Breaking Down and Building Up: Mixture of Skill-Based Vision-and-Language Navigation Agents

Anonymous ACL submission

Abstract

Vision-and-Language Navigation (VLN) poses significant challenges for agents to interpret natural language instructions and navigate complex 3D environments. While recent progress has been driven by large-scale pre-training and data augmentation, current methods still struggle to generalize to unseen scenarios, particularly when complex spatial and temporal reasoning is required. In this work, we propose SkillNav, a modular framework that introduces structured, skill-based reasoning into Transformer-based VLN agents. Our method decomposes navigation into a set of interpretable atomic skills (*e.g.*, Vertical Movement, Area and Region Identification, Stop and Pause), each handled by a specialized agent. To support targeted skill training without manual data annotation, we construct a synthetic dataset pipeline that generates diverse, linguistically natural, skill-specific instruction-trajectory pairs. We then introduce a novel training-free Vision-Language Model (VLM)-based router, which dynamically selects the most suitable agent at each time step by aligning sub-goals with visual observations and previous actions. SkillNav obtains competitive results on commonly used benchmarks and establishes state-of-the-art generalization to the GSA-R2R, a benchmark with novel instruction styles and unseen environments.

1 Introduction

Vision-and-Language Navigation (VLN) (Anderson et al., 2018; Zhang et al., 2024c) is a subfield of embodied AI that integrates natural language understanding, visual perception, and sequential decision-making to allow autonomous agents to navigate and interact within visual environments. With the rise of foundation models (Zhou et al., 2024a; Xiao and Zhu, 2025; Li et al., 2024; Zhang et al., 2024a), VLN has seen notable progress in multimodal grounding and generalization.

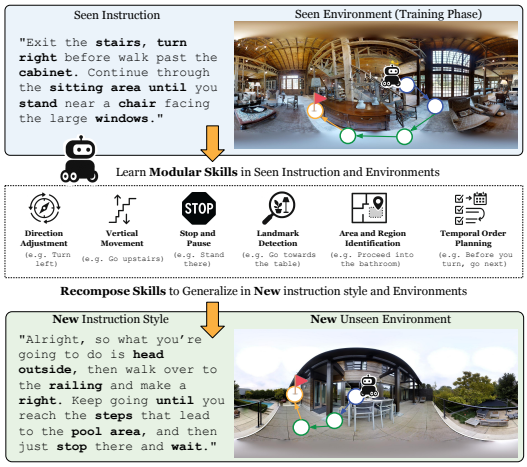


Figure 1: SkillNav decomposes complex navigation instructions into atomic skills, which can be flexibly recomposed to address new environments.

Despite recent advances, a key challenge in VLN lies in enabling agents to generalize reliably and interact with unseen environments and novel instructions. Previous approaches have enhanced VLN agents' generalization ability through extensive training on large-scale synthetic instruction-trajectory pairs across varied environments (Hao et al., 2020; Chen et al., 2022a; Wang et al., 2023, 2024b). While data-driven methods improve VLN agents' generalization, their main limitation is reliance on black-box, end-to-end models (Anderson et al., 2018; Hong et al., 2021) that tend to memorize training examples. This restricts their effectiveness in unobserved scenarios requiring deeper compositional reasoning, such as understanding diverse instructions, temporal relationships, or complex landmarks, and generalizing across a wide range of visual environments. Beyond data-intensive supervised approaches, recent work has explored zero-shot approaches leveraging Large Language Models (LLMs) for VLN tasks to improve generalization ability (Zhou et al., 2023; Long et al., 2024; Chen et al., 2024a; Zhang et al., 2025a). Although zero-shot LLM-based agents show relatively stable performance across seen and unseen environments,

they still considerably lag behind fine-tuned VLN models. Specifically, we observe a significant performance gap ($\sim 36\%$ in Success Rate), primarily arising from intrinsic limitations of LLMs, including their insufficient grounding in embodied environments and imprecise alignment of linguistic instructions with specific navigational actions. This gap highlights the urgent need for methods capable of bridging the broad generalization and compositional reasoning of LLMs with precise, visually grounded execution.

To address these limitations, we propose **SkillNav**, a modular VLN framework that decomposes navigation learning into individual and reusable skills, enabling flexible re-composition and enhanced generalization in new environments as shown in Figure 1. Unlike prior methods that treat instruction execution as an end-to-end mapping from instructions directly to actions, SkillNav explicitly captures the compositional nature of navigation tasks. Furthermore, we introduce a novel Vision-Language Model (VLM)-based router that leverages multi-modal reasoning to dynamically select the most appropriate skill at each navigation step, conditioned on the current sub-instruction, visual observation, and historical actions. SkillNav not only improves interpretability by making the decision-making processes more transparent but also facilitates robust adaptation to diverse instructions and unseen visual environments.

Specifically, we build on previous research (Wang et al., 2024a), and identify a set of atomic skills required for effectively completing the VLN task. For each skill, we construct a dataset containing relevant instructions paired with corresponding visual observations, and fine-tune a dedicated agent on top of a strong VLN backbone. This process yields five specialized skill agents, covering Direction Adjustment, Vertical Movement, Stop and Pause, Landmark Detection, and Area and Region Identification, each proficient in its designated capability. To orchestrate these specialized agents for complex navigation tasks, we introduce a VLM-based Action Router that operates in three distinct phases. First, a Temporal Reordering Module processes the natural language instruction to generate a chronologically ordered sequence of subgoals, resolving implicit temporal dependencies. Next, a Subgoal Localizer dynamically identifies the immediate objective by aligning the reordered plan with the agent's visual and action history. Finally, a Skill Router analyzes the active subgoal and context to select the most suitable expert agent to execute the next navigation action.

SkillNav attains a strong performance on the Room-to-Room (R2R) benchmark (Anderson et al., 2018), and achieves state-of-the-art (SOTA) generalization on the GSA-R2R benchmark (Hong et al., 2025). GSA-R2R contains novel dialogue-style instructions and diverse visual environments with both unseen residential and non-residential settings. Additionally, we evaluate individual skill-based agents using NavNuances (Wang et al., 2024a), a dataset specifically designed for fine-grained skill evaluation. We provide a comprehensive ablation study and qualitative analysis to thoroughly assess the effectiveness of each component within our framework and justify our router design choices. Our contributions are summarized as follows:

1. We propose **SkillNav**, a modular framework that decomposes navigation into atomic skills to combine VLM reasoning with specialized fine-tuned agents, significantly enhancing generalization to novel instructions and visual environments.
2. We design a synthetic data generation pipeline that creates skill-specific but linguistically diverse supervision without requiring human annotation.
3. We demonstrate state-of-the-art generalization on the challenging GSA-R2R dataset and validate our framework through a comprehensive ablation study.

2 Related Work

Vision-and-Language Navigation Models. A wide range of methods have been proposed for addressing VLN tasks. These methods have evolved from early LSTM-based architectures (Anderson et al., 2018; Tan et al., 2019) to Transformer-based models (Chen et al., 2021, 2022b; An et al., 2023) and, most recently, to Large Language Model (LLM)-based agents (Zhou et al., 2023; Chen et al., 2024a; Lin et al., 2024; Zhou et al., 2024b; Zheng et al., 2024; Zhang et al., 2025b,c). A primary challenge remains improving generalization to unfamiliar environments. Existing methods largely rely on data augmentation strategies for visual observations (Li et al., 2022; Liu et al., 2021; Li and Bansal, 2023) or instructions (Wang et al., 2023, 2024b; Hao et al., 2020; Zhang and Kordjamshidi, 2023; Zhang et al., 2024b). However, these end-to-

end supervised approaches often memorize training trajectories rather than mastering the compositional reasoning required in novel or unseen scenarios. While recent zero-shot LLM agents (Zhou et al., 2023; Chen et al., 2024a; Long et al., 2024; Zhang et al., 2025a) avoid task-specific training, they lack of detailed spatial understanding and precise grounding in action execution. In contrast, we propose SkillNav, a modular framework that explicitly decomposes VLN tasks into reusable navigation skills. Each skill is individually fine-tuned for precise spatial grounding, while high-level reasoning and flexible skill composition leverage LLMs and VLMs, significantly improving generalization to unseen environments and varied instructions.

Multi-Agent Collaboration in Vision-Language Navigation. Recent multi-agent VLN work has moved toward debate-style systems like DiscussNav (Long et al., 2024) and planner executor hierarchies like FlexVLN (Zhang et al., 2025a) and CLASH (Wang et al., 2025). Although these approaches can generalize better, they typically rely on ensembles that activate multiple models per step, incurring structural redundancy. When these generalist ensembles conflict, systems often default to ungrounded zero-shot LLM decisions, sacrificing in-domain precision for broader adaptability. SkillNav overcomes this dilemma by decomposing navigation into a library of specialized skill agents. A VLM router selects one best-fit specialist for the current sub-goal instead of invoking an ensemble. This enables the VLM to perform compositional planning by decomposing complex instructions into manageable sub-goals and mapping them to specific skills, while specialized agents ensure precise, grounded action execution. Consequently, SkillNav achieves state-of-the-art generalization on unfamiliar benchmarks like GSA-R2R while remaining competitive on in-domain datasets such as R2R, effectively balancing computational efficiency with robust performance.

3 Preliminaries

In the VLN problem setting, an agent navigates through an environment by following a natural language instruction I to reach a specified target location. The environment is discretized into a connectivity graph $\mathcal{G} = (V, E)$, where V denotes a non-empty set of navigable nodes, and E is a set of undirected connectivity edges. At each time step t, the agent located at viewpoint v_t receives a panorama represented by n images, denoted as $D_t = \{o_i\}_{i=0}^{n}$. The agent is aware of a subset of views $O_t \subseteq D_t$ heading towards its navigable neighboring nodes $\mathcal{N}(v_t)$. The local action space A_t contains navigating to node $v \in \mathcal{N}(v_t)$ or stopping at current node v_t.

