78.48	1.90	21.36	715.82	60.30	2.95	18.58	1078.59

Table 2: Effect of varying manipulation effort e on the behavior of our approach CoReLIN. ↓ indicates lower value is better, and ↑ indicates higher value is better. Bold indicates the best performance of the metric.

competitive and consistently improves over the learning-based baselines, indicating that the gains in larger environments arise from better long-horizon decision-making under exploration rather than privileged access to ground truth. Hence, our constraint-based planner CoReLIN emerges as the most effective strategy for lifelong interactive navigation.

We provide the implementation details for our baselines in Appendix D , and additional performance analyses in Appendix E. Additionally, we evaluate our approach under a range of clutter generation policies (Appendix G) and observe consistent trends: our method remains robust and continues to outperform the baselines across policies.

4.3 Analyses

a. Effect of Manipulation Cost: To examine the behavior of our planner as the cost of manipulating objects e increases, we systematically vary the per-object manipulation cost $e \in [1, 5, 10, 15, 20]$ and measure SR, PoC, and total distance traversed by the robot (PL) across the test set (Tab. 2). We report an additional metric of Interaction Encounters (IE) that describes the percentage of obstacles the agent chose to move during the episode. As the value of e increases, IE decreases sharply across all settings, approximately $4\times$ between $e = 1$ and $e = 20$, demonstrating that the agent actively avoids unnecessary manipulation when it is costly to do so. Correspondingly, PoC rises, indicating that the planner tolerates higher clutter levels instead of restoring full connectivity at excessive expense. Despite this trade-off, SR remains largely stable, showing that the LLM-driven reasoning identifies and removes only critical obstacles essential for task completion. Path length (PL) also either remains bounded or slightly decreases, suggesting that when manipulation is de-incentivized, the agent effectively compensates by discovering alternate routes rather than interacting with high-cost obstacles. Overall, our results confirm that our constraint-based planner internalizes manipulation cost and dynamically strategizes by favoring selective, high-impact interactions over exhaustive environment cleanup, without significant degradation in SR.

History Length	1 – 3 rooms				4 – 6 rooms				7 – 10 rooms			
CoReLIN (unk)	SR↑	PoC↓	TS↓	LES↑	SR↑	PoC↓	TS↓	LES↑	SR↑	PoC↓	TS↓	LES↑
$h = 1$	89.85	**1.44**	1793.97	86.50	**78.94**	1.67	3516.03	71.84	55.00	**2.71**	5641.48	41.37
$h = 3$	88.18	1.49	1492.15	86.46	71.36	1.73	2926.48	70.42	**64.59**	2.78	3334.70	**52.60**
$h = 6$	91.88	1.52	1444.21	88.64	77.12	1.80	3175.39	71.54	57.43	2.91	3996.73	49.67
$h = 9$	**92.88**	1.52	**1403.88**	**88.77**	75.40	**1.45**	**2634.03**	**75.82**	53.24	2.79	**1228.72**	49.66

Table 3: Effect of varying historical context length h on the behavior of our approach. ↑ indicates higher value is better for the metric. Bold indicates the best performance.

LLM Name	1 – 3 rooms				4 – 6 rooms				7 – 10 rooms			
	SR↑	PoC↓	TS↓	LES↑	SR↑	PoC↓	TS↓	LES↑	SR↑	PoC↓	TS↓	LES↑
Gemini	<u>87.12</u>	1.79	2094.79	<u>79.14</u>	**84.09**	**1.67**	<u>3545.79</u>	**68.44**	**71.06**	**2.73**	4768.48	**42.53**
GPT-5	**92.27**	**1.44**	<u>1732.55</u>	**86.49**	<u>59.05</u>	<u>1.70</u>	4037.52	53.11	<u>49.39</u>	<u>2.87</u>	<u>4107.61</u>	<u>40.78</u>
Deepseek	76.21	<u>1.63</u>	**1549.21**	78.12	51.52	1.82	**2506.70**	<u>58.54</u>	38.03	2.94	**3305.52**	39.56

Table 4: Performance of different large language models in our framework. ↑ indicates higher value is better for the metric. Bold indicates the best performance while underline indicates the second-best performance.

