SELFSKILL: SELF-STATE CONDITIONED SKILL DISCOVERY FOR EMBODIED AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Embodied agents have achieved remarkable progress in interacting with the physical world, yet existing evaluations primarily focus on task-level outcomes, which reveal whether an agent succeeds but provide little insight into what capabilities an agent possesses. In this work, we introduce SelfSkill, a framework for discovering skills of embodied agents. The key insight is that the skills are not fixed behavioral patterns, but emerge from how an agent utilizes different environmental factors under its current embodied conditions, which we characterize as self-states. To characterize such skills, SelfSkill constructs a initial behavioral basis space, where each basis represents a potential behavioral influence induced by an environmental factor. Given trajectory observations, SelfSkill infers the activation pattern over these bases under different self-states, producing a skill that captures how an agent generates behaviors under varying self-states. By associating discovered skills with trajectory outcomes, SelfSkill reveals the strengths and limitations of different agents across self-states. Furthermore, the discovered skills provide interpretable guidance for agent enhancement by identifying complementary capabilities among different agents and transferring effective behavioral knowledge. Experiments on ObjectGoal Navigation and Open-Vocabulary Mobile Manipulation demonstrate that SelfSkill offers a new perspective on understanding, evaluating, and improving embodied agents.

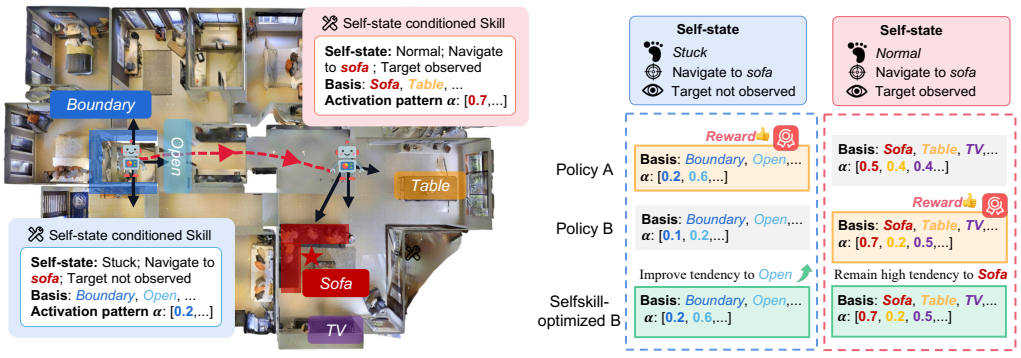
1 INTRODUCTION

Embodied agents have achieved remarkable progress in interacting with the physical world, driven by advances in foundation models, vision-language models, and large-scale robot learning (Reed et al., 2023; Kim et al., 2024; Li et al., 2025; Gao et al., 2026). These advances have enabled increasingly complex tasks, including language-guided manipulation, object navigation, and long-horizon interaction. However, despite their growing capabilities, our understanding of embodied agents remains limited. Current evaluations mainly rely on task-level metrics, such as success rate, efficiency, and accumulated rewards, which only indicate whether an agent completes a task but provide limited insight into the underlying capabilities that lead to success or failure. Therefore, beyond measuring final outcomes, understanding the functional capabilities underlying embodied behaviors is essential for analyzing and improving embodied agents.

Recent advances in general-purpose AI agents highlight the importance of capability-level understanding. By maintaining memories, reflecting on previous interactions, and accumulating reusable knowledge, these agents can identify effective strategies and determine when certain abilities should be applied (Packer et al., 2023; Zhou et al., 2025a; Jiang et al., 2026). Such capability awareness allows agents to reason about what they can do under different circumstances and improve future decision making. However, a similar understanding remains largely unexplored for embodied agents. Although embodied agents naturally generate rich interaction trajectories, existing approaches mainly treat these trajectories as evidence for task completion, rather than as observations of the underlying capabilities that produce the behaviors.

A key challenge in skill discovery is that behaviors are not fixed responses to environmental observations. They are shaped by the agent's embodied conditions, such as its motion capability, perceptual state, interaction status, and task objective, which determine how environmental factors are utilized during decision making. We refer to these conditions as self-states. This suggests that behavior is

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Self-state Conditioned Skills in Navigation

(b) Skill-guided Policy Improvement via Activation Pattern Transfer

Figure 1: **Self-state conditioned skill discovery and enhancement.** (a) Different self-states lead to different skills, where the environmental factors play different functional roles. (b) SelfSkill identifies more effective skills from successful behaviors and transfers them to improve existing navigation policies.

jointly determined by environmental factors and the agent's self-states (Gibson, 1979), rather than by environmental observations alone. For example, a frontier region may provide an exploration opportunity when the target is unobserved, while the same region may become less relevant for navigation once target information has been acquired. The observation indicate that skills should not be modeled as fixed behaviors, but as functional relations describing how agents utilize different environmental factors under varying embodied conditions.

Motivated by this observation, we introduce SelfSkill, a unified framework for discovering and analyzing self-state-conditioned skills of embodied agents as Fig 1(a). The key idea is to construct behaviors through a initial behavioral basis space, where each basis captures a potential behavioral influence induced by an environmental factor under the current slef-state, and then infer the activation pattern over the initial basis space by combining behavioral reconstruction with semantic priors. Each self-state conditioned skill can be represented by a behavioral basis space and the corresponding activation pattern over these bases. The activation pattern describes how the agent utilizes different behavioral factors to generate behaviors under the self-state.

By associating discovered skills with trajectory outcomes, SelfSkill enables fine-grained capability evaluation beyond overall task performance. Comparing skills across different agents reveals complementary capabilities and provides interpretable guidance for policy fusion and skill-guided enhancement as Fig 1(b). Rather than only comparing whether agents succeed, SelfSkill reveals skills underlying agent behaviors and explains why different agents perform differently under varying self-states.

2 Related Works

2.1 ObjectNav and OVMM Task

Object goal navigation (ObjectNav) requires an agent to navigate in unseen environments and locate objects specified by semantic goals. Early ObjectNav methods mainly learned navigation policies through reinforcement learning (Zhu et al., 2017; Chaplot et al., 2020; Zhang et al., 2025b; 2023; 2022) or supervised learning (Ramakrishnan et al., 2022; Zhang et al., 2024b; Yu et al., 2024). More recently, large models have enabled zero-shot ObjectNav without task-specific training (Gadre et al., 2023; Majumdar et al., 2022; Zhou et al., 2023b). LLM-based methods reason over textual or graph-based scene representations to infer likely goal locations (Wu et al., 2024b; Zhou et al., 2023b; Yin et al., 2024b; Chen et al., 2023; Yin et al., 2025b), while VLM-based methods guide navigation using semantic similarity between observations and targets (Li et al., 2022; Gadre et al., 2023; Huang et al., 2024; Yokoyama et al., 2024a; Wang & Lee, 2025). MLLM-based methods further integrate observations, instructions, and navigation history for multimodal scene understanding and spatial reasoning (Wang et al., 2025b; Zhong et al., 2024).

Open-Vocabulary Mobile Manipulation (OVMM) requires a robot in an unseen environment to navigate to a target object, pick it up, move to a target receptacle, and place it according to a user instruction. Existing methods mainly follow end-to-end or modular paradigms. End-to-end approaches directly control whole-body motions (Fu et al., 2024; Yan et al., 2025), while modular methods decompose the task into stages (Yenamandra et al., 2023; Liu et al., 2024). Recent work further improves individual stages, including gaze control (Gupta et al., 2024), placement (Ramrakhya et al., 2024; Wang et al., 2025a), and docking (Wu et al., 2025a).

Despite progress in ObjectNav and OVMM, existing methods mainly evaluate task-level performance, such as success rate, efficiency, providing limited insight into the behavioral capabilities behind agent decisions. Agents with similar performance may rely on different strategies under different embodied conditions. SelfSkill complements these evaluations by uncovering latent skills and analyzing how they are utilized across varying self-states.

2.2 Skill Discovery and Learning

Skill discovery aims to automatically identify reusable behavioral patterns from agent interactions. Early approaches formulate skills as temporally extended behaviors through hierarchical reinforcement learning. The option framework (Sutton et al., 1999) introduces temporal abstraction by representing reusable behaviors as options, while subsequent methods such as Option-Critic (Bacon et al., 2017) and FeUdal Networks (Vezhnevets et al., 2017) learn hierarchical policies with automatically discovered sub-behaviors. Beyond hierarchical policy learning, unsupervised skill discovery methods aim to discover diverse behaviors without external task rewards. DIAYN (Eysenbach et al., 2019) learns a set of distinguishable skills. DADS (Sharma et al., 2020a) further incorporates environment dynamics and discovers skills with predictable state transitions. Other approaches explore improved skill diversity, disentanglement, and transferability through alternative objectives and representations (Achiam & Sastry, 2018; Campos et al., 2020; Sharma et al., 2020b). Recent studies have further explored skill discovery in robotic settings, where reusable behaviors are learned from demonstrations or real-world interactions. Such approaches aim to acquire transferable manipulation or locomotion skills (Schaal, 2006; Paraschos et al., 2013; Konidaris et al., 2018).

However, existing skill discovery methods generally focus on learning reusable behaviors or latent policies, with each skill associated with a relatively consistent behavioral outcome. They do not explicitly model how the functional meaning of a behavior varies with the agent's embodied condition. In embodied environments, the same environmental factor may induce different behaviors depending on the agent's perceptual, motion, and interaction status, as well as its task objective. Different from prior skill discovery methods, SelfSkill does not learn new policies or behavioral primitives. Instead, it discovers latent functional skills in existing embodied agents by identifying how behavioral factors are activated under different self-states.

3 Method

3.1 Overview

In this section, we introduce SelfSkill, a unified framework as in Fig. 2 for discovering and analyzing the skills underlying embodied behaviors of a given agent policy.

Given an agent policy π interacting with a particular environment, we model its behavioral tendency at a time step t as a overall linear motion direction vector v_t in the coming period of time. The behavioral tendency is influenced by both the previous motion direction v_{t-1} and environment-induced behavioral factors. Specifically, we represent the environmental influence as directional vectors f_i of attraction or repulsion from the agent toward different semantic factors, including object clusters, frontier regions, open-space regions, and boundaries. We consider that the motion tendency can be approximated as:

$$v_t \approx \rho v_{t-1} + \sum_i \alpha_i f_i, \tag{1}$$

where ρ and α_i represent the activation degrees of the previous motion tendency and environmental behavioral factors, respectively. We hypothesize that these activation patterns vary with the agent's

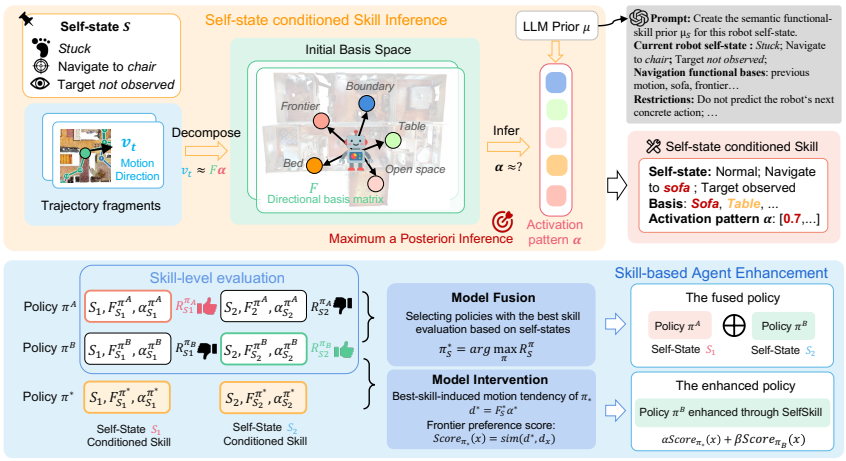


Figure 2: Overview. Given an agent self-state, SelfSkill infers self-state conditioned skill activations from trajectory fragments with LLM-guided priors. The discovered skills are evaluated across policies and transferred to enhance existing agents by skill fusion and policy intervention.

embodied conditions and reflect how the policy π responds to those different conditions, including its movement status (e.g., whether it is stuck), perceptual status (e.g., whether the target is observed), interaction status (e.g., whether a grasp can be executed) and task objective. Although environmental layouts may vary, a policy under similar embodied conditions can exhibit consistent activation patterns over behavioral factors across different environments. Such environment-reusable activation patterns changing along with conditions provide an abstract representation of behavioral skills.