In this work, we leverage DUET (Chen et al., 2022b) as our base VLN agent. DUET is a graph transformer-based architecture designed to balance long-term planning on a topological map with fine-grained visual grounding of the current panoramic view. To integrate this architecture into our framework, we formulate its policy as:

$$a_t^* = \pi(I, O_t, M_t). \tag{1}$$

where $M_t \subseteq \mathcal{G}$ denotes the online constructed topological map observed after t steps of navigation, and $a_t^* \in A_t$ is the predicted action.

4 Methodology

We propose **SkillNav**, a framework that coordinates a set of agents, each trained for a specific atomic skill, to solve the navigation problem. SkillNav enhances generalization by treating navigation as a composition of atomic skills rather than a direct language-to-action mapping. This design mirrors how humans transfer sub-skills across unfamiliar situations, preventing overfitting to complex trajectories and enabling systematic reuse of skills across environments and instruction styles. As shown in Figure 2, the framework comprises two key components: a VLM-based router that performs temporal reordering, subgoal localization, and skill selection, and a set of skill-specific agents. Each agent is built upon the DUET architecture and trained with tailored synthetic data to make skill-conditioned decisions, as detailed in this section.

4.1 Skill Taxonomy

We use the skills defined in NavNuances (Wang et al., 2024a) that appear to be essential for building a robust VLN agent. NavNuances provides skill categories and creates a diagnostic dataset to analyze models' errors. However, it does not provide solutions for improving the agent's skills. In this work, we extend the initially proposed skill categories and provide solutions for acquiring them by skill-based agents.

We adopt four frequently observed atomic skills from NavNuances, **Direction Adjustment**, **Vertical Movement**, **Landmark Detection**, and **Area**

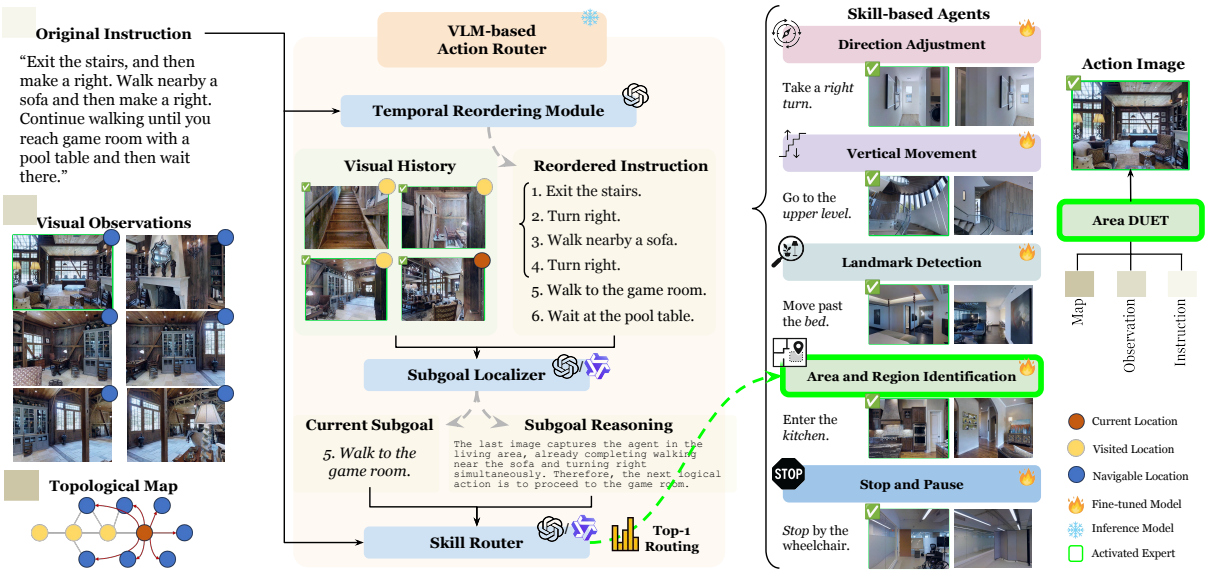


Figure 2: **SkillNav architecture.** Inputs are visual observations, the original instruction, and the online topological map. The VLM-based action router (i) applies temporal reordering to produce an ordered list of subgoals (used solely to assist subgoal localization), (ii) localizes the next subgoal using visual and subgoal history based on the reordered instruction, and (iii) routes to the most suitable skill-based agent. The chosen agent conditions on the original instruction, current observations, and map to output the navigation action.

and Region Identification. Detailed NavNuances skill definitions and annotation cues are provided in Appendix A.1. Moreover, we find persistent challenges in temporal reasoning and stop criteria. Errors in temporal reasoning often disrupt the correct order of subgoal execution. Critical stop decisions are sometimes made too early or too late, reducing navigation success. To address these issues, we extend the skill taxonomy with two additional skills: **Stop and Pause** and **Temporal Order Planning**. In the following, we elaborate on these two new skills and their roles in navigation.

Stop and Pause captures the agent's ability to dynamically control motion termination and temporary halting in response to visual or linguistic cues. This includes recognizing explicit stop commands (*e.g.*, "Stop at the doorway") or context-sensitive halts triggered by landmarks or obstacles (*e.g.*, "Pause when you see the red sign"). The stop and pause skill emphasizes precise temporal-spatial control to ensure safe, context-aware navigation.

Temporal Order Planning reflects the agent's capability to reason over the sequence and structure of subgoals. This includes understanding conditional immediacy (*e.g.*, "Once you enter the hallway, turn left"), maintaining actions for a bounded duration (*e.g.*, "Keep walking until you see the staircase"), executing forward sequential steps (*e.g.*, "Go forward, then turn right, and finally stop"), and handling backward references to prior states (*e.g.*, "Before turning, make sure you're at the hallway entrance"). Effective temporal order planning involves temporal relations that guide both when and how atomic skills should be executed.

To quantify the presence and frequency of these skills in R2R (Anderson et al., 2018), we perform a keyword-based analysis of the navigation instructions as shown in Figure 4 in Appendix B. Each instruction is scanned for a curated set of indicative keywords, compiled for each skill category based on linguistic patterns observed in prior datasets and real-world navigation discourse. For instance, terms like "wait" or "stay" are used to detect Stop and Pause, while words such as "stairs" or "elevator" signal Vertical Movement. An instruction can be counted for multiple skills if it exhibits multiple relevant keywords.

4.2 Skill-Specific Data Synthesis and Agent Training

We build six synthetic datasets where each trajectory–instruction pair targets a single skill. Paths are sampled from random Matterport3D (Chang et al., 2017) nodes, filtered with skill-specific heuristics (*e.g.*, frequent turns for Direction Adjustment), and kept short (4–7 steps) for comparable difficulty; Appendix B details the filters and Figure 5 in Appendix C shows path length analysis. For instruc-

Table 1: Statistics of core VLN training/augmentation corpora (top) and our skill-specific synthetic datasets (bottom).

Dataset	# Instr	# Vocab	Instr Len
R2R	14,039	4,597	26.28
R2R (ScaleVLN-Aug)	2,891,134	222	25.90
R2R (SRDF-Aug)	413,999	2,201	23.67
GSA-R2R	4,675	2,797	26.06
Temporal	2,000	1,653	56.60
Direction	450	707	26.78
Vertical	450	705	26.23
Stop	450	774	27.03
Landmark	450	1,025	27.62
Region	450	971	27.50

tion generation, we feed each trajectory's observations to GPT-4o (OpenAI, 2024) with skill cues so the text stays fluent and R2R-like while emphasizing the target skill (Table 1); Appendix D reports bias checks.

Training uses two stages and yields two SkillNav variants. Stage 1 fine-tunes DUET on R2R plus ScaleVLN augmentation and the Temporal synthetic set (or SRDF augmentation for the SRDF variant) to get a skill-agnostic backbone; Appendix F analyzes the Temporal Order Planning agent. Stage 2 fine-tunes that backbone on each skill dataset to produce five specialists: Direction Adjustment (π_{da}), Vertical Movement (π_{vm}), Stop and Pause (π_{sp}), Landmark Detection (π_{ld}), and Area/Region Identification (π_{ar}), denoted $\mathcal{S} = \{\pi_{\text{da}}, \pi_{\text{vm}}, \pi_{\text{sp}}, \pi_{\text{ld}}, \pi_{\text{ar}}\}$. Unless stated, SkillNav refers to the ScaleVLN-Aug variant.

4.3 VLM-based Action Router

After training specialized agents for distinct navigation skills, we integrate them into the SkillNav framework using a VLM-based Action Router to coordinate their execution. Inspired by planning systems such as LLM-Planner (Song et al., 2023), Mic (Qiao et al., 2023), and A2Nav (Chen et al., 2023), our router leverages a large VLM (*e.g.*, GPT-4o (OpenAI, 2024), Qwen2.5-VL-7B-Instruct (Bai et al., 2025)) in a zero-shot in-context fashion to dynamically select the most suitable agent at each time step.

We structure the routing process into three distinct reasoning phases:

Phase 1: Temporal Reordering Module. The Temporal Reordering Module only takes the original natural language instruction as input. It applies the instruction reordering prompt to turn navigation instructions into a list of subgoals I_{reorder}. It follows the four temporal relations described in the Temporal Order Planning skill in Section 4.1, making implicit temporal details explicit and ensuring the correct subgoal execution order. This procedure is formulated as

$$I_{\text{reorder}} = \text{LLM}_{\text{TemporalReorder}}(I). \tag{2}$$

Phase 2: Subgoal Localizer. Given the reordered subgoals $I_{\text{reorder}} = [p_1, p_2, \ldots, p_m]$, observed history H_{t-1}, and the sequence of previously executed subgoals $G_{t-1} = [p_1^*, \ldots, p_{t-1}^*]$, the model identifies the next subgoal p_t^* to be executed for the current time step t and outputs the corresponding reasoning trace r_t, later used by the router for decision verification. The output can be formalized as:

$$p_t^*, r_t = \texttt{Localize}(I_{\text{reorder}}, H_{t-1}, G_{t-1}). \tag{3}$$

The sequence of executed subgoals is then updated as:

$$G_t = G_{t-1} \parallel p_t^*. \tag{4}$$

Phase 3: Skill Router. At time step t, the skill router determines which skill-based agent $\pi_t^* \in \mathcal{S}$ is most appropriate for executing the selected subgoal p_t^*. Additionally, it receives the original instruction I as a part of the input context to capture additional linguistic cues such as verbs and spatial references. It also uses the reasoning trace r_t from Phase 2 to enhance its understanding of the current subgoal. At each step, exactly one skill is selected, formulated as

$$\pi_t^* = \arg\max_{\pi \in \mathcal{S}} \texttt{Router}(I, p_t^*, r_t). \tag{5}$$

Once the appropriate skill-based agent is selected, it is invoked by the following Eq. (1) to predict the navigation action at time step t:

$$a_t^* = \pi_t^*(I, O_t, M_t). \tag{6}$$

Our router enables modular skill execution by integrating natural language, visual inputs, and observed history, using the Temporal Reordering LLM to bridge instructions with actionable skill modules.