b. History Length: We analyze how the amount of prior context h affects performance using LES as the primary metric (Tab. 3). In $1-3$ rooms, LES increases monotonically with h, reflecting slightly higher SR and markedly lower TS as the planner reuses past decisions and avoids redundant exploration/manipulation. In $4-6$ rooms, LES improves substantially with longer histories. Although SR fluctuates, both TS and PoC benefit from better global consistency – longer context helps the LLM recall earlier constraints and target fewer, more central obstacles. In $7-10$ rooms, $h = 3$ yields the best LES value; the gain at $h = 3$ is contributed by an improvement in SR (while maintaining comparable TS and PoC), indicating that extended context accelerates progress once the agent has accumulated a rich constraint history. Overall, longer histories improve or match the best LES across floorplans, as additional context improves constraint reasoning: the planner avoids revisiting explored frontiers, concentrates interactions on high-impact bottlenecks, and reduces detours—thereby balancing success, execution time, and long-term environment optimality.

c. Effect of Language Model Choice: We further evaluate the impact of different large language models within our framework in Tab. 4. All LLMs are queried with the same prompt (described in Appendix K) and structured scene-graph serialization, hence the differences primarily reflect model-dependent decision behavior. In $1-3$ rooms, **GPT-5** achieves the best LES via the highest SR, which we attribute to locally consistent choices that work well when the reasoning horizon is short. However, GPT-5 degrades sharply as complexity grows, suggesting brittleness under long-horizon coupling: it tends to over-commit to salient local obstacle relations and under-weight global connectivity and the downstream effects of persistent rearrangements, leading to more harmful place-

Obstacle Density	1 – 3 rooms				4 – 6 rooms				7 – 10 rooms			
	SR	PoC	IE	TS	SR	PoC	IE	TS	SR	PoC	IE	TS
$d = 0.5$	86.52	1.31	39.24	1678.06	80.91	1.26	47.79	3351.85	65.00	2.50	47.39	4748.91
$d = 1.0$	88.94	1.48	39.41	1339.36	81.67	1.81	39.79	3211.20	62.27	2.79	32.70	4497.80
$d = 2.0$	81.88	3.19	16.06	1592.34	55.15	3.67	27.39	3135.45	48.79	4.55	23.21	4773.21

Table 5: Performance of our approach as the obstacle density varies across floorplans.

ments and reduced SR. **Gemini** is most robust in larger environments, achieving the best LES in $4-6$ and $7-10$ rooms while maintaining the highest SR. This suggests it better internalizes the "persistent world" constraint, favoring interventions that preserve broader navigability and avoid cascading clutter. **DeepSeek** is less consistent overall, likely due to noisier interpretation of the structured serialization, but it surpasses GPT-5 in the $4-6$ room setting, indicating that failures are driven by planning priors rather than raw language ability.

d. Obstacle Density: We assess the robustness of CoReLIN as the obstacle density d increases from 0.5 (sparse) to 2.0 (dense), keeping task configurations fixed (Tab. 5). Overall performance declines with higher d, as reduced free space hinders long-horizon planning. In $1-3$ rooms, SR drops moderately, while PoC more than doubles, indicating diminished environment optimality. TS shows non-monotonic behavior as dense clutter sometimes shortens exploration but also causes premature failures. This trend intensifies in larger environments, where SR degrades sharply and PoC rises $2\times$ to $3\times$, reflecting that, while the agent must detour or manipulate more often, it struggles to restore global connectivity. The decline in interaction encounters (IE) suggests increased conservative approach: as clutter grows, the agent avoids manipulations with low expected payoff. Together, these results highlight that higher obstacle density compounds perception and reasoning difficulty, limiting the agent's ability to maintain long-term environmental optimality. We provide analysis of performance comparison of our approach with baselines across varying obstacle density in Appendix F.

Additionally, we present a detailed failure-case analysis of CoReLIN in Appendix H, along with a step-by-step run-through in Appendix I.