Therefore, the key idea of SelfSkill is that skills are not fixed behavioral patterns, but should be modeled conditioning on self-states $\mathcal{S} = \{S_k\}_{k=1}^N$, as formalized in Sec. 3.2, allowing diverse behaviors can be characterized through different activation patterns.

To recover reusable activation patterns, SelfSkill constructs a shared initial behavioral basis space $\mathcal{F}$ across different agent policies and environments in Sec. 3.3. Each basis describes a potential behavioral influence induced by either the previous motion tendency or environmental factors.

As described in Sec. 3.4, given trajectory fragments generated by π under a specific self-state $S_k \in \mathcal{S}$, SelfSkill infers the activation pattern over the shared initial basis space. Since different behavioral factors may induce similar motion tendencies, the activation inference is ambiguous. To address this issue, SelfSkill incorporates a semantic prior from a large language model (LLM) to regularize the inference process, enabling the activation pattern to capture both observed motion tendencies and semantic consistency. Bases with significant activation values are selected to form the self-state-conditioned behavioral basis space $\mathcal{F}_k^\pi \subseteq \mathcal{F}$, and their corresponding activation values constitute the skills. Therefore, the skill representation of π is formulated as:

$$\mathcal{K}^\pi = \{S_k, \mathcal{F}_k^\pi, \alpha_k^\pi\}_{k=1}^N, \tag{2}$$

where N denotes the number of self-states, $\mathcal{F}_k^\pi$ represents the self-state S_k conditioned behavioral basis space, and α_k^π denotes the corresponding activation pattern over these selected bases.

Finally, we evaluate the skills of agent policies with task outcomes, enabling interpretable capability analysis and transferable behavioral knowledge across different policies in Sec. 3.5.

3.2 Self-State Representation

The self-state characterizes the current embodied condition of an agent, including its movement status, interaction status, perceptual status, and task objective. At each time step, we represent the self-state as $S = (S^{body}, S^{hand}, S^{eye}, S^{task})$. Specifically, S^{body} describes the movement status of the agent, indicating whether the agent can effectively move in the environment. $S^{body} \in \mathcal{S}^{body}$, which is defined as $\mathcal{S}^{body} = \{\text{normal motion}, \text{stuck}\}$. This status can be estimated by a vision language model (VLM) based on the agent's motion history and semantic map observations. S^{hand} describes the interaction status of the agent, indicating whether the agent can grasping an object from start-receptacle or placing an object on goal-receptacle, $S^{hand} \in \mathcal{S}^{hand}$

and $\mathcal{S}^{hand} = \{\text{grasping-capable}, \text{grasping-incapable}, \text{placing-capable}, \text{placing-incapable}\}$. This interaction status can be estimated by a VLM from RGB-D observations. Detailed prompts are provided in the App. A.2. S^{eye} represents the perceptual status of the agent with respect to the task objective, indicating whether the target has been observed. The corresponding state space is defined as $\mathcal{S}^{eye} = \{\text{target observed}, \text{target not observed}\}$. S^{task} represents the task objective, specifying the goal that the agent is required to accomplish. It is defined as $\mathcal{S}^{task} \in \{\text{navigate to a dining table}, \text{grasp a cup}, \text{place a cup}, ...\}$.

Depending on the task, different subsets of self-state dimensions are instantiated. For ObjectNav and OVMM, the self-state is represented as:

$$S_{\text{ObjectNav}} \in \mathcal{S}^{body} \times \mathcal{S}^{eye} \times \mathcal{S}^{task}, \text{ and } S_{\text{OVMM}} \in \mathcal{S}^{body} \times \mathcal{S}^{hand} \times \mathcal{S}^{eye} \times \mathcal{S}^{task}. \tag{3}$$

3.3 Initial Behavioral Basis Space

To recover reusable behavioral patterns, SelfSkill first constructs a shared initial behavioral basis space $\mathcal{F}$. A key challenge is to identify behavioral factors that remain meaningful across diverse environments. Instead of directly representing environment-specific instances, we abstract observations into generalizable factor types, including object clusters, frontier regions, open-space regions, and boundaries. These factor types provide a shared vocabulary for constructing the initial behavioral basis space. The complete set of behavioral factor types is defined as:

$$\mathcal{E} = \{e^{prev}, \mathcal{E}^{obj}, \mathcal{E}^{front}, \mathcal{E}^{open}, \mathcal{E}^{bound}\}, \tag{4}$$

where e^{prev} represents the previous motion tendency. The object factor subset $\mathcal{E}^{obj}$ captures object-related navigation influences. Specifically, we employ a LLM to cluster objects that are likely to co-occur within similar functional or spatial contexts into semantic groups. Therefore, each object factor corresponds to a group of related objects rather than an individual object, enabling semantic consistency across different environments. The frontier factor subset $\mathcal{E}^{front}$, open-space factor subset $\mathcal{E}^{open}$, and boundary factor subset $\mathcal{E}^{bound}$ describe different spatial navigation conditions. Instead of representing specific spatial instances, these factors are abstracted according to their relative spatial attributes with respect to the agent, including distance (near or far), scale (large or small), and relative position (in front of or behind the agent). More details are provided in the App. G.2.

Given a specific episode, each behavioral factor type is instantiated by the corresponding observations from the current environment. The instantiated factors are then converted into behavioral bases. Each basis represents the behavioral influence induced by a specific factor, consisting of a text component and a spatial directional component related to each $e_i \in \mathcal{E}$ ($i \in \{0, 1, ..., |\mathcal{E}| - 1\}$):

$$f_i = (d_i, c_i), \text{ where } f_i \in \mathcal{F}. \tag{5}$$

Here c_i represents the text description shared by the corresponding factor type, which is generated by a LLM, while d_i represents the directional influence induced by the instantiated factor in the current environment. Therefore, the text description of a basis remains consistent across different episodes, whereas its spatial direction d_i is dynamically determined by the specific environmental realization. For each instantiated factor, its spatial directional component is computed according to the relative position between the agent and the corresponding factor instance. For attractive factors $e_i \in \{\mathcal{E}^{obj}, \mathcal{E}^{front}, \mathcal{E}^{open}\}$, the directional component is defined as $d_i = \frac{p(e_i) - p_t}{\|p(e_i) - p_t\|}$, where $p(e_i)$ denotes the position of the instantiated factor and p_t denotes the current agent position. For $e_i \in \{\mathcal{E}^{bound}\}$, which represent spatial constraints and obstacles, the directional influence is defined as a repulsive direction $d_i = \frac{p_t - p(e_i)}{\|p_t - p(e_i)\|}$. In addtion, we denote the instaniated directional basis matrix as $F = [d_0, ..., d_i, ..., d_{|\mathcal{E}|-1}]$ for the convenience of the subsequent description of the solution.

3.4 Self-state-conditioned Skill Inference

Given the initial behavioral basis space, SelfSkill aims to infer how an agent activates different behavioral factors under a specific self-state. We assume that trajectory fragments generated under the same self-state share a consistent activation pattern over the behavioral basis space (We introduce the details of obtaining trajectory fragments through episode segmentation in App. D).

For the k-th trajectory fragment, the observed motion tendency can be represented as a combination of the corresponding behavioral bases:

$$v_k = F_k \alpha, \text{ where } F_k = F M_k, \tag{6}$$

and F is the instantiated initial directional basis matrix as mentioned above (the bases those are not involved in the current trajectory fragment are set by zero vector), M_k is a binary selection matrix indicating the behavioral bases involved, and α represents the activation pattern of these behavioral factors.

Although the basis space provides an interpretable decomposition of navigation behaviors, directly recovering the activation pattern from motion trajectories is ambiguous. Different behavioral factors may induce similar motion tendencies, resulting in multiple possible activation combinations that explain the same observed behavior. Therefore, motion consistency alone is insufficient to uniquely identify the underlying skill representation.

To resolve this ambiguity, SelfSkill incorporates semantic knowledge from a LLM. Given the current self-state S and the text decription $c = \{c_i\}_i$ of the initial bases in Eq. 5, the LLM estimates a prior over possible activation patterns $P_{\text{LLm}}(\alpha|S, c) = \mathcal{N}(\mu_S, \sigma^2)$, where μ represents the average activation pattern predicted by the LLM, and σ^2 indicates the uncertainty of the prediction. It provides a prior on how different behavioral factors should be activated under the current condition.

The activation inference is formulated by combining behavioral evidence and semantic prior,the activation pattern is obtained through maximum a posteriori inference (The proof is in the App. A.3):

$$\alpha^* = \arg\max_{\alpha} \left[\prod_k P(v_k|F_k, \alpha) \right] P_{\text{LLM}}(\alpha|S, c). \tag{7}$$

Assuming the reconstruction error follows an isotropic Gaussian distribution: $\epsilon_k \sim \mathcal{N}(0, \sigma^2 I)$, the likelihood term is $P(v_k|F_k, \alpha) \propto \exp\left(-\frac{1}{2\sigma^2}\|v_k - F_k\alpha\|_2^2\right)$. Equivalently, considering the sparse composition of embodied skills, we solve:

$$\alpha^* = \arg\min_{0 \le \alpha \le \alpha_{\max}} \sum_k \|v_k - F_k\alpha\|_2^2 - \lambda_1\|\alpha - \mu_S\|_2^2 + \lambda_2\|\alpha\|_1, \tag{8}$$

where the reconstruction term preserves consistency with the observed motion tendency, the semantic prior regularizes the attribution among correlated behavioral factors, and the sparsity constraint encourages a compact activation of relevant factors. The resulting activation pattern provides the skill representation of the agent under the corresponding self-state. The overall optimization procedure of Eq. (8) is summarized in Algorithm 1.

Finally, bases with significant activation values are selected to form the self-state-conditioned behavioral basis space $\mathcal{F}_S$. By inferring skill representations across all self-states, SelfSkill obtains a comprehensive skill profile as Eq. (2) of the policy π, characterizing how the agent utilizes different behavioral factors under diverse self-states.

Algorithm 1 Self-state Conditioned Skill Inference

Require: S, Instantiated $\{(v_k, F_k)\}_{k=1}^K$, μ_S
Ensure: α_S and $\mathcal{F}_S$
1: Initialize $\alpha^{(0)}$
2: **for** $l = 0, \cdots, L-1$ **do**
3: Compute gradient: $g^{(l)} = \sum_k F_k^T(F_k\alpha^{(l)} - v_k) + \lambda_2(\alpha^{(l)} - \mu_S)$
4: Gradient update: $\tilde{\alpha}^{(l+1)} = \alpha^{(l)} - \eta g^{(l)}$
5: Sparsity constraint: $\hat{\alpha}^{(l+1)} = SoftThreshold(\tilde{\alpha}^{(l+1)}, \eta\lambda_1)$
6: Project activation values: $\alpha^{(l+1)} = \Pi_{[0,\alpha_{\max}]}(\hat{\alpha}^{(l+1)})$
7: **end for**
8: Selection: $\alpha_S = \{\alpha_i^* | \alpha_i^* > \delta\}$, $\mathcal{F}_S = \{f_i \in \mathcal{F} | \alpha_i^* > \delta\}$
9: **return** $\alpha_S, \mathcal{F}_S$

3.5 Skill-based Agent Evaluation and Enhancement

The discovered SelfSkill representation provides a fine-grained mechanism for evaluating embodied agents beyond conventional task-level metrics. By associating self-state-conditioned skills with the outcomes of trajectory fragments, SelfSkill evaluates the capability of different skills under different embodied conditions.