5 Experiments

Evaluation Datasets. We primarily use the Room-to-Room (R2R) dataset (Anderson et al., 2018), especially the unseen split of validation (Val Unseen) and test (Test Unseen) splits. R2R is a commonly used benchmark in VLN consisting of panoramic

Table 2: Performance comparison on R2R and GSA-R2R benchmarks. † indicates large-scale data augmentation; – indicates the unavailable statistics; bold is best, underline is second-best, and Δ rows report SkillNav minus its base (ScaleVLN/SRDF), with dark gray for gains and light gray for drops. Supervised and skill-based models are trained on R2R only and without GSA-R2R fine-tuning. SkillNav variants follow the ScaleVLN and SRDF settings with their respective augmentations.

Methods	#	R2R								GSA-R2R					
		Val-Unseen				Test-Unseen				Test-R-Basic		Test-N-Basic		Test-N-Scene	
		NE↓	OSR↑	SR↑	SPL↑	NE↓	OSR↑	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
LLM-based VLN															
MapGPT (GPT4v) (Chen et al., 2024a)	1	5.63	58	44	35	–	–	–	–	34	30	25	23	25	23
NavCoT (LLaMA2) (Lin et al., 2024)	2	6.26	42	34	29	–	–	–	–	37	35	29	26	29	26
NavGPT-2 (FlanT5-5B) (Zhou et al., 2024b)	3	3.13	81	72	61	3.33	80	72	60	58	45	48	35	**57**	43
NaviLLM (Vicuna-7B) (Zheng et al., 2024)	4	3.51	–	67	59	3.71	–	68	60	–	–	–	–	–	–
DiscussNav (GPT4) (Long et al., 2024)	5	5.32	61	43	40	–	–	–	–	–	–	–	–	–	–
Supervised VLN															
HAMT (Chen et al., 2021)	6	2.29	–	66	61	3.93	72	65	60	48	44	42	38	34	30
DUET (Chen et al., 2022b)	7	3.31	81	72	60	3.65	76	69	59	58	47	48	37	40	30
BEVBERT (An et al., 2023)	8	2.81	84	75	64	3.13	81	73	62	58	45	46	35	39	27
GR-DUET (Hong et al., 2025)	9	–	–	–	–	–	–	–	–	69	64	57	52	48	43
SAME (Zhou et al., 2024c)†	10	2.73	–	76	66	3.03	–	74	64	–	–	–	–	–	–
ScaleVLN (Wang et al., 2023) †	11	2.39	88	79	70	2.73	84	77	68	78	67	69	57	55	43
SRDF (Wang et al., 2024b) †	12	1.83	89	84	78	1.88	**88**	**84**	77	71	63	59	49	52	43
Mixture of Skill-based VLN															
SkillNav (ScaleVLN-Aug)†	13	1.97	**89**	83	77	2.53	83	78	70	**79**	**69**	**72**	**61**	**57**	**48**
Δ SkillNav (ScaleVLN-Aug) - ScaleVLN		−0.42	+1.77	+3.36	+6.54	−0.20	−1.65	+0.88	+1.80	+0.71	+2.18	+2.45	+4.18	+2.16	+5.26
SkillNav (SRDF-Aug)†	14	**1.79**	89	**84**	**78**	**1.76**	87	84	**77**	71	64	61	50	54	45
Δ SkillNav (SRDF-Aug) - SRDF		−0.04	−0.26	+0.20	+0.22	−0.12	−0.82	−0.28	+0.09	+0.56	+1.02	+2.83	+0.87	+2.78	+2.02

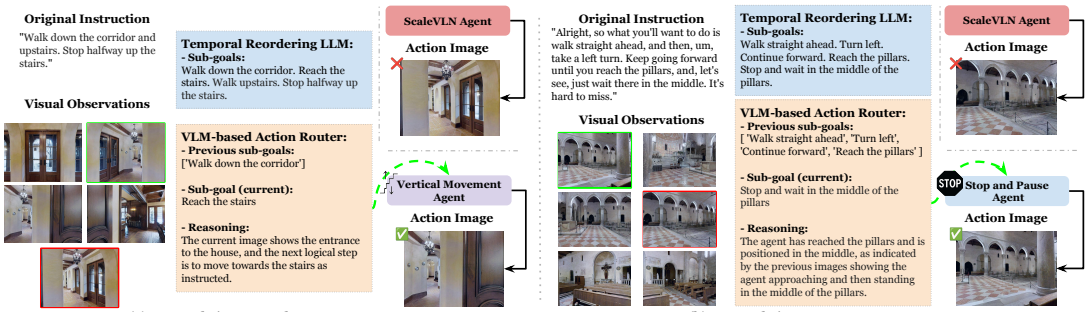
RGB-D scans from the Matterport3D (Chang et al., 2017) simulator and providing crowd-sourced instructions paired with navigation paths. Moreover, we evaluate the generalization ability of SkillNav on GSA-R2R (Hong et al., 2025), which includes residential (R) and non-residential (N) scenes (*e.g.*, shops, restaurants, and museums) from Habitat-Matterport3D (Ramakrishnan et al., 2021), and diverse instruction styles with role-specific dialogues (*e.g.*, travel guides (Scene) beyond the basic style of R2R (Basic).

Evaluation Metrics. We use the standard metrics to evaluate the navigation performance (Anderson et al., 2018; Zhao et al., 2023): (1) Navigation Error (NE): the distance between the stop location and the target; (2) Oracle Success Rate (OSR): the agent ever gets close enough to the goal at any point along its trajectory, regardless of where it decides to stop; (3) Success Rate (SR): the ratio of agents stopping within 3 meters of the target; (4) Success rate weighted by Path Length (SPL): measure navigation efficiency by weighting the success rate with the ratio between the shortest path length and the agent's actual path length, penalizing unnecessarily long trajectories.

Implementation Details. We develop two SkillNav variants tailored to the ScaleVLN and SRDF settings, respectively. For the ScaleVLN configuration, we employ CLIP-B/16 (Radford et al., 2021) as the visual encoder and BERT-base-uncased (Devlin et al., 2018) as the language backbone. Conversely, the SRDF setup swaps the visual encoder to InternViT-6B (Chen et al., 2024b). In our SkillNav VLM-based action router, we adopt GPT-4o (OpenAI, 2024) as the temporal reordering module due to its superior instruction-following capabilities and employ Qwen2.5-VL-7B-Instruct (Bai et al., 2025) for the other two phases because of its strong multi-modal alignment and reasoning abilities. All inferences with the action router are performed using in-context prompting. Unless otherwise specified, SkillNav denotes the ScaleVLN setting variant by default. Additional skill-based agent training, hyperparameter, and routing details are provided in Appendix E.

5.1 Main Results

Table 2 compares SkillNav against state-of-the-art VLN agents, demonstrating its superior generalization to novel instructions and environments while maintaining competitive performance on the R2R benchmark. Regarding generalization, while the strong baseline SRDF (Method #12) overfits to R2R's specific style, the ScaleVLN-based SkillNav (Method #13) excels, outperforming baselines on the challenging Test-N-Scene (48% vs. 43% SPL). On the R2R benchmark, SkillNav consistently improves or matches its respective baselines. Regarding R2R, SkillNav consistently improves or matches its specific backbones: the ScaleVLN-based variant boosts its baseline (Method #11) by $\sim$ 6.5% SPL on Val-Unseen, while the SRDF-based variant (Method #14) maintains parity with the state-of-the-art SRDF at 77% SPL, successfully



(a) A sample in R2R Val Unseen (b) A sample in GSA-R2R Test-N-Scene

Figure 3: Qualitative examples of routing and navigation results. These examples include cases where the instruction is temporally complex, colloquial, or spatially ambiguous.

Table 3: Temporal reordering ablation for the VLM-based action router on GSA-R2R across residential (R) and non-residential (N) scenarios with Basic/Scene instructions. Reorder: ✗ disables the LLM-guided Temporal Reordering module; ✔ enables it, and the ordered subgoal plan is fed only to the router while skill agents still receive the original instruction. Router: includes the Subgoal Localizer and Skill Router phases; **Qwen**: Qwen2.5-VL-7B-Instruct; **GLM**: GLM-4.1V-9B-Thinking.

Reorder	Router	Test-R-Basic		Test-N-Basic		Test-N-Scene	
		SR	SPL	SR	SPL	SR	SPL
✗	Qwen	78.42	67.80	71.01	59.62	55.46	45.43
✔	Qwen	**78.83**	**68.88**	**71.58**	**61.34**	56.66	**47.96**
✗	GLM	77.46	66.27	70.70	58.63	55.62	42.64
✔	GLM	78.60	67.93	71.13	59.73	**56.80**	46.51

Table 4: VLM choice for subgoal localization and skill routing on GSA-R2R. The Temporal Reordering module is fixed to *enabled*. Router Definitions: **Random** selects skill-based agents without action routing; **GPT-4o** (GPT-4o-2024-08-06); **Qwen** and **GLM** follow the definitions in Table 3. Results indicate that once temporal structure is provided, performance differences across routers become small, indicating robustness to the specific VLM router.

Router	Test-R-Basic		Test-N-Basic		Test-N-Scene	
	SR	SPL	SR	SPL	SR	SPL
Random	78.39	67.46	70.93	59.71	54.61	43.17
GLM	78.60	67.93	71.13	59.73	56.80	46.51
Qwen	78.83	68.88	71.58	61.34	56.66	47.96
GPT-4o	**79.41**	**69.18**	**72.75**	**62.48**	**58.16**	**48.96**

balancing versatility with precision.