4.4 Hardware Experiments

We validate sim-to-real transfer by deploying our framework on a Boston Dynamics Spot with the Spot Arm (Fig. 1). Perception uses only the front fisheye cameras, while the wrist camera supports grasping and placement verification. Metric localization comes from the onboard state estimator, and stereo depth with category-level segmentation incrementally updates a structured scene graph in real time. Experiments are conducted in a three-room indoor environment with ≈ 50.0 m^2 navigable space, evaluating tasks similar to simulation and limited to objects within the gripper envelope. The planner interleaves exploration and

targeted rearrangement based on incremental scene updates from fisheye RGB-D observations. This setup preserves our lifelong, partially observed, cluttered assumptions without task-specific fine-tuning or hardware-specific prompts. Qualitatively, behavior mirrors simulation: selective removal of central obstacles, efficient detours when beneficial, and robust retrieve-and-place execution under real sensing and actuation noise. Implementation details are provided in Appendix J.

5 Limitations and Future Work

Several extensions to CoReLIN remain promising. We currently model manipulation effort with a fixed cost; future work could infer object-specific effort directly from RGB-D observations to enable richer cost-aware reasoning. Our interaction space is limited to pickup-and-place of small obstacles, and extending it to include pushing, whole-body interaction, or larger objects would broaden applicability. We also decouple high-level constraint reasoning from low-level motion control for modularity; tighter integration could further improve whole-body efficiency. Scaling to extreme clutter and richer partial observability remains challenging, motivating stronger uncertainty-aware perception and structural priors. Finally, extending our framework to multi-agent collaborative settings – where multiple robots coordinate exploration and environment restructuring – offers an exciting direction for future lifelong embodied systems.

6 Conclusion

Our approach redefines the role of large language models in embodied decision-making. Specifically, we recast the role of LLMs as constraint reasoners rather than sequence generators. Our method leverages structured scene graphs and cost–benefit abstractions to choose which environmental constraints to resolve, such as which obstacles to relocate where, or which regions to explore, based on their long-term impact on performance. Through extensive experiments, we showcase that our formulation enables zero-shot, long-horizon planning in unknown environments, achieving competitive task success with substantially fewer manipulations and lower cumulative cost compared to fully informed baselines. Finally, we validate the feasibility of our method on a Boston Dynamics Spot robot equipped with a depth camera and manipulator arm, demonstrating effective active perception and environment restructuring.

7 Acknowledgements

This work was supported by the DEVCOM Army Research Laboratory under cooperative agreement W911NF2520170.

Bibliography

[1] Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al.: Do as i can, not as i say: Grounding language in robotic affordances. arXiv preprint arXiv:2204.01691 (2022)

[2] Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., Van Den Hengel, A.: Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3674–3683 (2018)

[3] Bar, A., Zhou, G., Tran, D., Darrell, T., LeCun, Y.: Navigation world models. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 15791–15801 (2025)

[4] Chadzelek, T., Eckstein, J., Schömer, E.: Heuristic motion planning with movable obstacles. In: CCCG. pp. 131–136 (1996)

[5] Chaplot, D.S., Gandhi, D., Gupta, S., Gupta, A., Salakhutdinov, R.: Learning to explore using active neural slam. In: 8th International Conference on Learning Representations, ICLR 2020 (2020)

[6] Chen, P., Hwang, Y.: Practical path planning among movable obstacles. In: Proceedings. 1991 IEEE International Conference on Robotics and Automation. pp. 444–449. IEEE (1991)

[7] Deitke, M., VanderBilt, E., Herrasti, A., Weihs, L., Ehsani, K., Salvador, J., Han, W., Kolve, E., Kembhavi, A., Mottaghi, R.: Procthor: Large-scale embodied ai using procedural generation. Advances in Neural Information Processing Systems **35**, 5982–5994 (2022)

[8] Du, Q., Li, B., Du, Y., Su, S., Fu, T., Zhan, Z., Zhao, Z., Wang, C.: Fast task planning with neuro-symbolic relaxation. IEEE Robotics and Automation Letters (2026)

[9] Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE robotics & automation magazine **4**(1), 23–33 (1997)