3.5.1 Skill-level Evaluation

We evaluate trajectory fragments from a policy π under different self-states using a fragment-level reward. The fragment-level reward is defined as $r_k = R_{success} + R_{slack}$. Here, $R_{success}$ denotes

the task completion reward and R_{slack} introduces a step penalty to encourage efficient navigation, as shown in App. A.4. Based on r_k, we measure the skill of π under a specific self-state S as $R_S^\pi = \frac{1}{K}\sum_{k=1}^{K} r_k$, where K denotes the number of trajectory fragments associated with S.

By evaluating different self-states, we obtain $\mathcal{R}^\pi = \{R_{S_1}^\pi, R_{S_2}^\pi, ..., R_{S_N}^\pi\}$, which reveals how π performs under different self-states rather than only reflecting its overall task-level performance.

Given multiple navigation agents, SelfSkill further reveals complementary functional capabilities among different policies. For a specific self-state S, agents with higher fragment-level rewards indicate more effective utilization of functional skills under this condition.

3.5.2 Skill-based Agent Enhancement

The skill evaluation provides a quantitative understanding of how different agents utilize skills under different self-states. Beyond evaluation, these discovered skill patterns can be further exploited to enhance navigation agents. Specifically, we develop two complementary strategies: (1) skill-aware policy fusion, and (2) skill-guided intervention.

Skill-aware Policy Fusion. Different agents may exhibit complementary strengths across different self-states. Instead of relying on a fixed ensemble strategy, SelfSkill enables adaptive policy fusion by selecting the agent with the most suitable skill activation pattern for the current condition.

For a given self-state S, we obtain the policy with the highest expected reward under the corresponding self-state based on the reward-conditioned skill evaluation $\pi_S^* = \arg\max_\pi R_S^\pi$. During inference, the selected policy generates the next waypoint or frontier for the specific self-state. This skill-aware fusion strategy allows different agents to contribute their strengths under different navigation conditions.

Skill-guided Policy Intervention. Beyond evaluating and analyzing embodied skills, SelfSkill enables skill-level intervention to improve existing navigation policies. Instead of modifying the policy parameters, we exploit the discovered skill activation patterns as an additional behavioral prior to guide the frontier selection process.

Given a self-state S, SelfSkill first extracts the effective skill activation pattern α^* and the state-conditioned functional basis space F_S^* from high-performing agents π^*. Based on them, the corresponding skill-induced navigation tendency is obtained as $d^* = F_S^*\alpha^*$. The obtained tendency d^* represents the preferred motion direction under the current self-state, reflecting how successful agents utilize different environmental factors to make navigation decisions. During inference, for each candidate frontier x, we compute a skill-consistency score according to the alignment between the frontier direction and the learned skill-induced tendency $Score_{\pi^*}(x) = \mathrm{sim}(d^*, d_x)$, where d_x denotes the direction from the current agent state to the candidate frontier x. Meanwhile, the original navigation policy π produces its own frontier preference score $Score_\pi(x)$. We combine the original policy decision and the transferred skill knowledge through a weighted intervention $Score(x) = \alpha Score_\pi(x) + \beta Score_{\pi^*}(x)$, where α and β control the contribution of the original policy and the discovered skill, respectively.

By incorporating skill-induced behavioral tendencies into frontier selection, SelfSkill improves existing navigation agents without additional policy optimization, allowing them to exploit effective behavioral patterns discovered from successful agents.

4 Experiments

We evaluate SelfSkill on two embodied tasks, including ObjectNav and OVMM.

Datasets and Metrics. All details of the part in App. B.1 **Implementation details** All details of the part in App. A.1 **Our Models and Compared Methods.** For ObjectNav, we evaluate four SelfSkill variants based on the two model improvement strategies introduced in Sec. 3.5.2. Specifically, (1) **SelfSkill (VLFM+ApexNav)** and (2) **SelfSkill (VLFM+ApexNav+MSGNav)** are constructed through *Skill-aware Policy Fusion*. In contrast, (3) **SelfSkill (VLFM+BasicSkills)** and (4) **SelfSkill (ApexNav+BasicSkills)** are obtained through *Skill-guided Policy Intervention*. Here, **BasicSKILL** denotes a set of skills derived from the optimal paths collected on the training dataset. We leverage the successful fragments of trajectories to excavate high-performance skill patterns, which subse-

quently serve as transferable behavioral guidance for the existing ObjectNav policies. For OVMM, we evaluate three SelfSkill variants including (1) **SelfSkill (MoTo+3D-IC)** is constructed through *Skill-aware Policy Fusion*. In contrast, (2) **SelfSkill (MoTo+BasicSkills)** and (3) **SelfSkill (3D-IC+BasicSkills)** are obtained through *Skill-guided Policy Intervention*. Detailed descriptions of the remaining comparison methods are provided in App. C.1.

Skill-based Agent Enhancement on ObjectNav. Table 1 presents the performance of SelfSkill for enhancing existing ObjectNav agents through skill-aware policy fusion and skill-guided policy intervention. For skill-aware policy fusion, SelfSkill(VLFM+ApexNav) consistently improves over the original VLFM and ApexNav across different benchmarks, achieving 60.4% SR and 33.2% SPL on HM3D-v1, and 77.2% SR and 38.5% SPL on HM3D-v2. Further incorporating MSGNav skills leads to additional improvements, achieving 78.3% SR on HM3D-v2 and 41.0% SR on MP3D, demonstrating that SelfSkill can effectively identify and combine complementary skills from different navigation policies.

For skill-guided policy intervention, SelfSkill transfers BasicSkill patterns extracted from PointNav trajectories to guide existing ObjectNav policies without modifying their parameters. The proposed intervention achieves substantial improvements, where SelfSkill(ApexNav+BasicSkills) obtains 62.2% SR and 33.7% SPL on HM3D-v1, outperforming the original ApexNav by 2.6% SR and 0.7% SPL. Similar improvements are observed on HM3D-v2 and MP3D, where it achieves 80.3% SR and 43.1% SR, respectively. These results demonstrate that effective skill knowledge discovered from high-performing agents can be transferred to improve existing navigation policies.

Table 1: Comparison with existing ObjectNav methods on HM3D-v1, HM3D-v2 and MP3D. Higher SR and SPL values correspond to better task performance. Here, **BasicSKILLs** denotes navigation skills derived from optimal paths collected on the training dataset.

Method	HM3D-v1		HM3D-v2		MP3D	
	SR(%) ↑	SPL (%) ↑	SR	SPL	SR	SPL
DD-PPO (Wijmans et al., 2019)	27.9	14.2	-	-		
ZSON (Majumdar et al., 2022)	25.5	12.6	-	-	15.3	4.8
ESC (Zhou et al., 2023a)	39.2	22.3	-	-	28.7	14.2
PSL (Sun et al., 2024)	42.4	19.2	-	-	-	-
SGM (Zhang et al., 2024a)	60.2	30.8	-	-	37.7	14.7
VLFM (Yokoyama et al., 2024b)	52.5	30.4	63.6	32.5	36.4	17.5
SG-Nav (Yin et al., 2024a)	54.0	24.9	49.6	25.5	40.2	16.0
VoroNav (Wu et al., 2024a)	42.0	26.0	-	-		
Unigoal (Yin et al., 2025a)	54.5	25.1	-	-	41.0	16.4
BeliefMapNav (Zhou et al., 2025b)	61.4	30.6	-	-	37.3	17.6
InstructNav (Long et al., 2025)	-	-	58.0	20.9	-	-
ApexNav (Zhang et al., 2025a)	59.6	33.0	76.2	38.0	39.2	17.8
MSGNav (Huang et al., 2026)	-	-	74.1	33.4	-	-
SelfSkill (VLFM+ApexNav)	60.4	33.2	77.2	38.5	40.2	18.7
SelfSkill (VLFM+ApexNav+MSGNav)	61.0	33.4	78.3	38.7	41.0	19.4
SelfSkill (VLFM+BasicSkills)	58.7	31.2	73.7	34.4	40.5	18.5
SelfSkill (ApexNav+BasicSkills)	**62.2**	**33.7**	**80.3**	**39.4**	**43.1**	**20.3**

Skill-Based Agent Enhancement on OVMM Table 2 demonstrates the effectiveness of SelfSkill on OVMM. Through skill-aware policy fusion, SelfSkill(MoTo+3D-IC) improves the average SR from 52.7% to 53.5%, showing that complementary skills from different policies can be effectively combined. Moreover, skill-guided policy intervention further transfers BasicSkills extracted from high-performing trajectories to existing OVMM agents. SelfSkill(3D-IC+BasicSkills) achieves the best performance with 55.0% average SR, improving over 3D-IC by 2.3%. The consistent gains across FindObj, Pick, FindRec, and Place demonstrate that SelfSkill provides transferable behavioral knowledge for enhancing mobile manipulation agents.

Table 2: Comparison with OVMM methods on the standard baseline setting as mentioned above. SR across all four stages is evaluated. Higher SR values correspond to better task performance.

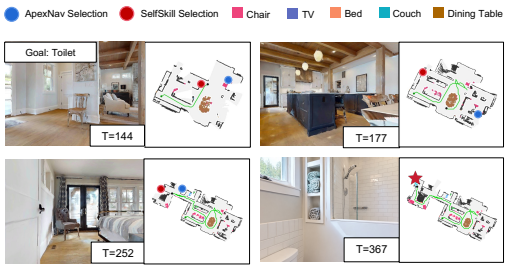
Method	FindObj. SR(%) ↑	Pick SR(%) ↑	FindRec. SR(%) ↑	Place SR(%) ↑	Avg. SR(%) ↑
OVMM(RL)(Yenamandra et al., 2023)	66.6	61.1	50.9	14.8	48.4
OVMM(Hear.)(Yenamandra et al., 2023)	54.1	48.5	31.3	5.1	34.8
UniTeam (Melnik et al., 2023)	66.1	62.7	54.7	18.0	50.4
MoMainpVLA (Wu et al., 2025b)	66.1	62.6	53.1	15.8	49.4
MoTo (Wu et al., 2025c)	66.7	61.0	49.9	20.6	49.6
3D-IC (Zhang et al.)	69.1	64.0	55.6	21.9	52.7
SelfSkill (MoTo+3D-IC)	70.6	65.0	56.1	22.1	53.5
SelfSkill (MoTo+BasicSkills)	69.7	63.5	51.2	22.5	51.7
SelfSkill (3D-IC+BasicSkills)	**72.2**	**66.0**	**57.6**	**24.3**	**55.0**

Table 3: Effect of λ_1 and λ_2 on skill activation inference.

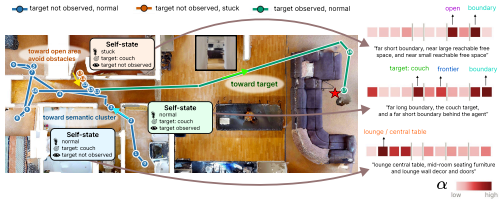
Median Angular Error (°) ↓				
$\lambda_1 \backslash \lambda_2$	0.001	0.01	0.1	0.3
0.01	33.86	32.78	34.10	33.21
0.1	32.62	32.53	33.45	33.18
1.0	**29.59**	30.35	30.13	30.54
10.0	36.12	36.47	36.44	36.79

Table 4: Effect of the intervention weights α and β.