5.2 Ablation Study and Analysis

Action Router. We evaluate the Action Router structural mechanism in Table 3 and backbone capability in Table 4. We maintain that operating on raw instructions forces difficult implicit temporal reasoning that limits open-world generalization (Intelligence et al., 2025). Disabling Temporal Reordering causes consistent performance drops, marked by a 2.5% SPL decline on Test-N-Scene, which confirms that explicit decomposition acts as a necessary structural scaffold. Complementing this, Table 4 shows that while a Random baseline suffices for basic instructions, novel environments require advanced reasoning. GPT-4o demonstrates the strongest generalization, proving that explicit reasoning and structural priors are key differentiators for handling unseen scenes.

Skill Evaluation. We validate our skill-based agents on NavNuances, where categorization into four atomic skills reveals that each agent excels in its corresponding domain in Table 5. These results confirm that targeted supervision fosters functional specialization superior to SOTA VLN baselines in isolated skill settings. See Appendix H.4 for mixed-skill analysis. Furthermore, Table 6 reveals that the agent adopts a conservative, precision-first strategy. The dominance of the control-focused policies defined in Secection 4.2, specifically π_{sp} and π_{da} (nearly 60%), indicates that robust navigation depends less on constant semantic recognition and more on continuous state verification. In contrast, semantic skills (π_{ar}, π_{ld}) act as sparse anchors, activated only when specific visual grounding is strictly necessary for decision-making. In addition, we provide a detailed skill routing analysis in Appendix H, and the human evaluation summary and the qualitative failure-mode breakdown in Appendix H.3.

5.3 Efficiency Analysis

Training Cost. Fine-tuning five skills on the Temporal Order Planning agent with R2R and synthetic skill-specific datasets under the SRDF setting requires approximately $3,329$ minutes (~ 55.5 hours) in total. For comparison, SRDF training on R2R with large data augmentation takes $2,521$ minutes (~ 42 hours), suggesting that SkillNav's

Table 5: Evaluation of each skill-based agent on the NavNuances benchmark across four skill categories: Direction Change (DC), Vertical Movement (VM), Landmark Recognition (LR), and Room Recognition (RR). Following the NavNuances, evaluation metrics differ across skill subsets: DC and LR are reported only with SR, VM includes SR/OSR/SPL, and RR provides SR/OSR. We retain this heterogeneous metric design to ensure comparability with prior work. Ident.: Identification.

	Methods	DC	VM			LR	RR	
		SR	SR	OSR	SPL	SR	SR	OSR
VLN Agents	ScaleVLN (Wang et al., 2023)	68.39	81.76	88.82	76.34	28.32	82.91	95.27
	SRDF (Wang et al., 2024b)	59.93	82.94	**91.18**	80.98	26.28	77.09	94.55
	Mixed Skills	66.84	84.11	87.65	79.22	**48.90**	81.82	90.91
Skill-based Agents	Direction Adjustment	**70.81**	81.76	**91.18**	76.28	31.39	81.82	94.91
	Vertical Movement	70.68	**87.65**	89.41	**83.83**	30.22	82.18	96.00
	Landmark Detection	70.29	82.35	85.29	78.94	**31.53**	83.64	**97.09**
	Area and Region Ident.	67.53	84.12	88.82	80.49	29.20	**85.09**	96.36
	Stop and Pause	68.91	84.71	87.06	80.67	29.78	83.64	**97.09**

Table 6: Expert call distribution on GSA-R2R Test-N-Scene. The values indicate the percentage of steps each specialist policy (π) was activated by the router.

π_{sp}	π_{da}	π_{ar}	π_{ld}	π_{vm}
34.42%	23.61%	18.75%	14.23%	8.99%

skill-based training introduces a relatively higher training cost. However, this represents a one-time training investment; unlike prior supervised VLN models that require repeated retraining to adapt to new environments or instruction styles, SkillNav achieves strong generalization across datasets without additional retraining.

Inference Cost. We provide inference time and throughput comparison in Table 7. SkillNav introduces overhead due to the VLM-based action router, reaching 0.49 throughput on Test-N-Basic of GSA-R2R, which is roughly 50× slower than ScaleVLN but still nearly 20× faster than MapGPT. See Appendix Table 16 for router-level breakdowns. Overall, while SkillNav is less efficient than supervised models, it achieves a better efficiency-generalization trade-off. Also, it advances both efficiency and generalization compared to LLM-based VLN agents.

5.4 Qualitative Examples

Figure 3 highlights SkillNav's dynamic decision-making capabilities, showing how the framework effectively reorders temporal instruction plans and identifies active subgoals to select the precise skill and action required for every navigation step. Specifically, in example (a), the router correctly identifies the destination (pillars) to trigger the stop action, while in (b), it detects the need for elevation and selects the vertical movement skill. These cases confirm the framework's robustness across diverse instruction styles and environments.

Table 7: Runtime and throughput of baselines and SkillNav. Numbers are wall-clock runtime in seconds. Random = randomly select skill-based agents without utilizing the action router.

Method	Split	Runtime (s)	Inferences/s
Supervised VLN			
ScaleVLN	Test-R-Basic	513.8	**28.03**
	Test-N-Basic	342.7	**26.26**
LLM-based VLN			
MapGPT	Test-R-Basic	$\sim 597,000$	0.02
	Test-N-Basic	$\sim 373,000$	0.02
Our Mixture of Skill-based VLN			
SkillNav	Test-R-Basic	$\sim 27,000$	0.54
	Test-N-Basic	$\sim 18,360$	0.49

6 Conclusion

We introduce SkillNav, a framework combining skill-based learning with VLM-based routing to dynamically select expert policies. Evaluations demonstrate strong performance on R2R and superior generalization on GSA-R2R. While incurring moderate overhead, SkillNav achieves a better efficiency-performance trade-off than pure VLM or supervised baselines. Our framework provides a novel and interpretable approach that advances compositional reasoning and generalization for the VLN research community.

Limitations

First, SkillNav is evaluated only in discrete VLN simulator environments (R2R, GSA-R2R, and NavNuances), leaving open the challenge of extending to continuous or real-world robotic navigation. While we evaluate our method on widely used benchmarks to enable controlled and fair comparison with prior work, these simulated environments

remain limited in reflecting the full complexity of real-world deployment settings.

Second, similar to other recent multi-agent and hierarchical frameworks (Long et al., 2024; Zhang et al., 2025a; Wang et al., 2025), our modular design incurs higher computational overhead compared to monolithic end-to-end models. Integrating high-level VLM planners with multiple low-level experts increases inference latency. While this trade-off is necessary to achieve superior generalization and interpretable reasoning, optimizing the efficiency of multi-agent coordination remains a broad challenge for the real-time deployment of LLM-based navigation systems.

Despite these constraints, our strong performance on the GSA-R2R benchmark, which introduces diverse scenes and novel instruction styles, provides evidence of improved generalization capabilities beyond standard VLN settings. Extending our framework to real-world embodied platforms, similar to the demonstration settings in DiscussNav (Long et al., 2024), and continuous environments like VLN-CE remains an important direction for future work.

Finally, while our skill decomposition framework shares foundational principles with recent advances in computer use and GUI agents, such as the Hierarchical Multimodal Skills module in Mirage-1 (Xie et al., 2025), our current investigation is strictly confined to embodied navigation. Although SkillNav and GUI agents both decompose complex behaviors into reusable atomic skills, we do not empirically verify whether our specific routing mechanisms and skill definitions are directly transferable to non-embodied domains. Validating this cross-domain extensibility remains an open question for future research.

References

Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. 2023. Bevbert: Multimodal map pre-training for language-guided navigation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*.

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

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee K. Wong. 2024a. MapGPT: Map-Guided Prompting with Adaptive Path Planning for Vision-and-Language Navigation. *arXiv preprint*. ArXiv:2401.07314 [cs].

Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H. Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. 2023. A2nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *CoRR*, abs/2308.07997.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021. History aware multimodal transformer for vision-and-language navigation. *Advances in neural information processing systems*, 34:5834–5847.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. 2022a. Learning from unlabeled 3d environments for vision-and-language navigation. In *ECCV*.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. 2022b. Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-Language Navigation. *arXiv preprint*. ArXiv:2202.11742 [cs].

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, and 1 others. 2024b. Internvl: Scaling up vision foundation models and aligning

for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13137–13146.

Haodong Hong, Yanyuan Qiao, Sen Wang, Jiajun Liu, and Qi Wu. 2025. General scene adaptation for vision-and-language navigation. In *The Thirteenth International Conference on Learning Representations*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. A Recurrent Vision-and-Language BERT for Navigation. *arXiv preprint*. ArXiv:2011.13922 [cs].

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, and 17 others. 2025. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*.

Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, Jianfeng Gao, and 1 others. 2024. Multimodal foundation models: From specialists to general-purpose assistants. *Foundations and Trends® in Computer Graphics and Vision*, 16(1-2):1–214.

Jialu Li and Mohit Bansal. 2023. Panogen: Text-conditioned panoramic environment generation for vision-and-language navigation. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*.

Jialu Li, Hao Tan, and Mohit Bansal. 2022. Envedit: Environment editing for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15407–15417.

Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. 2024. NavCoT: Boosting LLM-Based Vision-and-Language Navigation via Learning Disentangled Reasoning.

Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. 2021. Vision-language navigation with random environmental mixup. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1644–1654.

Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2024. Discuss before moving: Visual language navigation via multi-expert discussions. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17380–17387. IEEE.

OpenAI. 2024. Hello gpt-4o.

Yanyuan Qiao, Yuankai Qi, Zheng Yu, Jing Liu, and Qi Wu. 2023. March in chat: Interactive prompting for remote embodied referring expression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15758–15767.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

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. 2021. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*.

Liuyi Wang, Zongtao He, Jinlong Li, Xiaoyan Qi, Mengxian Hu, Chenpeng Yao, Chengju Liu, and Qijun Chen. 2025. Clash: Collaborative large-small hierarchical framework for continuous vision-and-language navigation. *Preprint*, arXiv:2512.10360.