[10] Gadre, S.Y., Wortsman, M., Ilharco, G., Schmidt, L., Song, S.: Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23171–23181 (2023)

[11] Gupta, S., Davidson, J., Levine, S., Sukthankar, R., Malik, J.: Cognitive mapping and planning for visual navigation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2616–2625 (2017)

[12] He, B., Chen, G., Wang, W., Zhang, J., Fermuller, C., Aloimonos, Y.: Interactive-far: Interactive, fast and adaptable routing for navigation among movable obstacles in complex unknown environments. In: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5402–5409. IEEE (2024)

[13] Helmert, M.: The fast downward planning system. Journal of Artificial Intelligence Research **26**, 191–246 (2006)

[14] Huang, W., Wang, C., Zhang, R., Li, Y., Wu, J., Fei-Fei, L.: Voxposer: Composable 3d value maps for robotic manipulation with language models. In: Conference on Robot Learning. pp. 540–562. PMLR (2023)

[15] Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al.: Inner monologue: Embodied reasoning through planning with language models. In: 6th Annual Conference on Robot Learning (2022)

[16] Jiang, H., Huang, B., Wu, R., Li, Z., Garg, S., Nayyeri, H., Wang, S., Li, Y.: Roboexp: Action-conditioned scene graph via interactive exploration for robotic manipulation. In: Conference on Robot Learning. pp. 3027–3052. PMLR (2025)

[17] Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Deitke, M., Ehsani, K., Gordon, D., Zhu, Y., et al.: Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474 (2017)

[18] Kwon, O., Kim, N., Choi, Y., Yoo, H., Park, J., Oh, S.: Visual graph memory with unsupervised representation for visual navigation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15890–15899 (2021)

[19] Levihn, M., Scholz, J., Stilman, M.: Hierarchical decision theoretic planning for navigation among movable obstacles. In: Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics. pp. 19–35. Springer (2013)

[20] Li, H., Wang, Z., Yang, X., Yang, Y., Mei, S., Zhang, Z.: Memonav: Working memory model for visual navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17913–17922 (2024)

[21] Likhachev, M., Ferguson, D.: Planning long dynamically feasible maneuvers for autonomous vehicles. The International Journal of Robotics Research **28**(8), 933–945 (2009)

[22] Marza, P., Matignon, L., Simonin, O., Wolf, C.: Multi-object navigation with dynamically learned neural implicit representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11004–11015 (2023)

[23] Mattamala, M., Frey, J., Libera, P., Chebrolu, N., Martius, G., Cadena, C., Hutter, M., Fallon, M.: Wild visual navigation: Fast traversability learning via pre-trained models and online self-supervision. Autonomous Robots **49**(3), 19 (2025)

[24] Mousavian, A., Toshev, A., Fišer, M., Košecká, J., Wahid, A., Davidson, J.: Visual representations for semantic target driven navigation. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 8846–8852. IEEE (2019)

[25] Nieuwenhuisen, D., van der Stappen, A.F., Overmars, M.H.: An effective framework for path planning amidst movable obstacles. In: Algorithmic

Foundation of Robotics VII: Selected Contributions of the Seventh International Workshop on the Algorithmic Foundations of Robotics. pp. 87–102. Springer (2008)

[26] Okada, K., Haneda, A., Nakai, H., Inaba, M., Inoue, H.: Environment manipulation planner for humanoid robots using task graph that generates action sequence. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566). vol. 2, pp. 1174–1179. IEEE (2004)

[27] Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., et al.: Habitat: A platform for embodied ai research. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9339–9347 (2019)

[28] Schoch, P., Yang, F., Ma, Y., Leutenegger, S., Hutter, M., Leboutet, Q.: In-sight: Interactive navigation through sight. In: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 7794–7800. IEEE (2024)

[29] Singh, K.P., Salvador, J., Weihs, L., Kembhavi, A.: Scene graph contrastive learning for embodied navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10884–10894 (2023)

[30] Song, X., Chen, W., Liu, Y., Chen, W., Li, G., Lin, L.: Towards long-horizon vision-language navigation: Platform, benchmark and method. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12078–12088 (2025)