SR / SPL (%) ↑					
$\alpha \backslash \beta$	0.1	0.3	0.5	0.7	0.9
0.9	57.4 / 29.1	58.2 / 29.8	58.8 / 30.4	59.1 / 30.7	58.5 / 30.1
0.7	58.3 / 29.9	59.1 / 30.6	59.8 / 31.4	60.3 / 31.9	59.6 / 31.2
0.5	59.2 / 30.7	60.1 / 31.5	60.9 / 32.3	61.4 / 32.9	60.7 / 32.1
0.3	59.8 / 31.2	60.8 / 32.1	61.6 / 33.0	**62.2 / 33.7**	61.3 / 32.8
0.1	58.9 / 30.4	59.7 / 31.2	60.4 / 31.9	60.9 / 32.4	60.1 / 31.7



(a) Qualitative comparison between ApexNav and SelfSkill (ApexNav+BasicSkills). SelfSkill selects frontiers closer to target-relevant regions, enabling successful navigation in cases where ApexNav fails.



(b) Visualization of self-state conditioned skill activation patterns. The activation values α reflect the importance of different environmental factors under the current self-state.

Figure 3: Qualitative visualization of SelfSkill, including skill-guided policy intervention and self-state conditioned skill activation.

Effect of λ_1 and λ_2. We investigate the influence of λ_1 and λ_2 on skill activation inference using HM3D-v1. The collected episodes are randomly divided into 80% for optimization and 20% for validation. For each configuration, we optimize the activation vector α on the training split and evaluate it on the validation split. We report the median angular error between the motion direction predicted from the inferred skill representation and the ground-truth next-step direction, where lower values indicate better behavioral alignment. As shown in Table 3, varying λ_1 leads to more pronounced performance changes than varying λ_2, indicating a stronger sensitivity to λ_1. The lowest error of $29.59°$ is obtained with $\lambda_1 = 1.0$ and $\lambda_2 = 0.001$.

Effect of Skill Intervention Weights. We study the fusion weights α and β in the Skill-guided Policy Intervention of Sec. 3.5.2 on HM3D-v1. ApexNav is used as the base policy, while a PointNav agent with access to the target location provides high-performing skills. For each candidate frontier x, the original policy score and the transferred skill score are fused as $Score(x) = \alpha Score_{\pi}(x) + \beta Score_{\pi^*}(x)$, where α and β control the contributions of the original policy and skill guidance, respectively. As shown in Table 4, $\alpha = 0.3$ and $\beta = 0.7$ achieve the best performance, with 62.2% SR and 33.7% SPL, compared with 59.6% SR and 33.0% SPL for the original ApexNav. This indicates that appropriately weighted skill guidance can effectively complement the original policy.

Visualization of Self-state Conditioned Skills. Figure 3b shows that SelfSkill discovers self-state conditioned behaviors. Different self-states induce different skill activations, shifting from exploration under target uncertainty to goal-oriented navigation after target observation. Figure 3a further demonstrates that skill-guided intervention enables SelfSkill to select target-relevant frontiers and succeed in cases where ApexNav fails.

5 Conclusion

This work introduces SelfSkill for discovering self-state-conditioned skills from embodied behaviors. SelfSkill represents each skill through a behavioral basis space and its activation pattern, capturing how environmental factors are utilized under different self-states. The discovered skills support capability-level analysis, cross-agent comparison, and policy enhancement beyond task-level evaluation.

AI USE STATEMENT

Generative large language models were used exclusively as supportive tools in this work. They assisted with manuscript language polishing, data processing script debugging, and structural suggestions for figure layout and result organization. No AI tool participated in the design of the navigation methodology, the generation of experimental data, the interpretation of quantitative results, or the formulation of core technical claims. All AI-assisted outputs were manually reviewed, revised, and validated by the authors. The authors take full responsibility for all content and conclusions in the final manuscript.

REFERENCES

Joshua Achiam and Shankar Sastry. Variational option discovery algorithms. In *arXiv*, 2018.

Peter Anderson et al. On evaluation of embodied navigation agents. In *arXiv preprint arXiv:1807.06757*, 2018.

Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *AAAI*, 2017.

Victor Campos et al. Learning intrinsic representations for reinforcement learning. In *ICLR*, 2020.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from RGB-D data in indoor environments. In *International Conference on 3D Vision (3DV)*, 2017.

Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

Junting Chen, Guohao Li, Suryansh Kumar, Bernard Ghanem, and Fisher Yu. How to not train your dragon: Training-free embodied object goal navigation with semantic frontiers. *Proceedings of Robotics: Science and System XIX*, pp. 075, 2023.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *ICLR*, 2019.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation, 2024. URL https://arxiv.org/abs/2401.02117.

Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. 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.

Chen Gao et al. Octonav: Towards generalist embodied navigation. In *CVPR*, 2026.

James J. Gibson. *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979.

Gunshi Gupta, Karmesh Yadav, Yarin Gal, Dhruv Batra, Zsolt Kira, Cong Lu, and Tim GJ Rudner. Pre-trained text-to-image diffusion models are versatile representation learners for control. *Advances in Neural Information Processing Systems*, 37:74182–74210, 2024.

Hao Huang, Yu Hao, Congcong Wen, Anthony Tzes, Yi Fang, et al. Gamap: Zero-shot object goal navigation with multi-scale geometric-affordance guidance. *Advances in Neural Information Processing Systems*, 37:39386–39408, 2024.

Xun Huang, Shijia Zhao, Yunxiang Wang, Xin Lu, Wanfa Zhang, Rongsheng Qu, Weixin Li, Yunhong Wang, and Chenglu Wen. MSGNav: Unleashing the Power of Multi-modal 3D Scene Graph for Zero-Shot Embodied Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37154–37163, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Huang_MSGNav_Unleashing_the_Power_of_Multi-modal_3D_Scene_Graph_for_CVPR_2026_paper.html.

Yanna Jiang, Delong Li, Haiyu Deng, Baihe Ma, Xu Wang, Qin Wang, and Guangsheng Yu. Sok: Agentic skills – beyond tool use in llm agents. *arXiv preprint arXiv:2602.20867*, 2026.

Moo Jin Kim et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

George Konidaris et al. Robot learning from demonstration with skill discovery. In *RSS*, 2018.

Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10965–10975, 2022.

Siyuan Li et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Peiqi Liu, Yaswanth Orru, Jay Vakil, Chris Paxton, Nur Shafiullah, and Lerrel Pinto. Demonstrating ok-robot: What really matters in integrating open-knowledge models for robotics. In *Robotics: Science and Systems XX*, RSS2024. Robotics: Science and Systems Foundation, July 2024. doi: 10.15607/rss.2024.xx.091. URL http://dx.doi.org/10.15607/RSS.2024.XX.091.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. InstructNav: Zero-shot System for Generic Instruction Navigation in Unexplored Environment. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2049–2060. PMLR, 2025. URL https://proceedings.mlr.press/v270/long25b.html.

Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings. In *Advances in Neural Information Processing Systems*, volume 35, pp. 32340–32352. Curran Associates, Inc., 2022. doi: 10.52202/068431-2343. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/d0b8f0c8f79d3a621af945cafb669f4b-Abstract-Conference.html.

Andrew Melnik, Michael Büttner, Leon Harz, Lyon Brown, Gora Chand Nandi, Arjun PS, Gaurav Kumar Yadav, Rahul Kala, and Robert Haschke. Uniteam: Open vocabulary mobile manipulation challenge. *arXiv preprint arXiv:2312.08611*, 2023.

Charles Packer et al. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Alexandros Paraschos et al. Probabilistic movement primitives. In *NeurIPS*, 2013.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M. Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X. Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3D): 1000 large-scale 3d environments for embodied AI. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18890–18900, 2022.

Ram Ramrakhya, Aniruddha Kembhavi, Dhruv Batra, Zsolt Kira, Kuo-Hao Zeng, and Luca Weihs. Seeing the unseen: Visual common sense for semantic placement. In *CVPR*, 2024.

Scott Reed et al. A generalist robot policy. *arXiv preprint arXiv:2310.08864*, 2023.

Stefan Schaal. Dynamic movement primitives - a framework for motor control in humans and humanoid robotics. *Adaptive Motion of Animals and Machines*, 2006.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *ICLR*, 2020a.

Archit Sharma et al. Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. In *ICML*, 2020b.

Xinyu Sun, Lizhao Liu, Hongyan Zhi, Ronghe Qiu, and Junwei Liang. Prioritized Semantic Learning for Zero-shot Instance Navigation. In *Computer Vision – ECCV 2024*, volume 15070 of *Lecture Notes in Computer Science*, pp. 161–178. Springer, 2024. doi: 10.1007/978-3-031-73254-6_10. URL https://link.springer.com/chapter/10.1007/978-3-031-73254-6_10.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 1999.

Alexander Sasha Vezhnevets et al. Feudal networks for hierarchical reinforcement learning. In *ICML*, 2017.

Hongyu Wang, Chuyan Xiong, Ruiping Wang, and Xilin Chen. Bitvla: 1-bit vision-language-action models for robotics manipulation. *arXiv preprint arXiv:2506.07530*, 2025a.

Yiyao Wang, Sixian Zhang, Keming Zhang, Xinhang Song, Songjie Du, and Shuqiang Jiang. TrajRAG: Retrieving Geometric-Semantic Experience for Zero-Shot Object Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15166–15176, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Wang_TrajRAG_Retrieving_Geometric-Semantic_Experience_for_Zero-Shot_Object_Navigation_CVPR_2026_paper.html.

Zihan Wang and Gim Hee Lee. g3d-lf: Generalizable 3d-language feature fields for embodied tasks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14191–14202, 2025.

Zihan Wang, Seungjun Lee, and Gim Hee Lee. Dynam3d: Dynamic layered 3d tokens empower vlm for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 2025b.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

Pengying Wu, Yao Mu, Bingxian Wu, Yi Hou, Ji Ma, Shanghang Zhang, and Chang Liu. VoroNav: Voronoi-based Zero-shot Object Navigation with Large Language Model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53757–53775. PMLR, 2024a. URL https://proceedings.mlr.press/v235/wu24u.html.

Pengying Wu, Yao Mu, Bingxian Wu, Yi Hou, Ji Ma, Shanghang Zhang, and Chang Liu. Voronav: Voronoi-based zero-shot object navigation with large language model. In *International Conference on Machine Learning*, pp. 53757–53775. PMLR, 2024b.

Zhenyu Wu, Yuheng Zhou, Xiuwei Xu, Ziwei Wang, and Haibin Yan. Momanipvla: Transferring vision-language-action models for general mobile manipulation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 1714–1723, 2025a.

Zhenyu Wu, Yuheng Zhou, Xiuwei Xu, Ziwei Wang, and Haibin Yan. Momanipvla: Transferring vision-language-action models for general mobile manipulation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1714–1723. IEEE, 2025b.

Zhenyu Wu et al. Moto: A zero-shot plug-in interaction-aware navigation for general mobile manipulation. In *CoRL*, 2025c.

Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, No @inproceedingschang2017matterport3d, title = Matterport3D: Learning from RGB-D Data in Indoor Environments, author = Chang, Angel and Dai, Angela and Funkhouser, Thomas and Halber, Maciej and Niessner, Matthias and Savva, Manolis and Song, Shuran and Zeng, Andy and Zhang, Yinda, booktitle = International Conference on 3D Vision

(3DV), year = 2017 ah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, Alexander William Clegg, and Devendra Singh Chaplot. Habitat-matterport 3d semantics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4927–4936, 2023.

Sixu Yan, Zeyu Zhang, Muzhi Han, Zaijin Wang, Qi Xie, Zhitian Li, Zhehan Li, Hangxin Liu, Xinggang Wang, and Song-Chun Zhu. M2 diffuser: Diffusion-based trajectory optimization for mobile manipulation in 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–17, 2025. ISSN 1939-3539. doi: 10.1109/tpami.2025.3553454. URL http://dx.doi.org/10.1109/TPAMI.2025.3553454.