Zehao Wang, Minye Wu, Yixin Cao, Yubo Ma, Meiqi Chen, and Tinne Tuytelaars. 2024a. Navigating the nuances: A fine-grained evaluation of vision-language navigation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4681–4704, Miami, Florida, USA. Association for Computational Linguistics.

Zun Wang, Jialu Li, Yicong Hong, Songze Li, Kunchang Li, Shoubin Yu, Yi Wang, Yu Qiao, Yali

Wang, Mohit Bansal, and Limin Wang. 2024b. Bootstrapping Language-Guided Navigation Learning with Self-Refining Data Flywheel. *arXiv preprint*. ArXiv:2412.08467 [cs].

Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. 2023. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12009–12020.

Tong Xiao and Jingbo Zhu. 2025. Foundations of large language models. *arXiv preprint arXiv:2501.09223*.

Yuquan Xie, Zaijing Li, Rui Shao, Gongwei Chen, Kaiwen Zhou, Yinchuan Li, Dongmei Jiang, and Liqiang Nie. 2025. Mirage-1: Augmenting and updating gui agent with hierarchical multimodal skills. *Preprint*, arXiv:2506.10387.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024a. Vision-language models for vision tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 46(8):5625–5644.

Siqi Zhang, Yanyuan Qiao, Qunbo Wang, Longteng Guo, Zhihua Wei, and Jing Liu. 2025a. Flexvln: Flexible adaptation for diverse vision-and-language navigation tasks. *arXiv preprint arXiv:2503.13966*.

Yue Zhang, Quan Guo, and Parisa Kordjamshidi. 2024b. Navhint: Vision and language navigation agent with a hint generator. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 92–103, St. Julian's, Malta. Association for Computational Linguistics.

Yue Zhang and Parisa Kordjamshidi. 2023. Vln-trans: Translator for the vision and language navigation agent. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13219–13233, Toronto, Canada. Association for Computational Linguistics.

Yue Zhang, Tianyi Ma, Zun Wang, Yanyuan Qiao, and Parisa Kordjamshidi. 2025b. Vision-and-language navigation with analogical textual descriptions in LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 15017–15025, Suzhou, China. Association for Computational Linguistics.

Yue Zhang, Ziqiao Ma, Jialu Li, Yanyuan Qiao, Zun Wang, Joyce Chai, Qi Wu, Mohit Bansal, and Parisa Kordjamshidi. 2024c. Vision-and-Language Navigation Today and Tomorrow: A Survey in the Era of Foundation Models. *arXiv preprint*. ArXiv:2407.07035 [cs].

Yue Zhang, Zhiyang Xu, Ying Shen, Parisa Kordjamshidi, and Lifu Huang. 2025c. SPARTUN3d: Situated spatial understanding of 3d world in large language model. In *The Thirteenth International Conference on Learning Representations*.

Chongyang Zhao, Yuankai Qi, and Qi Wu. 2023. Mind the Gap: Improving Success Rate of Vision-and-Language Navigation by Revisiting Oracle Success Routes. *arXiv preprint*. ArXiv:2308.03244 [cs].

Duo Zheng, Shijia Huang, Lin Zhao, Yiwu Zhong, and Liwei Wang. 2024. Towards learning a generalist model for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13624–13634.

Ce Zhou, Qian Li, Chen Li, Jun Yu, Yixin Liu, Guangjing Wang, Kai Zhang, Cheng Ji, Qiben Yan, Lifang He, and 1 others. 2024a. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. *International Journal of Machine Learning and Cybernetics*, pages 1–65.

Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. 2024b. NavGPT-2: Unleashing Navigational Reasoning Capability for Large Vision-Language Models. *arXiv preprint*. ArXiv:2407.12366 [cs].

Gengze Zhou, Yicong Hong, Zun Wang, Chongyang Zhao, Mohit Bansal, and Qi Wu. 2024c. Same: Learning generic language-guided visual navigation with state-adaptive mixture of experts. *arXiv preprint arXiv:2412.05552*.

Gengze Zhou, Yicong Hong, and Qi Wu. 2023. NavGPT: Explicit Reasoning in Vision-and-Language Navigation with Large Language Models. *arXiv preprint*. ArXiv:2305.16986 [cs].

Appendix

A NavNuances Skill Details

A.1 Atomic Skill Definitions and Annotations

Our atomic skills are derived from the NavNuances framework, which decomposes VLN instructions using a context-free grammar into fundamental instruction categories. In SkillNav, we adopt four atomic skills that are most directly involved in physical navigation and spatial grounding, namely Direction Adjustment, Vertical Movement, Landmark Detection, and Area and Region Identification. Each skill is constructed using strict geometric, semantic, and visibility constraints to ensure that it represents a minimal and isolatable navigation capability in the discrete VLN setting.

In SkillNav, atomic skills are defined at the semantic task level, rather than at the low-level motor execution level. This distinction is crucial for understanding what each skill represents and why they are considered atomic.

At the semantic level, an atomic skill corresponds to a single, indivisible navigation intent expressed in language, such as turning, changing floors, following a landmark, or transitioning between regions. Each skill isolates one core spatial concept and does not mix multiple independent goals. At the execution level, the same semantic skill may be realized through multiple discrete actions in the simulator, such as several forward steps or minor orientation changes. These low-level motions are treated as motor realizations of the same semantic skill, not as separate skills.

Take "Walk to the far end of the room" as an example: At the semantic level, the instruction "Walk to the far end of the room" expresses a single region-based navigation intent: move to an extreme spatial location within the same room. Therefore, it is classified as one atomic skill under Area and Region Identification. At the execution level, the agent may: (1) Slightly adjust its heading. (2) Move forward across multiple viewpoints. (3) Stop near the far boundary of the room. Although multiple low-level actions are involved, these are all sub-actions serving the same semantic goal, so the instruction still activates only one atomic skill. It becomes a multi-skill instruction only when additional independent semantic goals are explicitly introduced, such as landmarks, turning commands, or vertical motion.

Direction Adjustment. This skill captures pure changes in heading direction, corresponding to actions such as turn left, turn right, and turn around. (1) The direction change is defined by the heading angle difference between successive viewpoints. (2) The angular change must be significant enough to indicate a clear directional shift, but not so large as to cause a U-turn or double-turn behavior, which would introduce ambiguity. (3) We also incorporate linguistic variants of directional intent, such as "slightly turn", "face to", and "adjust towards". These phrases are normalized to their effective heading change in degrees, ensuring that natural language variation does not alter the underlying atomic skill semantics.

Vertical Movement. This skill models motion across different floor levels and corresponds to go upstairs and go downstairs. (1) Only candidates with significant elevation change (absolute height difference greater than 2 units) are considered, filtering out small slopes and near-flat transitions. (2) The candidate viewpoint must be explicitly associated with vertical structures, such as staircases. (3) The sign of the elevation change determines the movement type and must be consistent along the entire trajectory, for example, it is not allowed to first go upstairs and later go downstairs within a single instance.

Landmark Detection. This skill corresponds to actions such as walk towards or walk past a specific object and focuses on visually grounded object-level navigation. (1) The viewpoint must contain salient and visually distinctive landmarks, such as sofas, desks, lamps, paintings, or large appliances, which are clearly visible. (2) If a landmark is referenced over multiple steps, it must persist across successive views, allowing the agent to maintain a

stable spatial relationship with respect to the same object. This ensures that the skill evaluates continuous landmark grounding rather than accidental encounters.

Area and Region Identification. This skill models semantic transitions between rooms or functional areas, such as go into the bedroom or exit the living room. (1) Each trajectory must include at least one explicit region transition. (2) Paths involving ambiguous, erroneous, or unrecognized region labels are removed or sanitized. (3) Only horizontal region transitions are isolated, avoiding confounding factors from vertical movement. Unlike landmark-based skills, region transitions are annotated using region inclusion criteria rather than a single distance-based endpoint, reflecting the semantic nature of rooms.

We intentionally exclude Numerical Comprehension from the skill executor set because it mainly tests sequential counting and memory along corridors, rather than continuous motion control. It is instead handled implicitly by the higher-level routing module when needed.

These four skills are considered "atomic" because they correspond to the lowest-level, non-decomposable navigation operations in the discrete VLN setting. They each focus on one isolated capability. They do not mix multiple semantic or geometric concepts. Each instruction in NavNuances is generated to isolate exactly one such capability using the CFG production rules.

In the R2R-style discrete viewpoint selection setting, these skills are operationally independent. For example, for the sub-goal "Go through the double doors", only the Landmark Recognition skill is activated. The agent directly selects the candidate viewpoint that contains the double doors in its panoramic observation, and then traverses them. This execution does not rely on Direction Adjustment, because the simulator action is viewpoint selection rather than continuous steering.

A.2 Explanation on Lowest Success Rate of Landmark Recognition in NavNuances

The relatively low performance in the Landmark Recognition (LR) category mainly stems from the fine-grained nature of both the task and its evaluation. First, LR requires instance-level grounding rather than category-level recognition. The evaluation protocol does not merely check whether an object category is detected, but whether the agent's final position satisfies precise geometric constraints with respect to the specific landmark instance, such as reducing the distance to the object center for "walk towards," or truly crossing the object for "walk past." This level of precision goes well beyond the coarse object semantics typically learned in VLN training.

Second, annotation difficulty and viewpoint sparsity further increase the challenge. Due to the discrete navigation graph and limited viewpoints in Matterport3D environments, landmarks are often visible only from a narrow range of poses. Even small deviations in viewpoint selection can cause large geometric errors under the distance-based evaluation, which amplifies minor perception or control mistakes into outright failures.

Third, models frequently struggle with the spatial relational concept of "walk past." As shown by failure cases, agents often stop beside the object and mistakenly interpret proximity as having passed it, or confuse front and back views of the landmark. This indicates that the bottleneck is not simply recognizing the object but understanding the temporal and geometric meaning of passing an object, which requires consistent spatial reasoning across multiple steps.