[31] Stilman, M., Kuffner, J.: Planning among movable obstacles with artificial constraints. The International Journal of Robotics Research **27**(11-12), 1295–1307 (2008)

[32] Stilman, M., Kuffner, J.J.: Navigation among movable obstacles: Real-time reasoning in complex environments. International Journal of Humanoid Robotics **2**(04), 479–503 (2005)

[33] Sun, X., Chen, P., Fan, J., Chen, J., Li, T., Tan, M.: Fgprompt: Fine-grained goal prompting for image-goal navigation. Advances in Neural Information Processing Systems **36**, 12054–12073 (2023)

[34] Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D.S., Maksymets, O., et al.: Habitat 2.0: Training home assistants to rearrange their habitat. Advances in neural information processing systems **34**, 251–266 (2021)

[35] Uppal, S., Agarwal, A., Xiong, H., Shaw, K., Pathak, D.: Spin: Simultaneous perception interaction and navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18133–18142 (2024)

[36] Wang, H., Wang, W., Liang, W., Hoi, S.C., Shen, J., Gool, L.V.: Active perception for visual-language navigation. International Journal of Computer Vision **131**(3), 607–625 (2023)

[37] Wang, H., Liu, P., Cai, W., Wu, M., Qian, Z., Dong, H.: Mo-ddn: a coarse-to-fine attribute-based exploration agent for multi-object demand-driven

navigation. Advances in Neural Information Processing Systems **37**, 64176–64214 (2024)
[38] Wang, X., Liu, Y., Song, X., Liu, Y., Zhang, S., Jiang, S.: An interactive navigation method with effect-oriented affordance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16446–16456 (2024)
[39] Wang, X., Liu, Y., Song, X., Wang, B., Jiang, S.: Camp: Causal multi-policy planning for interactive navigation in multi-room scenes. Advances in Neural Information Processing Systems **36**, 15855–15868 (2023)
[40] Wang, Y., Fermoselle, L., Kelestemur, T., Wang, J., Li, Y.: Curiousbot: Interactive mobile exploration via actionable 3d relational object graph. IEEE Robotics and Automation Letters (2026)
[41] Werby, A., Huang, C., Büchner, M., Valada, A., Burgard, W.: Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. In: First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024 (2024)
[42] Wilfong, G.: Motion planning in the presence of movable obstacles. In: Proceedings of the fourth annual symposium on Computational geometry. pp. 279–288 (1988)
[43] Xia, F., Zamir, A.R., He, Z., Sax, A., Malik, J., Savarese, S.: Gibson env: Real-world perception for embodied agents. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 9068–9079 (2018)
[44] Yu, X., Zhang, S., Song, X., Qin, X., Jiang, S.: Trajectory diffusion for objectgoal navigation. Advances in Neural Information Processing Systems **37**, 110388–110411 (2024)
[45] Zeng, K.H., Weihs, L., Farhadi, A., Mottaghi, R.: Pushing it out of the way: Interactive visual navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9868–9877 (2021)
[46] Zeng, Y., Ren, H., Wang, S., Huang, J., Cheng, H.: Navidiffusor: Cost-guided diffusion model for visual navigation. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 11994–12001. IEEE (2025)
[47] Zhai, A.J., Wang, S.: Peanut: Predicting and navigating to unseen targets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10926–10935 (2023)
[48] Zhang, S., Song, X., Bai, Y., Li, W., Chu, Y., Jiang, S.: Hierarchical object-to-zone graph for object navigation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15130–15140 (2021)
[49] Zhang, S., Yu, X., Song, X., Wang, X., Jiang, S.: Imagine before go: Self-supervised generative map for object goal navigation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16414–16425 (2024)
[50] Zhang, Y., Kantaros, Y.: Namo-llm: Efficient navigation among movable obstacles with large language model guidance. IEEE Robotics and Automation Letters (2025)

[51] Zhu, Y., Mottaghi, R., Kolve, E., Lim, J.J., Gupta, A., Fei-Fei, L., Farhadi, A.: Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: 2017 IEEE international conference on robotics and automation (ICRA). pp. 3357–3364. ieee (2017)