Sriram Yenamandra et al. Homerobot: Open-vocabulary mobile manipulation. In *CoRL*, 2023.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. SG-Nav: Online 3D Scene Graph Prompting for LLM-based Zero-shot Object Navigation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 5285–5307. Curran Associates, Inc., 2024a. doi: 10.52202/079017-0171. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/098491b37deebbe6c007e69815729e09-Abstract-Conference.html.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in Neural Information Processing Systems*, 37:5285–5307, 2024b.

Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. UniGoal: Towards Universal Zero-shot Goal-oriented Navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19057–19066, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yin_UniGoal_Towards_Universal_Zero-shot_Goal-oriented_Navigation_CVPR_2025_paper.html.

Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19057–19066, 2025b.

Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 42–48. IEEE, 2024a.

Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 42–48. IEEE, 2024b.

Xinyao Yu, Sixian Zhang, Xinhang Song, Xiaorong Qin, and Shuqiang Jiang. Trajectory diffusion for objectgoal navigation. *Advances in Neural Information Processing Systems*, 37:110388–110411, 2024.

Keming Zhang, Sixian Zhang, Xinhang Song, Hongyu Wang, Yiyao Wang, Yingjie Wang, and Shuqiang Jiang. Joint navigation and manipulation planning with 3d interaction chains. In *Forty-third International Conference on Machine Learning*.

Mingjie Zhang, Yuheng Du, Chengkai Wu, Jinni Zhou, Zhenchao Qi, Jun Ma, and Boyu Zhou. ApexNav: An Adaptive Exploration Strategy for Zero-Shot Object Navigation with Target-centric Semantic Fusion. *IEEE Robotics and Automation Letters*, 10(11):11530–11537, 2025a. doi: 10.1109/LRA.2025.3606388. URL https://doi.org/10.1109/LRA.2025.3606388.

Sixian Zhang, Weijie Li, Xinhang Song, Yubing Bai, and Shuqiang Jiang. Generative meta-adversarial network for unseen object navigation. In *European Conference on Computer Vision*, pp. 301–320. Springer, 2022.

Sixian Zhang, Xinhang Song, Weijie Li, Yubing Bai, Xinyao Yu, and Shuqiang Jiang. Layout-based causal inference for object navigation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pp. 10792–10802. IEEE, 2023.

Sixian Zhang, Xinyao Yu, Xinhang Song, Xiaohan Wang, and Shuqiang Jiang. Imagine before go: Self-supervised generative map for object goal navigation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16414–16425. IEEE, 2024a.

Sixian Zhang, Xinyao Yu, Xinhang Song, Xiaohan Wang, and Shuqiang Jiang. 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, 2024b.

Sixian Zhang, Xinhang Song, Xinyao Yu, Yubing Bai, Xinlong Guo, Weijie Li, and Shuqiang Jiang. Hoz++: Versatile hierarchical object-to-zone graph for object navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Linqing Zhong, Chen Gao, Zihan Ding, Yue Liao, and Si Liu. Topv-nav: Unlocking the top-view spatial reasoning potential of mllm for zero-shot object navigation. *ArXiv*, abs/2411.16425, 2024. URL https://api.semanticscholar.org/CorpusID:274234805.

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. ESC: Exploration with Soft Commonsense Constraints for Zero-shot Object Navigation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 42829–42842. PMLR, 2023a. URL https://proceedings.mlr.press/v202/zhou23r.html.

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International Conference on Machine Learning*, pp. 42829–42842. PMLR, 2023b.

Yifei Zhou, Qianlan Yang, Kaixiang Lin, Min Bai, Xiong Zhou, Yu-Xiong Wang, Sergey Levine, and Erran Li. Proposer-agent-evaluator (pae): Autonomous skill discovery for foundation model internet agents. In *International Conference on Machine Learning*, pp. 79490–79528, 2025a.

Zibo Zhou, Yue Hu, Lingkai Zhang, Zonglin Li, and Siheng Chen. BeliefMapNav: 3D Voxel-Based Belief Map for Zero-Shot Object Navigation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 83008–83036. Curran Associates, Inc., 2025b. doi: 10.52202/085713-2782. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/77b277ef10f16c9d4ecd3b2c0313da0a-Abstract-Conference.html.

Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. 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.

A Method Details

A.1 Implementation Details

In the following, we provide the implementation details of skill activation inference. We take the skill inference process of the ApexNav (Zhang et al., 2025a) agent on the HM3D-v1 benchmark as an example. The LLM prior regularization weight is set to $\lambda_p = 1.0$, and the sparsity regularization weight is set to $\lambda_s = 0.01$. The activation value is constrained within $[0, 1]$ by setting $\alpha_{\max} = 1.0$. The learning rate η is set to 0.0002. The optimization terminates when $\|\alpha^{(l+1)} - \alpha^{(l)}\|_2 < 10^{-6}$ or when the maximum iteration number reaches 50,000. We initialize the activation vector as $\alpha^{(0)} = \mu(S)$. o analyze the effect of initialization, we compare the proposed LLM prior initialization with zero initialization. Across all self-states, prior initialization reduces the accumulated optimization iterations from 282,768 to 173,909, corresponding to a 38.5% reduction.

We utilize GLM-5.3 to generate semantic clusters, employ GPT-5.6-sol to produce prior values (μ), and adopt Qwen2.5-VL-7B for self-state prediction.

A.2 THE PROMPTS OF VLM

We use the prompts as in (Zhang et al.) to obtain specific self-states.

```
CHAIN_SELECTION_PROMPT = """You are an expert in robot manipulation
and navigation. Your task is to analyze the feasibility of each
candidate plan for successfully completing the given instruction.
You will provided with the task instruction and some candidate
interaction plans, where each plan includes a sequence of observations
and the specific actions to be executed at the corresponding locations.

The task instruction: {instruction}
Candidate interaction plans:
{interaction_chains}

For observations corresponding to the "navigate to {target_object}",
consider:
- If the {target_object} is not visible, the likelihood that the
surroundings indicate it is nearby.
- If the {target_object} is visible, proximity to the object
and readiness for grasping.

For observations corresponding to the "pick {target_object}",
consider:
- Visibility and clarity of the {target_object} for grasping.
- Accessibility of the object without occlusion or obstruction.
- Suitability of the current angle and position for reliable grasping.

For observations corresponding to the "navigate to {target_receptacle}",
consider:
- If the {target_receptacle} is not visible, the likelihood that the
surroundings indicate it is nearby.
- If the {target_receptacle} is visible, suitability of the current
position for placement.

For observations corresponding to the "place at {target_receptacle}",
consider:
- Obstacles or clutter that could make placement difficult.
- Stability of the object if placed here.
- Suitability of the current angle and position for placement.

Evaluate the feasibility of each interaction waypoint based on the
above criteria, and then combine these evaluations to determine
an overall score for each plan.

Please provide a concise reasoning summary that compares all
candidate plans. Then assign a raw feasibility score
(a continuous value between 0.0 and 1.0) to each plan. Output
your evaluation strictly in valid JSON format only,
with no extra explanation or natural language outside the JSON
structure.

{{
    "reason": "<Concise explanation comparing the plans and justifying
    the scores>",
    "plan_1": <score_1>,
    "plan_2": <score_2>,
    "plan_3": <score_3>,
    "plan_4": <score_4>,
```

```
    "plan_5": <score_5>,
    "plan_6": <score_6>,
    "answer": <plan_ID>
}}
"""
```

A.3 THE PROOF OF 7

Given a set of observed behavioral evidence and their corresponding instantiated basis matrix,

$$\mathcal{D} = \{(v_k, F_k)\}_{k=1}^{K},$$

where v_k denotes the observed behavior representation and F_k represents the instantiated basis, we infer the skill activation pattern α by maximizing the posterior probability:

$$\alpha^* = \arg\max_{\alpha} P(\alpha|\{(v_k, F_k)\}_{k=1}^{K}, S, c), \tag{9}$$

where S denotes the current self-state and c represents the environment context. Applying Bayes' rule, Eq. 9 can be rewritten as:

$$\alpha^* = \arg\max_{\alpha} P(\{v_k\}_{k=1}^{K}|\alpha, \{F_k\}_{k=1}^{K}) P(\alpha|S, c). \tag{10}$$

The first term represents the likelihood of explaining the observed behaviors with the activated functional bases. Assuming conditional independence among different behavioral observations, the likelihood can be factorized as:

$$P(\{v_k\}_{k=1}^{K}|\alpha, \{F_k\}_{k=1}^{K}) = \prod_{k=1}^{K} P(v_k|F_k, \alpha). \tag{11}$$

The second term denotes the prior distribution over skill activation patterns. Instead of learning this prior from additional optimization, we leverage the semantic knowledge of a large language model conditioned on the agent's self-state and environmental context:

$$P(\alpha|S, c) = P_{\text{LLM}}(\alpha|S, c). \tag{12}$$

Combining the behavioral evidence and semantic prior, the final activation inference objective becomes:

$$\alpha^* = \arg\max_{\alpha} \left[\prod_{k=1}^{K} P(v_k|F_k, \alpha) \right] P_{\text{LLM}}(\alpha|S, c) \tag{13}$$

This formulation enables SelfSkill to infer a interpretable skill activation pattern by jointly considering observed behavioral evidence, behavioral bases, and semantic priors.

A.4 REWARD

$$r_i = R_{success} + R_{slack}, \tag{14}$$

Here, $R_{success}$ denotes the task completion reward as follows, while R_{slack} introduces a step penalty to encourage efficient navigation, and η represents the penalty coefficient.

$$R_{success} = \begin{cases} r_s, & \text{if the goal is reached,} \\ 0, & \text{otherwise,} \end{cases} \quad \text{and } R_{slack} = -\eta \tag{15}$$

B EXPERIMENTAL DETAILS

B.1 DATASETS AND METRICS

(1) **ObjectNav** We evaluate our method on three widely used ObjectNav benchmarks: HM3D-v1 and HM3D-v2, constructed upon the Habitat-Matterport 3D (HM3D) dataset Ramakrishnan et al. (2021) with semantic annotations from HM3D-Semantics Yadav et al. (2023), and Matterport3D (MP3D) Chang et al. (2017). These datasets provide diverse indoor environments with realistic layouts, object distributions, and navigation challenges, enabling comprehensive evaluation of ObjectNav agents.(2) **OVMM** We further evaluate our method on the OVMM benchmark Yenamandra et al. (2023), where each episode consists of four sequential stages: FindObj., Pick, FindRec., and Place. .

ObjectNav **HM3D-v1.** HM3D-v1 is constructed from the Habitat-Matterport 3D (HM3D) dataset and contains high-quality reconstructed indoor environments with diverse spatial structures and object categories. It provides realistic geometric layouts and semantic annotations, making it suitable for evaluating navigation agents under complex indoor scenarios.

HM3D-v2. HM3D-v2 is an updated version of the HM3D benchmark with improved scene quality and expanded semantic annotations. Compared with HM3D-v1, it provides more diverse environments and object configurations, offering a more challenging testbed for assessing the generalization ability of embodied agents.

MP3D. Matterport3D (MP3D) is a widely adopted benchmark for embodied navigation, consisting of large-scale real-world indoor scans collected from diverse building environments. Its complex layouts, long-range navigation paths, and realistic object arrangements make it a challenging benchmark for evaluating navigation efficiency and robustness.

Success Rate (SR). SR measures the task completion rate of an agent, indicating whether the agent successfully reaches the target object within the predefined success criteria. Specifically, SR is defined as: $SR = \frac{1}{N}\sum_{i=1}^{N} S_i$, where N denotes the number of evaluation episodes and $S_i \in \{0, 1\}$ is a binary indicator representing whether the agent successfully completes the i-th episode. A higher SR indicates that the agent is more capable of locating the target object.