Finally, we hypothesize that the main limitation is not low-level object detection but higher-level spatial reasoning and decision-making. This is supported by the observation that stronger vision-language models substantially improve the "walk towards" subset, where recognition dominates, but still struggle with "walk past," where reasoning about relative motion and spatial relations is critical. This directly motivates our design choice of using a VLM as a routing and reasoning module, where rich visual perception is explicitly coupled with instruction-conditioned geometric and commonsense reasoning, rather than relying solely on perception-driven navigation policies.

B Primary Factors of Trajectory Generation

As introduced in Section Skill-Specific Data Synthesis and Agent Training in Methodology, we construct 5 skill-specific datasets and train the agents based on them. The primary factors for the construction of each skill are as follows:

Temporal Order Planning. (1) A random initial move is selected. (2) Staying in the same region (*e.g.*, hallway $\rightarrow$ hallway) for the first half of the

trajectory to encourage temporal continuity at first. (3) Once halfway through, the agent is allowed (and encouraged) to transition to new regions.

Direction Adjustment. (1) The direction change is based on the heading degree. (2) It should be significant enough to indicate a directional shift, but not so large as to cause a reversal or double-turn behavior.

Vertical Movement. (1) Only candidates with significant elevation (more than ± 2) are considered, which filters out nearly flat or slight inclines/declines. (2) The candidate viewpoint must be explicitly marked as vertically relevant (*e.g.*, stairs). (3) The elevation sign determines movement type, and it must be consistent with the applied trajectory. For instance, it is impossible to go upstairs and then go downstairs in one case.

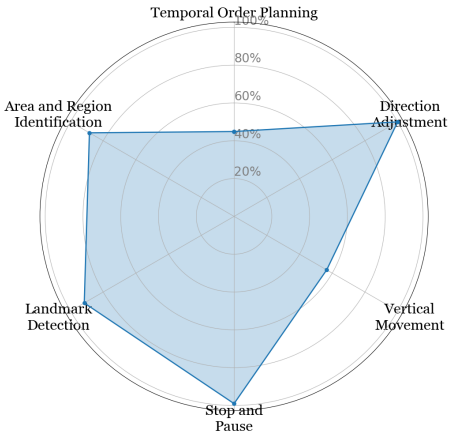


Figure 4: Distribution of instructions in the R2R dataset categorized by the proposed skill taxonomy.

Stop and Pause. (1) The stop should occur at a place with or after semantically relevant context for pausing, *e.g.*, in front of a painting, at the foot of stairs. (2) The candidate image is very similar to the previous viewpoints.

Landmark Detection. (1) The viewpoint must include obvious, visually distinctive landmarks or objects (*e.g.*, sofa, desk, painting, lamp) clearly visible in the image. (2) If a landmark is to be referenced over multiple steps, it should appear persistently in successive views, allowing the agent to maintain spatial awareness relative to that object.

Area and Region Identification. (1) A trajectory must include at least one region change. (2) Paths with "Error" or unrecognized regions are ignored or sanitized. (3) All horizontal region changes are isolated.

C Path Length in Trajectory Generation

We constrain trajectory length to 4–7 steps to keep the difficulty and temporal context comparable to natural VLN data. Figure 5 shows the statistics of the path length. To be noted, the R2R, ScaleVLN, SRDF datasets, and our Temporal Order Planning datasets have significantly fewer instructions with a 4-step trajectory.

D Synthetic Data Integrity and Leakage Analysis

We analyze our synthetic data to test whether agents learn meaningful semantics rather than overfitting to linguistic triggers, and to verify that the datasets do not leak information into evaluation splits.

D.1 Skill Keyword Coverage

We randomly sampled 100 instructions from each skill-specific dataset and manually extracted high-frequency keywords or phrases associated with motion, regions, and objects. The utilization of GPT-4o for instruction generation during dataset construction introduces linguistic diversity, expanding the vocabulary beyond the R2R and GSA-R2R datasets. For instance, the Direction Adjustment synthetic dataset includes complex phrases such as “adjust your direction to align with...”.

Human evaluation on the synthetic datasets reveals that keyword presence is not a reliable proxy for action, preventing agents from overfitting to specific words. A notable example is the preposition “down". In the instruction “proceed down the hallway", the term “down" implies forward progression from one end to the other, whereas in vertical contexts, it denotes an elevation change. This semantic ambiguity forces the agent to rely on visual context and sentence structure rather than mapping the word “down” exclusively to a vertical motion primitive.

Table 8 reports keyword coverage across datasets. Each synthetic dataset exhibits a dominant proportion of keywords associated with its target skill, confirming the effectiveness of targeted data generation. At the same time, substantial linguistic overlap and semantic complexity across datasets encourage the learning of robust, context-aware policies instead of dataset-specific shortcuts.

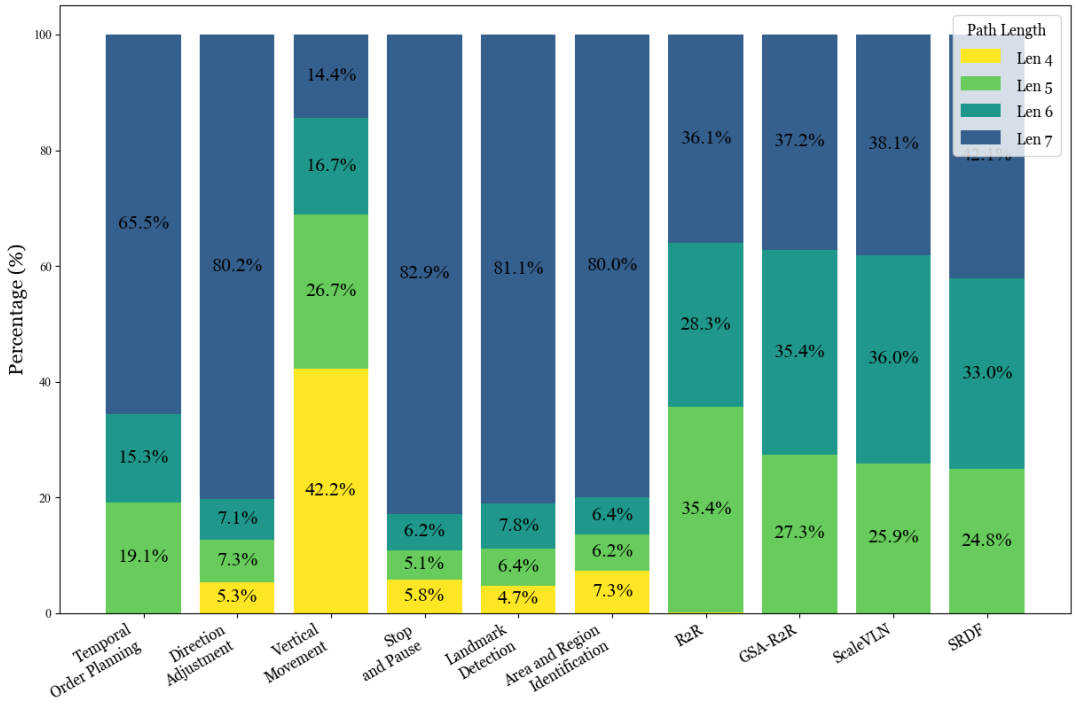


Figure 5: The statistics of the path length of our synthetic datasets compared with existing VLN datasets. The R2R, ScaleVLN, SRDF datasets, and our 6 skill-specific datasets are all for training, while only GSA-R2R is for evaluation.

The Vertical Movement dataset provides a particularly informative case. As shown in Table 8, it contains a high density of non-vertical concepts, including Landmark Object keywords (18.72%) and Direction Motion keywords (8.05%), compared with only 3.32% Vertical Motion keywords. This reflects the physical nature of vertical navigation, which necessarily involves horizontal motion relative to landmark structures. Despite this lexical overlap, the Vertical Movement dataset remains the primary source of Vertical Motion keywords, with less than 0.2% in all other datasets. Moreover, all trajectories in this dataset are strictly filtered to ensure consistent elevation changes in the simulator. As a result, the agent cannot rely on generic movement words or visual objects such as stairs as direct triggers for vertical behavior. Instead, performance gains arise from the precise alignment between linguistic cues, visual affordances, and trajectory constraints. This alignment encourages the learning of a robust compositional policy rather than a brittle keyword-to-action mapping.

Consequently, the performance gains come from the precise alignment of instruction-trajectory pairs, which sharpens the agent's focus on the target skill. Simultaneously, the diverse lexical environment ensures this focus evolves into a robust, compositional policy rather than degrading into a simple keyword-to-action trigger.

D.2 Skill Agents to Skill Synthetic Datasets

To further assess specialization and generalization, we evaluate each skill-specific agent on both its own synthetic dataset and the other skill-specific datasets. Table 9 reports the success rate (SR) and success weighted by path length (SPL) for all cross-skill evaluations. Each agent achieves its strongest performance on the dataset corresponding to its trained skill, while maintaining non-trivial performance on the remaining datasets. This pattern indicates that although the agents develop skill-specific expertise, they do not collapse into narrowly overfit behaviors.

D.3 Skill to Temporal Subsets

We additionally conduct cross-style evaluation on each skill-based agent using synthetic temporal datasets. For this setting, each agent is trained on its skill-specific synthetic data combined with the temporal order planning synthetic data, and then evaluated on four temporal subsets filtered from R2R Val Unseen. These subsets capture different forms of temporal reasoning, including Conditional Immediate, Duration Bound, Forward Sequential,

Table 8: Keyword coverage analysis across synthetic datasets. The coverage is measured by the proportion of words in the dataset that match the target skill keywords. Ident.= Identification.

Dataset / Keyword	**Direction Motion**	**Vertical Motion**	**Stop Motion**	**Landmark Object**	**Region Type**
Direction Adjustment	14.56%	0.03%	0.06%	13.64%	9.62%
Vertical Movement	8.05%	3.32%	0.01%	18.72%	9.46%
Stop and Pause	12.56%	0.18%	18.12%	18.02%	9.57%
Landmark Detection	9.53%	0.11%	0.97%	27.88%	7.58%
Area and Region Ident.	7.86%	0.05%	0.10%	16.39%	19.89%

Table 9: Each synthetic subset reports both SR and SPL for agents trained and tested on synthetic instructions.