Success weighted by Path Length (SPL). While SR only measures task completion, SPL further considers navigation efficiency by penalizing unnecessary exploration and long trajectories. It is defined as: $SPL = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{\max(p_i, l_i)}$, where p_i denotes the actual trajectory length of the agent in the i-th episode, and l_i represents the shortest possible path length from the starting position to the goal. SPL jointly evaluates effectiveness and efficiency: a successful agent following a trajectory close to the optimal path achieves a higher SPL score. Therefore, higher SR and SPL values indicate better navigation performance.

OVMM Different from traditional ObjectNav tasks that only require an agent to locate a target object, OVMM extends embodied intelligence to long-horizon mobile manipulation, where an agent needs to navigate in unseen household environments, identify open-vocabulary objects, grasp the target object, and place it at a specified receptacle. This benchmark integrates multiple embodied capabilities, including visual perception, language grounding, navigation, and object manipulation.

In the OVMM benchmark, each episode consists of four sequential stages: FindObj., Pick, FindRec., and Place. FindObj. evaluates whether the agent can successfully locate the target object, while Pick measures the ability to approach and grasp the object. FindRec. evaluates whether the agent can locate the target receptacle after grasping, and Place measures the final placement success. Following the standard evaluation protocol, we report the Success Rate (SR) of each stage independently, where the success rate of the final Place stage represents the overall task success.

To evaluate execution efficiency, we additionally report the number of interaction steps required by the agent. Lower step counts indicate fewer actions required for task completion. However, this metric may introduce bias because agents that terminate episodes prematurely can obtain fewer steps. Therefore, we additionally adopt the Success weighted by normalized inverse Path Length (SPL) metric Anderson et al. (2018).Following the OVMM evaluation protocol, the shortest path

is estimated using the simulator's ground-truth information and optimal navigation/manipulation planning.

C Detailed Introduction of Comparison Methods

In this section, we provide detailed introductions of the comparison methods used in our experiments. For ObjectNav, we compare with representative navigation approaches spanning reinforcement learning based methods, semantic exploration methods, and recent vision-language-based navigation agents. For OVMM, we include representative mobile manipulation approaches covering reinforcement learning, heuristic planning, and vision-language-action models.

C.1 Compared methods

ObjectNav **DD-PPO.** DD-PPO (Wijmans et al., 2019) is a reinforcement learning based navigation method that adopts decentralized distributed optimization for training embodied agents. It learns navigation policies through large-scale simulation and has become a widely used baseline in embodied navigation benchmarks.

ZSON. ZSON (Majumdar et al., 2022) introduces zero-shot object navigation by leveraging pretrained vision-language representations. It enables navigation towards novel object categories without requiring task-specific training, demonstrating the potential of semantic priors for embodied navigation.

ESC and PSL. ESC (Zhou et al., 2023a) and PSL (Sun et al., 2024) improve object navigation through semantic exploration strategies. These methods utilize high-level semantic information to guide exploration and reduce unnecessary searching in unseen environments.

SGM. SGM (Zhang et al., 2024a) incorporates semantic goal modeling into navigation decision making, improving the ability of agents to reason about the relationship between observations and target objects.

VLFM. VLFM (Yokoyama et al., 2024b) integrates vision-language foundation models into embodied navigation pipelines. By leveraging the semantic reasoning ability of pretrained multimodal models, VLFM provides open-vocabulary navigation capability without task-specific policy training.

SG-Nav. SG-Nav (Yin et al., 2024a) introduces scene graph based reasoning for ObjectNav. It represents environmental semantics through structured relationships among objects and regions, enabling more effective goal-directed exploration.

VoroNav. VoroNav (Wu et al., 2024a) utilizes Voronoi-based spatial representations for exploration and navigation. It improves exploration efficiency by reasoning about free-space structures in complex environments.

Unigoal and BeliefMapNav. Unigoal (Yin et al., 2025b) and BeliefMapNav (Zhou et al., 2025b) focus on goal inference and belief modeling. They maintain semantic representations of possible goal locations to improve decision making under partial observability.

InstructNav and ApexNav. InstructNav Long et al. (2025) and ApexNav Zhang et al. (2025a) explore the use of instruction understanding and large-scale semantic priors for embodied navigation. These approaches improve navigation by integrating language-based reasoning into exploration and planning.

TrajRAG and MSGNav. TrajRAG Wang et al. (2026) retrieves previously observed geometric-semantic experiences to assist navigation in unseen environments. MSGNav Huang et al. (2026) further exploits multimodal semantic guidance for zero-shot object navigation.

OVMM **OVMM(RL) and OVMM(Hear.).** OVMM(RL) and OVMM(Hear.) Yenamandra et al. (2023) are the standard baselines introduced in the HomeRobot OVMM benchmark. OVMM(RL) learns mobile manipulation behaviors through reinforcement learning, while OVMM(Hear.) follows a heuristic-based pipeline for object search and interaction.

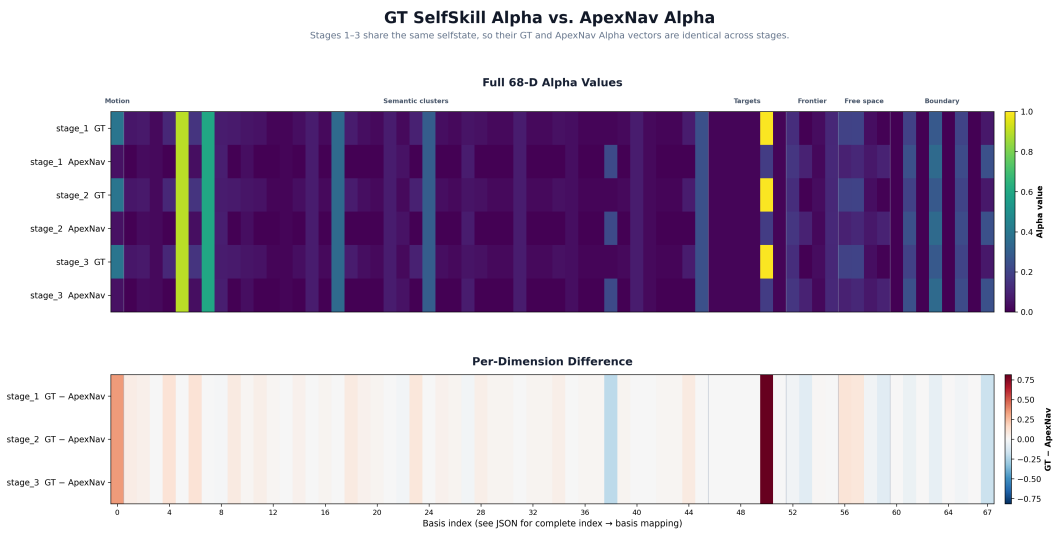


Figure 4: Visualization of the SelfSkill activation patterns of two navigation agents. Each row represents the α vector under a specific self-state, where the dimensions correspond to different functional skill bases. Although the agents share the same self-state across stages, their activation patterns exhibit distinct behavioral preferences. The bottom panel shows the dimension-wise activation differences between the two agents, highlighting the skill activation patterns responsible for their behavioral discrepancies.

UniTeam. UniTeam explores unified embodied intelligence by integrating perception, navigation, and manipulation capabilities within a common framework.

MoManipVLA and MoTo. MoManipVLA Wu et al. (2025b) and MoTo Wu et al. (2025c) are vision-language-action approaches that leverage multimodal foundation models for open-vocabulary mobile manipulation. They aim to connect visual understanding, language instructions, and robot actions for long-horizon tasks.

3D-IC. 3D-IC Zhang et al. incorporates 3D-aware reasoning into embodied manipulation, improving the ability of agents to understand spatial relationships and perform object interaction tasks.

C.2 Visualization of Discovered Skill Activation Patterns

To investigate whether SelfSkill can reveal behavioral differences among navigation agents, we visualize the inferred skill activation patterns of GT and ApexNav under identical self-states. As shown in Fig. 4, each agent is represented by a 68-dimensional activation vector over the shared functional basis space. Although the two agents encounter the same self-state conditions, their activation distributions exhibit clear differences, indicating that they adopt different behavioral compositions to address the same navigation context.

The dimension-wise difference further identifies the specific functional bases responsible for these discrepancies. For example, several semantic and exploration-related bases show consistent activation shifts between the two agents, suggesting that GT and ApexNav differ not only in overall performance but also in how they allocate behavioral preference among different functional skills.

These results demonstrate that SelfSkill provides a model-agnostic representation for analyzing embodied agents, where behavioral variations can be attributed to interpretable skill compositions rather than only task-level outcomes.

C.3 Self-State Conditioned Skill Activation Matrix

Fig. 5 visualizes the relationship between self-states and basis activations. We observe that different self-states induce different activation patterns, demonstrating that skills are not fixed behaviors but emerge from the interaction between agent conditions and environmental factors. For example, the

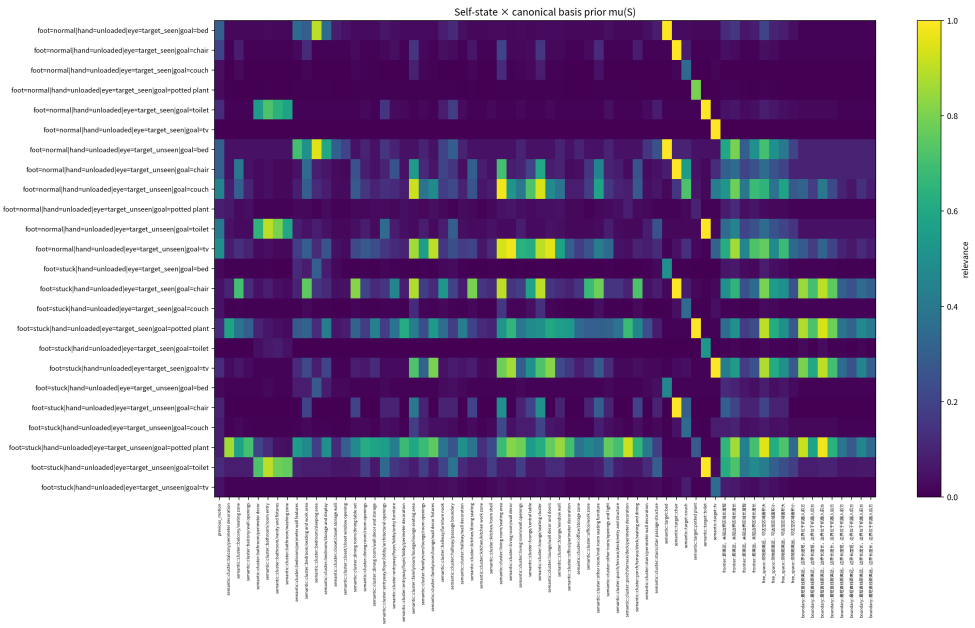


Figure 5: Self-state conditioned skill activation matrix. Each row corresponds to a self-state and each column represents a functional basis. The activation intensity indicates how strongly each basis contributes to the generated behavior under the corresponding self-state.