Agent	**Direction Synthetic**		**Vertical Synthetic**		**Stop Synthetic**		**Landmark Synthetic**		**Region Synthetic**	
	SR	**SPL**	**SR**	**SPL**	**SR**	**SPL**	**SR**	**SPL**	**SR**	**SPL**
Direction DUET	49.33	43.35	38.89	30.77	41.11	36.44	38.44	34.12	35.56	30.92
Vertical DUET	35.56	31.11	65.11	55.65	39.78	35.96	37.78	33.69	36.00	30.72
Stop DUET	36.22	31.79	34.67	27.37	52.67	48.55	42.22	38.76	42.22	36.12
Landmark DUET	36.00	30.46	46.00	38.35	41.11	36.69	44.22	39.98	38.67	34.02
Region DUET	34.00	28.89	49.56	41.51	38.44	33.97	36.44	32.90	36.67	43.02

and Reverse Sequential instructions.

Table 10 presents SR and SPL on the four temporal subsets. All evaluations are performed on synthetic instructions that are specifically designed to remove superficial linguistic cues. The consistent performance across temporal subsets demonstrates that agents trained with skill-specific synthetic data retain strong compositional and temporal reasoning abilities, rather than overfitting to surface-level instruction patterns.

D.4 Data Leakage Assessment

We evaluate whether GSA-R2R gains could be driven by data leakage from shared scenes or instruction augmentation. Two potential channels are considered: (1) *vision-level scene overlap* through HM3D-based augmentation and (2) *language-level instruction overlap* between training data and the GSA-R2R test sets. HM3D overlap is shared by all competing methods and does not uniquely benefit SkillNav, and we find no meaningful language-level leakage from either ScaleVLN or our synthetic datasets.

Vision-Level Scene Data Overlap with HM3D. Each SkillNav agent is fine-tuned with ScaleVLN augmentation data that includes 899 HM3D scans; GSA-R2R also contains HM3D scenes. SRDF and ScaleVLN use the same HM3D-based augmentation, so any scene-level overlap affects all methods equally rather than uniquely benefiting SkillNav. Our synthetic skill trajectories are constructed only from the seen split of MP3D R2R, not from GSA-R2R, preventing direct trajectory or instruction leakage.

No Language-Level or Instruction Leakage. We first analyze the vocabulary overlap between the **ScaleVLN augmentation instructions** and the **GSA-R2R test instructions**, since ScaleVLN is used for vision-level fine-tuning and is the primary potential source of language leakage.

All instructions are lowercased and tokenized using the regular expression (`[a-zA-Z']+`) to construct unique vocabularies. Under this protocol, the ScaleVLN augmentation file contains **208 unique tokens**. Although ScaleVLN includes 2,891,134 instruction entries, these instructions are generated by a constrained template-based pipeline with a fixed pool of action verbs, spatial prepositions, and object categories. As a result, the number of unique token types remains small despite the large number of instruction instances.

ScaleVLN vs. GSA-R2R Vocabulary Overlap. Table 11 reports the vocabulary overlap between ScaleVLN augmentation instructions and the three GSA-R2R test splits. While ScaleVLN covers a large fraction of its own limited vocabulary in GSA-R2R, the coverage of the GSA-R2R test vocabulary by ScaleVLN remains low. In particular, across the union of all three GSA-R2R test sets, only 196 out

Table 10: Each temporal subset reports both SR and SPL, and all agents are trained and tested on synthetic instructions designed to remove superficial linguistic cues.

Agent	Conditional Immediate		Duration Bound		Forward Sequential		Reverse Sequential	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL
ScaleVLN	84.29	76.29	76.27	67.45	79.53	68.92	74.29	66.97
Temporal DUET	88.57	82.18	84.18	74.90	85.83	76.93	88.57	81.72
Direction DUET	84.29	77.57	83.05	76.69	85.83	77.99	84.29	76.91
Vertical DUET	86.43	79.71	79.66	74.31	86.61	80.35	80.00	75.08
Stop DUET	85.00	79.91	81.92	74.59	87.40	80.34	82.86	77.15
Landmark DUET	87.14	81.58	80.79	75.43	88.19	81.47	81.43	77.72
Region DUET	85.00	80.06	79.66	72.93	87.40	80.42	74.29	69.86

of 1,645 tokens overlap, corresponding to **11.9% test vocabulary coverage**. This indicates that the majority of GSA-R2R test words are not observed in ScaleVLN augmentation.

Table 11: Vocabulary overlap between ScaleVLN augmentation instructions and GSA-R2R test sets.

GSA-R2R Test Split	Overlap	Test Vocab Size	Coverage (%)
Residential Basic	186	514	36.2
Non-Residential Basic	166	578	28.7
Non-Residential Scene	168	1,376	12.2
Union of All Tests	196	1,645	11.9

We also report the vocabulary coverage of our GPT-4o-generated synthetic skill datasets with respect to the union of all GSA-R2R test vocabularies. The atomic skill coverage ranges from **21.95% to 27.72%** across different skill types. Even in the highest case (Temporal Order Planning skill), more than 58% of the GSA-R2R test vocabulary remains unseen. This confirms that our synthetic language does not duplicate the GSA-R2R test distribution.

Taken together, these results demonstrate that the performance gains on GSA-R2R cannot be explained by either vision-level or language-level data leakage from HM3D, ScaleVLN, or our synthetic annotations. Instead, the improvements stem from the proposed skill decomposition and compositional learning framework.

E Reproducibility Details

We evaluate two model configurations: the standard ScaleVLN setup and the SRDF-augmented SkillNav variant. Both architectures utilize ViT-B/16 to extract 768-dimensional object features and adhere to an identical two-stage training protocol.

For the ScaleVLN configuration, we employ CLIP-B/16 (Radford et al., 2021) as the visual encoder with a feature dimension of 512 and BERT-base-uncased (Devlin et al., 2018) as the language backbone, while utilizing ViT-B/16 to extract object features with a dimension of 768. During the skill training, we fine-tune the DUET pretrained model with Temporal Order synthetic data, ScaleVLN augmentation data, and R2R Train data for $50,000$ iterations using a batch size of 32 and a learning rate of 5×10^{-5} on 1 NVIDIA A6000 GPU with the random seed 0. The best finetuned Temporal DUET model is selected based on the SPL performance on the R2R Validation Unseen dataset. Based on the Temporal DUET, we employ the second round fine-tuning with atomic skill synthetic data for $30,000$ iterations with a batch size of 16 on the same GPU, a protocol that the SRDF-augmented SkillNav variant also follows regarding optimizer, learning rate, and iteration budget.

For the SRDF-augmented SkillNav variant, we replace the visual encoder with InternViT-6B and set the feature dimension as 3200 while retaining the same 768-dimension object features from ViT-B/16. Training follows the same two-stage recipe as the ScaleVLN variant: (1) 50,000 iterations on Temporal Order synthetic data, ScaleVLN augmentation, and R2R Train with AdamW, batch size 32, and learning rate 5×10^{-5}; (2) 30,000 iterations on atomic skill synthetic data with batch size 16. Both stages run on a single NVIDIA A6000 GPU with random seed 0, keeping optimizer and hyperparameters identical across variants.

The router is run with a maximum context length of 40,960 tokens, a temperature of 0, and greedy decoding. All router inferences are executed under

the vLLM framework using five beam search candidates. The routing decision is performed in `top1` mode with `argmax` feedback. Instruction reordering is enabled, and routing weights are treated as integers and normalized before use.

All experiments are evaluated in test-only mode with detailed output and submission enabled. Multiple expert checkpoints are loaded through `resume_files`, with their corresponding normalized `resume_weights`. The external VLM router is accessed via a dedicated server URL specified by `router_server_url`. The feature backbone uses CLIP ViT-B/16 with 512-dimensional visual features, and all evaluations are conducted with a fixed random seed 0 for deterministic behavior. Batch size is explicitly specified in the execution scripts for each model configuration.

To support reproducibility of the routing and fallback behavior, we release the exact action routing prompt template used for Qwen2.5-VL-7B-Instruct and GLM-v4.1-9B-Thinking in Appendix M.3, Listing 4 and Listing 5.

F Temporal Order Planning Agent

As introduced earlier, the training of each skill-based agent follows a two-stage fine-tuning strategy. In the first stage, we fine-tune a pre-trained DUET model using a combination of the R2R training split, ScaleVLN augmentation data, and our proposed Temporal Synthetic dataset, resulting in a strong skill-agnostic backbone. We evaluate this first-stage model on the R2R Val Unseen split across four temporal logic subsets.

Temporal Order Planning captures the agent's ability to reason over the sequence and structure of subgoals. Compared to ScaleVLN, our model demonstrates improved temporal reasoning capabilities, as detailed in Table 12. This improvement comes from enhanced **Temporal Order Planning**, which enables the agent to reason about the sequence and structure of subgoals. The Temporal Order Planning subsets include:

- **Conditional immediacy**: The agent must execute an action immediately after a specific condition is met. These instructions are typically triggered by phrases such as *once*, *as soon as*, or *upon*. (*e.g.*, "Once you enter the hallway, turn left")

- **Bounded duration**: The agent is required to maintain an action until a specific condition becomes true. These instructions use keywords such as *until* or *while*. (*e.g.*, "Keep walking until you see the staircase")

- **Forward sequential**: These instructions describe a sequence where Action B follows Action A in order. Temporal cues include *then*, *finally*, *before*, and *after*. (*e.g.*, "Go forward, then turn right, and finally stop")

- **Backward sequential**: Action B is described first but should occur only after Action A. These often use similar cues as (*e.g.*, "Before turning, make sure you're at the hallway entrance"), but the order of mention and execution differs.

Unlike low-level action chaining, temporal order planning involves higher-level temporal logic that determines when and how atomic skills should be executed in sequence. As shown in Table 12, our Temporal Synthetic Data improves navigation in failure cases where prior methods such as ScaleVLN struggle.