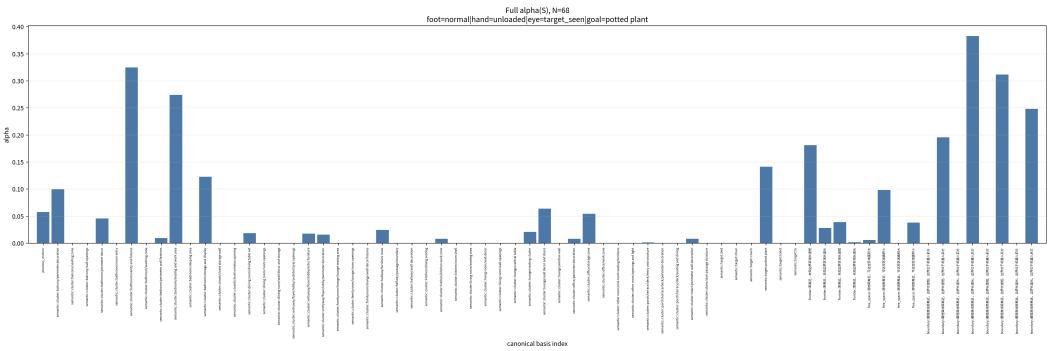


Figure 6: Full Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Seen, Goal-Potted-Plant Condition

same environmental factor may contribute differently when the agent is in different navigation states, highlighting the necessity of modeling skills in a self-state conditioned manner.

C.4 Distribution of skill activation values across functional bases.

Beyond qualitative visualization, we further analyze the contribution of each functional basis through activation distributions (Fig. 5). The results reveal that navigation behaviors are composed of heterogeneous skill factors rather than a single unified strategy. Different agents rely on different subsets of functional bases, which provides a fine-grained perspective for understanding their behavioral strengths and limitations.

Distribution of skill activation values across bases. The results demonstrate that different navigation agents exhibit distinct skill compositions, where specific behavioral factors dominate under different navigation contexts.

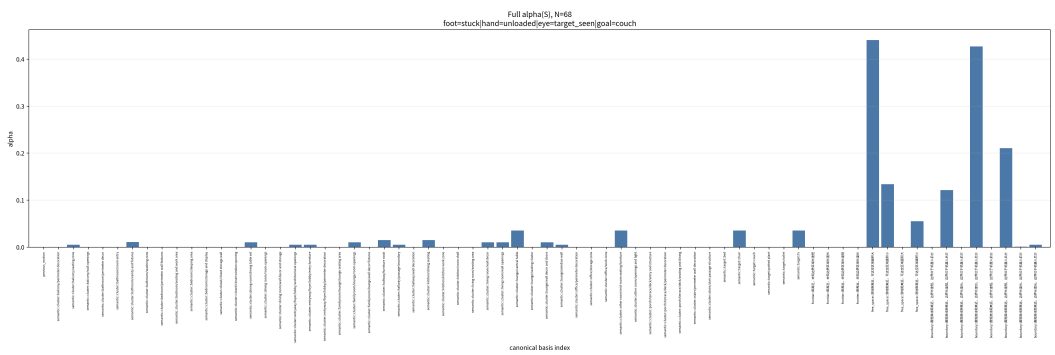


Figure 7: Full Skill Activation Profile ($N = 68$) under Foot-Stuck, Hand-Unloaded, Eye-Target-Seen, Goal-Couch Condition

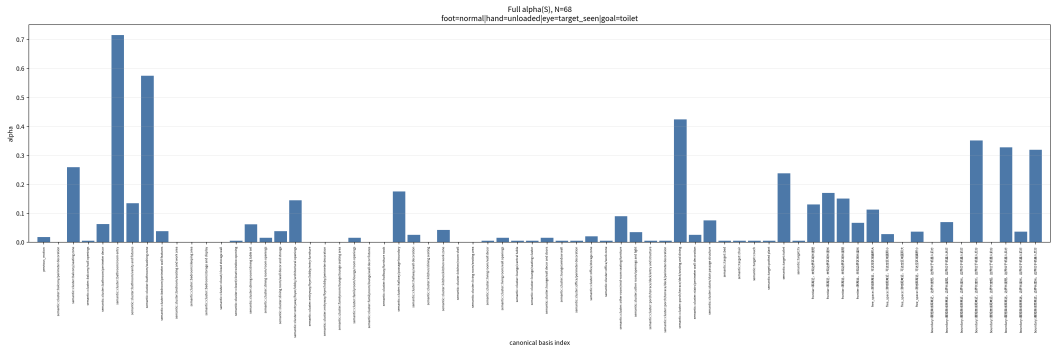


Figure 8: Full Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Seen, Goal-Toilet Condition.

For example, Fig. 7 and Fig. 10 show that, under the same conditions of *Hand-Unloaded*, *Target-Seen*, and a *couch* goal, the model exhibits distinct attention patterns depending on the agent state. When the agent is *stuck*, the model places greater emphasis on boundary-related factors that constrain navigation, whereas under the *normal* state, it focuses more on factors associated with the target object.

A comparison between Fig. 10 and Fig. 11 further reveals that, when the target object *couch* is not observed, the model tends to rely on directions associated with semantic clusters that are semantically related to *couch*. It also integrates multiple cues from the surrounding environment to support exploratory navigation.

A similar pattern can be observed for different target objects. As illustrated in Fig. 12, which shows the top-30 influential factors of the navigation self-state, when the goal is a *Potted Plant*, the agent clearly shifts its preference toward the target direction once the target becomes visible. Meanwhile, when the agent is not *stuck*, the contributions of other factors decrease markedly, suggesting that target-related cues dominate the navigation decision at this stage.

D Episode Segmentation and Trajectory-Graph Construction

Instead of storing every action step across the full navigation trajectory, we extract a sparse sequence of event-driven skeleton points. A skeleton point represents a critical decision moment along the path, which preserves the complete navigation state and all object observations accumulated up to that point. A skeleton point is recorded when at least one of the following conditions is satisfied:

1. the episode starts or ends;
2. the accumulated heading change reaches 35 degrees;

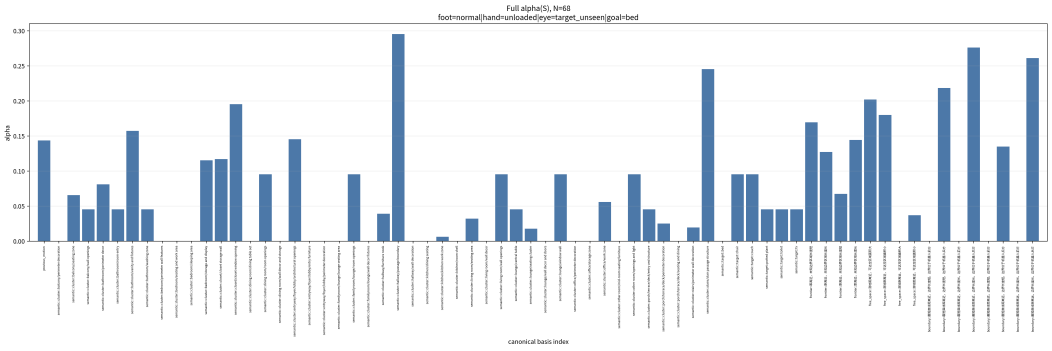


Figure 9: Full Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Unseen, Goal-Bed Condition.

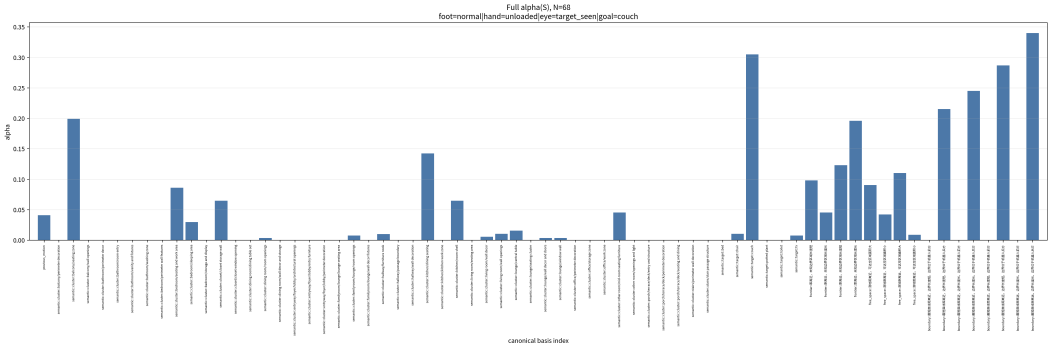


Figure 10: Full Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Seen, Goal-Couch Condition.

3. the target category is observed for the first time;
4. the agent enters or escapes a stuck state;
5. a consecutive in-place rotation finishes.

Ordinary heading-triggered points must be separated by at least two action steps, whereas target, stuck, and boundary events bypass this temporal constraint. Object observations collected between two skeleton points are buffered and attached to the next retained point.

The recorded skeleton sequence is converted into a *directed trajectory graph*. Candidate graph nodes include episode boundaries, translated turns, room transitions, stuck and recovery events, the first target observation, and the first observation matched to the successful target instance. A heading change becomes a translated turn only if the displacement from the preceding graph anchor is at least 0.45 m. Consecutive in-place rotations are represented by their final rotation state, while their complete observation interval is retained.

Short graph fragments are then merged iteratively:

- Adjacent nodes separated by less than 0.45 m are merged unless both represent *hard events*. Hard events include episode boundaries, room transitions, stuck or recovery events, and target-observation events.
- If two nearby turn nodes compete, the node with the larger heading change is retained.
- An ordinary node preceding an event or turn is absorbed into that node; otherwise, it is absorbed into the preceding node.
- An event node and its immediately following turn are preserved as separate nodes.
- The retained node uses the final representative pose and the union of observations from the absorbed interval.

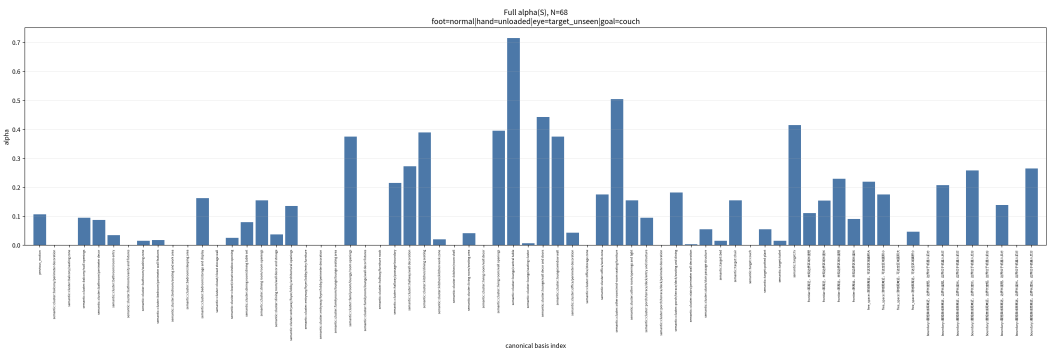


Figure 11: Full Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Unseen, Goal-Couch Condition.

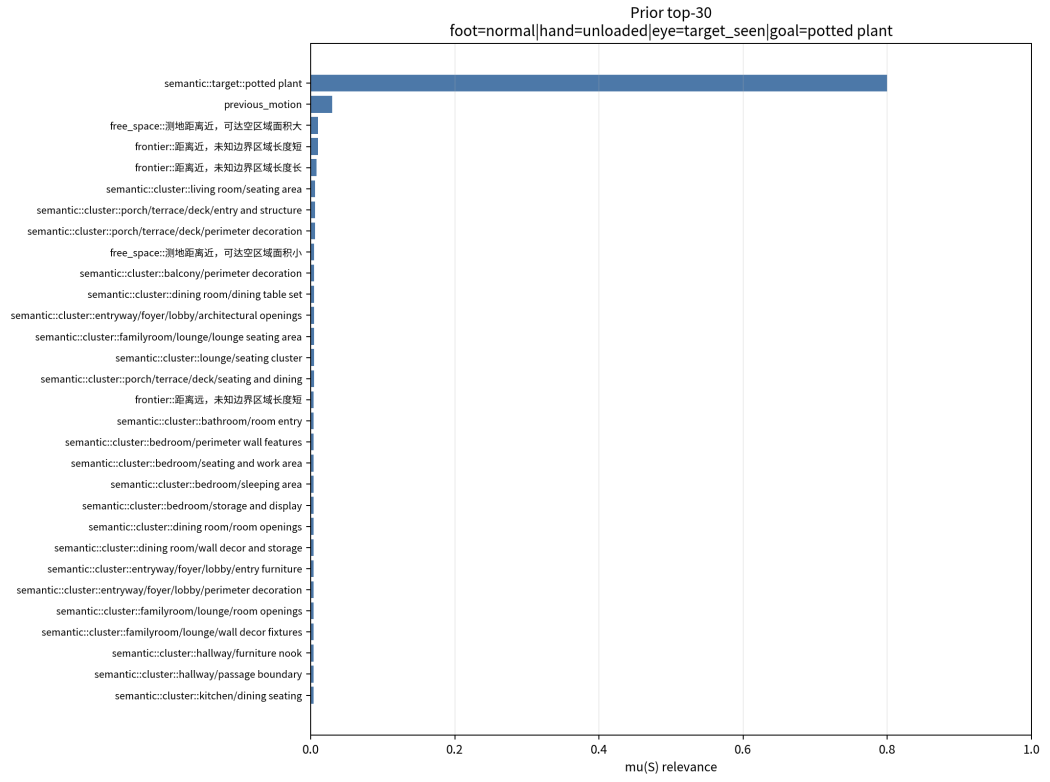


Figure 12: Top-30 Skill Activation Profile ($N = 68$) under Foot-Normal, Hand-Unloaded, Eye-Target-Seen, Goal-Potted Plant Condition.

The resulting graph is a temporally ordered graph:

$$G = (V, E), \quad E = \{(v_i, v_{i+1})\}_{i=1}^{|V|-1} \tag{16}$$

Each node stores the navigation state and observations at a decision point. Each edge stores the intervening trajectory segment, including its path length, heading change, goal-distance change, planner states, and aggregated spatial evidence.

E Semantic-Cluster Construction

Semantic clusters are generated offline for each room type using ground-truth Matterport3D object inventories. Room types represented by fewer than ten rooms are discarded, and object classes with

fewer than twenty instances are excluded. For each remaining class, we compute its instance count, room-occurrence frequency, and median distance to the nearest wall. For each sufficiently frequent class pair, we additionally compute the number of rooms containing both classes and their median within-room centroid distance.

An LLM partitions the retained classes of each room type into three to five navigation-relevant clusters. The partition jointly considers functional compatibility, empirical co-location, and wall-distance tendencies. Incoherent classes are omitted, singleton clusters are allowed only for dominant room-defining objects, and room types without a reliable structure may be skipped. Each cluster is represented by an area label, an object-class set, and a textual region prototype. Clusters are room-specific; therefore, the same object class may belong to different clusters in different room types.

F Extraction of Direction Bases

All direction bases are expressed in the agent body frame, where $+x$ denotes forward and $+y$ denotes left. For a candidate center $\boldsymbol{c}$, agent position $\boldsymbol{p}$, and heading θ, the direction is

$$f(\boldsymbol{c}) = \boldsymbol{R}(-\theta)\frac{\boldsymbol{c}-\boldsymbol{p}}{\|\boldsymbol{c}-\boldsymbol{p}\|_2} \tag{17}$$

where $\boldsymbol{R}(\theta)$ is the 2D rotation matrix and $\|\cdot\|_2$ denotes the Euclidean (L_2) norm.

The final representation contains the target category and 45 semantic-cluster categories, together with four categories for each geometric basis. Each category retains at most six directions; missing slots are represented by zero vectors.

F.1 Semantic basis

Object detections newly introduced within the incoming node interval are grouped by class. Repeated detections of the same class within $0.5\,\mathrm{m}$ are merged by averaging their positions and retaining the maximum detector confidence and goal-similarity score. Target detections with confidence at least 0.3 are grouped into spatial components using a $1.5\,\mathrm{m}$ radius. Each component produces a direction toward its centroid. Non-target objects are assigned to one fixed semantic cluster using class membership, agreement with wall or mid-room geometry, and co-occurrence with other observed classes. Instances assigned to the same cluster are spatially grouped, and each group centroid defines one semantic direction. Goal observations are prioritized; otherwise, semantic clusters are ranked by their strongest goal-similarity evidence, with instance count used to break ties.

F.2 Frontier basis

A frontier cell is an unknown occupancy-grid cell with at least one free four-neighbor. Frontier cells are grouped using eight-connectivity and associated with the nearest active planner frontier within $0.75\,\mathrm{m}$. For each planner frontier, only the connected component closest to its center is retained. Its centroid defines the direction, and its size is measured as the number of cells multiplied by the grid resolution. Candidates are ordered by increasing Euclidean distance and decreasing frontier length.

F.3 Free-space basis

The free-cell mask is morphologically opened with a $0.3\,\mathrm{m}$ disk to remove narrow or noisy regions. Cells within $0.6\,\mathrm{m}$ of the agent are excluded, and connected regions smaller than $0.1\,\mathrm{m}^2$ are removed. Reachability is computed on the original free grid using eight-connected geodesic search without diagonal corner cutting. Each reachable region produces a direction toward its centroid and is described by its geodesic distance and reachable area. Candidates are ordered by increasing geodesic distance and decreasing area.

F.4 Boundary basis

An occupied boundary cell is an occupied cell with at least one non-occupied four-neighbor. Boundary cells are grouped using eight-connectivity. Each component produces a direction toward its centroid, while its distance is the minimum Euclidean distance from the agent to any component cell

and its length is the number of cells multiplied by the grid resolution. Candidates are ordered by increasing distance and decreasing boundary length.

For frontier, free-space, and boundary bases, near/far and short/long or small/large labels are obtained by comparing each measurement with its dataset-level mean. This yields four fixed categories per basis: near-short, near-long, far-short, and far-long for frontier and boundary; and near-small, near-large, far-small, and far-large for free space.

G SEMANTIC-CLUSTER GENERATION PROMPT

The following prompt is used to generate semantic clusters with an LLM, conditioned on scene statistics and clustering constraints.

```
[fontsize=\small]
You are grouping object categories into navigation-relevant semantic clusters
from ground-truth scene statistics (Matterport3D train scenes,
human-annotated furniture inventories).

Room type: "{room}". Statistics aggregated from {n_rooms} {room} rooms
across {n_scenes} scenes.

Evidence 1 -- object categories found in these rooms. For each:
instance count, share of rooms containing it, and median distance (m)
from the object to the nearest wall (small = hugs the perimeter/corners,
large = stands in the open middle):
{classes}

Evidence 2 -- strongest co-location pairs: number of rooms containing both
categories, and the median distance (m) between the two categories' centroids
inside a room (small = physically clustered together):
{pairs}

Task: partition the categories into a SMALL number (3-5) of semantic clusters.
A cluster is a set of categories that share ONE navigation meaning in a {room},
e.g. "food-preparation appliances", "seating furniture", "perimeter decoration".
Merge freely; do not fragment.

Rules:
- Use BOTH your prior knowledge of how {room}s are furnished AND the evidence:
  frequent co-location and small centroid distance are strong support;
  your prior can group categories even without observed pairs.
- Use the wall-distance signal to separate perimeter clusters (curtains, plants,
  pictures) from mid-room clusters (tables, seating).
- DROP categories that fit no coherent cluster. Never create a singleton
  cluster just to rescue a category. A singleton is allowed ONLY for a
  high-count category that anchors the room by itself.
- For each cluster write a short "area" tag (2-5 words, your own taxonomy)
  and ONE English sentence describing it as a navigation-relevant region
  prototype naming the concrete objects and the room type.
  Plain factual style, no invented objects.
- Be conservative: if the evidence is unstructured and no 3 coherent
  clusters emerge, skip this room type.

Return ONLY JSON, no explanations:
{"skip": false, "clusters": [{"area": "...", "objects": ["a", "b"], "text_prototype"
or
{"skip": true, "reason": "one line"}
```

G.1 GT-SCENE SEMANTIC CLUSTERS (ROOM-TYPE LEVEL)

Below we present the complete set of room-type-level semantic clusters derived from the GT scene dataset (55 training scenes with ground-truth annotated object inventories). Each room type is partitioned into 3–4 navigation-relevant clusters, each defined by an area tag, an object set, and a textual region prototype description.

G.2 GT-SCENE SEMANTIC CLUSTERS (ROOM-TYPE LEVEL)

Below we present representative room-type-level semantic clusters derived from the GT scene dataset (55 training scenes with ground-truth annotated object inventories). Each room type is partitioned into 3–4 navigation-relevant clusters, each defined by an area tag, an object set, and a textual region prototype description. Core room types are shown below; additional room types follow the same partitioning scheme.

Bathroom (46 rooms)

- **vanity and fixtures**: sink, mirror, cabinet, counter, toilet. The wall-hugging fixture wall of a bathroom containing the sink, mirror, cabinet, counter, and toilet, which co-locate tightly near the perimeter.
- **washing zone**: shower, bathtub, curtain, towel. The bathing region of the bathroom where the shower, bathtub, curtain, and towels cluster together for washing.
- **perimeter decor**: picture, window, lighting, plant, shelving, table. Decorative and storage elements placed along the bathroom walls, including pictures, windows, lighting, plants, shelving, and small tables.
- **room entry**: door. The doorway anchoring the bathroom, found flush to the wall in nearly every room.

Bedroom (44 rooms)

- **sleeping area**: bed, cushion, chest of drawers, lighting. In a bedroom, the bed with its cushions stands against the wall, flanked by a chest of drawers and nearby lighting, forming the central sleep region the agent navigates around.
- **perimeter wall features**: window, curtain, picture, mirror. Wall-mounted features such as windows with curtains, pictures, and mirrors line the bedroom perimeter and are recognizable landmarks but not traversable regions.
- **seating and work area**: chair, table, stool, sofa, seating. Chairs, stools, sofas, and tables placed near the bed or wall form the bedroom's seating and small work or lounging area.
- **storage and display**: cabinet, shelving, plant, tv monitor. Cabinets, shelving, and a tv monitor against the bedroom walls, often with a plant beside them, mark storage and entertainment corners used for orientation.

Hallway (44 rooms)

- **passage boundary**: door, column. In hallways, doors and columns mark the passage structure, with columns standing near doors along the circulation path.
- **wall decoration**: picture, window, lighting. In hallways, pictures, windows, and light fixtures hug the perimeter as wall-mounted visual landmarks for navigation.
- **furniture nook**: chair, table, cabinet, plant. In hallways, chairs, tables, cabinets, and potted plants form small furnished nooks along the corridor.

Kitchen (43 rooms)

- **kitchen work zone**: cabinet, counter, sink, appliances. The perimeter food-preparation strip of the kitchen where cabinets, counters, sink, and appliances line the walls in tight clusters.

- **dining seating**: table, chair, stool. A mid-room dining area in the kitchen where a table is surrounded by chairs and stools, with chair and table centroids nearly overlapping.
- **room shell**: door, window, shelving. The kitchen's openings and storage fixtures, with doors and windows on the perimeter and freestanding shelving nearby, marking room boundaries.

Living Room (40 rooms)

- **seating area**: sofa, chair, cushion, table. The central seating area of the living room where sofas and chairs with cushions cluster tightly around a coffee table.
- **wall openings**: window, door, curtain. Perimeter openings of the living room consisting of windows and doors with curtains hung along the walls.
- **wall decor**: picture, lighting, plant, fireplace. Decorative features mounted or placed along the living room walls, including pictures, lamps, plants, and the fireplace.

Additional room types Dining room, entryway/foyer/lobby, office, family room/lounge, closet, porch/terrace/deck, stairs, lounge, other room, and balcony are partitioned following the same methodology.