G Stop and Pause Agent

The Stop and Pause agent integrates two stopping mechanisms within the DUET framework: (1) the agent can explicitly issue a stop action at a given viewpoint; and (2) if the agent does not explicitly stop when the navigation loop ends, DUET retrospectively selects the visited location with the highest stop probability and optionally appends a shortest path to reach it. Since we apply a stopping-focused data augmentation strategy that exposes the model to diverse stop-relevant cues during training, this supervision enables the agent to distinguish between the two stopping mechanisms and to learn when stopping aligns with the instruction intent and visual context. Although NavNuances does not include a dedicated stopping split, our Stop agent still outperforms generalist baselines like ScaleVLN and SRDF across all skill categories (Table 5), suggesting that effective stopping is a foundational capability that influences the success of diverse navigation behaviors.

H Skill Routing Effectiveness

H.1 Action Router Ablations

The Random baseline achieves a high success rate of 78.39% on Test-R-Basic, confirming that the

Table 12: Navigation performance across 4 temporal logic instructions from R2R Val Unseen dataset. **Bold** values denote metrics that exceed the R2R Val Unseen average, while gray values indicate metrics that fall below the average. Temporal DUET is the agent fine-tuned with the Temporal Order Planning synthetic dataset in the first training stage.

Model	Val Unseen	Conditional Immediacy	Bounded Duration	Forward Sequential	Backward Sequential
ScaleVLN (SR)	79.23	**84.29**	76.27	**79.53**	74.29
Temporal DUET (SR)	82.59	**88.57**	**84.18**	**85.83**	**88.57**
ScaleVLN (SPL)	69.96	**76.29**	67.45	68.92	66.97
Temporal DUET (SPL)	75.40	**82.18**	74.90	**76.93**	**81.72**

Table 13: VLM choice for subgoal localization and skill routing on GSA-R2R. The Temporal Reordering module is fixed to *enabled*. Router Definitions: **Random** selects skill-based agents without action routing; **GPT-4o** (GPT-4o-2024-08-06); **Qwen** (Qwen2.5-VL-7B-Instruct); **GLM** (GLM-4.1V-9B-Thinking). Results indicate that once temporal structure is provided, performance differences across routers become small, indicating robustness to the specific VLM router.

Router	Test-R-Basic		Test-N-Basic		Test-N-Scene	
	SR	SPL	SR	SPL	SR	SPL
Random	78.39	67.46	70.93	59.71	54.61	43.17
GLM	78.60	67.93	71.13	59.73	56.80	46.51
Qwen	78.83	68.88	71.58	61.34	56.66	47.96
GPT-4o	**79.41**	**69.18**	**72.75**	**62.48**	**58.16**	**48.96**

underlying skill-based agents are inherently capable of executing navigation subgoals. However, the necessity of active reasoning becomes evident in the Test-N-Scene split. In these novel environments, VLM-based routers consistently outperform the random baseline, proving that semantic understanding is crucial for correctly routing instructions to skills when the scene is unknown. Among the models, GPT-4o demonstrates the strongest generalization, achieving the highest Test-N-Scene performance with $\mathbf{58.16}\%$ SR and $\mathbf{48.96}$ SPL. This indicates that while the base agents provide a solid foundation, advanced reasoning models are the key differentiator for handling the complexity of unseen scenes.

H.2 Skill Transition

Regarding inter-skill interaction and interference, we computed the transition probability matrix $P(\text{next} \mid \text{current})$ to map temporal dependencies between skills (Table 4). The data suggests cooperative chaining rather than negative interference. For instance, Landmark Detection has a very high probability of transitioning to Stop and Pause (0.3516), indicating that the agent correctly utilizes landmarks as verification signals to identify the goal location. Similarly, Directional Adjustment frequently leads to Area and Region Identification (0.3916), suggesting a behavior where the agent re-orients itself physically before re-evaluating the semantic context of the room. Notably, the transition from Landmark to Direction is 0.00, implying that once a specific landmark is identified, the agent prioritizes verification (stopping) or vertical navigation over immediate rotational correction.

We analyze the skill routing in R2R Val Unseen of SkillNav with Qwen2.5-VL-7B in the following table

H.3 Human Evaluation of Skill Routing

We examined 17 distinct failure cases where specific step-level data was available. Quantitatively, the average success rate of the routing mechanism, defined as the proportion of steps completed successfully before the first failure, is 24.5%:

$$\frac{1}{N}\sum \frac{\text{first_failure_step} - 1}{\text{\#routing_steps}}$$

This indicates that while the agent effectively initiates navigation, critical routing errors tend to occur in the early-to-mid stages of the trajectory.

Skill Selection and Visual Grounding. The core analysis reveals that the router's failures are not random but instead exhibit a clear pattern of visual grounding deficits. As shown in Table **??**, there is a significant discrepancy between the skills selected by the router and those actually required by the environment. The *Landmark Detection* expert is severely under-utilized, being the optimal skill in 76% (13 of 17) of the analyzed failures, yet the router correctly selected it in only one instance. Instead of activating the *Landmark* expert to identify specific objects—such as a curio cabinet or

Table 14: Inter skill transition probabilities $P(\text{next} \mid \text{current})$.

	Direction	Vertical	Stop	Landmark	Region
Direction	0.3449	0.1971	0.1374	0.2108	0.3916
Vertical	0.3415	0.2414	0.1729	0.0931	0.1035
Stop	0.0610	0.2385	0.2296	0.1854	0.2984
Landmark	0.0000	0.2044	0.3516	0.2711	0.0280
Region	0.2525	0.1186	0.1084	0.2397	0.1785

bed—the router frequently defaulted to coarser navigational skills like *Region Identification* (6 cases) or *Directional Adjustment* (5 cases). This mismatch indicates that while the router successfully interprets the linguistic instruction to move or change areas, it fails to bind the instruction to the specific visual target required to trigger the *Landmark* expert. This confirms that the VLM experts still struggle to bind visual object references effectively even with Temporal Reordering in place, suggesting that the primary error source is visual grounding rather than high-level planning logic.

Failure Modes. Qualitative analysis identifies several specific mechanisms by which these failures manifest.

A frequent cause of early failure is initial viewpoint misalignment, where the agent starts with an orientation that does not align with the instruction; this initial drift often propagates downstream as the router struggles to recover from an incorrect starting heading.

Hallucinations also play a significant role in derailing routing. For example, the agent generated a subgoal to "Position yourself beside the glass case with the dolls" when the instruction simply stated "Enter the bedroom," illustrating that the action router can create irrelevant descriptions grounded in neither the text nor the view which subsequently propagate incorrect skill choices.

Furthermore, even when the router selects the correct skill, execution can still fail due to controller limitations. For instance, valid *Direction* transitions failed when multiple direction subgoals were stacked (e.g., "Turn left. Turn right. Turn left again."), indicating that the skill-conditioned controllers themselves can become a bottleneck during complex sequences. Finally, the router occasionally fails to generate necessary "Stop" subgoals or struggles with ambiguous spatial prepositions, leading to incorrect termination points.

In summary, the analysis demonstrates that the routing mechanism is not acting as a random black box. It exhibits strong instruction understanding capabilities—often sequencing subgoals correctly in text—but acts as a bottleneck when it relies on textual priors over visual evidence. The frequent substitution of *Region/Direction* skills for *Landmark* skills highlights that improving the router's ability to attend to visual object cues is the most critical path for future performance gains.

H.4 SkillNav is Not Data Augmentation

We clarify that SkillNav is not a data augmentation method, even though it uses synthetic skill-specific data for training individual skill agents. The role of synthetic data in our framework is to enable explicit specialization of atomic skills, not to improve a single monolithic policy. The core contribution of SkillNav is the explicit decomposition of navigation into atomic skills and the dynamic recomposition of these skills at inference time through a structured MoE routing mechanism. This is fundamentally different from data augmentation, which still relies on a single end-to-end policy without modular execution or dynamic expert selection.

To empirically address this concern, we conducted a controlled experiment (Table 15 and Table 5) comparing SkillNav against a "Mixed Skills" baseline, a single monolithic DUET agent fine-tuned on the union of all synthetic skill data used in SkillNav.

If the performance gains were driven solely by data augmentation, the "Mixed Skills" agent should match SkillNav's performance. However, our results show the opposite, confirming that structural decomposition (MoE), not data volume, is the driver of improvement.

As shown in Table 15, the "Mixed Skills" agent achieves an SR of 54.26% on **Test-N-Scene**, which is statistically indistinguishable from the baseline ScaleVLN (54.50%). In stark contrast, SkillNav achieves 56.66% SR ($+2.4\%$ absolute gain). This gap confirms that simply exposing a model to compositional data is insufficient. The model requires

the MoE architecture to dynamically route subproblems to specialized experts rather than forcing a single network to learn conflicting behaviors.

Table 5 illustrates why the monolithic approach fails and why our MoE method aligns with the motivation of compositional reasoning.

- **Task Interference in Monolithic Models:** The "Mixed Skills" agent attempts to learn all behaviors simultaneously, resulting in diluted performance. For example, on Direction Change (DC), the Mixed model scores 66.84%.
- **Superiority of Specialized Experts:** The specialized "Direction Adjustment" expert achieves 70.81% on DC, significantly outperforming the mixed model. Similarly, the "Vertical Movement" expert achieves 87.65% SR on VM tasks, compared to 84.11% for the mixed model.

This demonstrates that SkillNav functions as a true MoE system: it leverages agents that have "overfit" to specific atomic skills (high performance in their domain, lower in others) and uses the router to compose them. A data augmentation approach cannot replicate this sharpness in skill execution.

The Role of Temporal Reordering in MoE Finally, we clarify that the Temporal Reordering Module is not an auxiliary processing step but a structural prerequisite for the MoE router. As shown in our ablations, without reordering, the router struggles to simultaneously decouple temporal segmentation from skill classification. The reordering module provides the necessary planning scaffold (an ordered list of subgoals), allowing the router to focus exclusively on visual grounding and expert selection.

H.5 Novel Instruction Handling Mechanism

We clarify that the improvement on Test-N-Scene does not come from the Temporal Reordering module alone, nor is it due to any scene-specific instruction handling built into the model. Instead, the gain arises from the joint effect of structured temporal planning and Mixture-of-Skills execution, which together enable SkillNav to better handle complex, long-horizon, and stylistically diverse instructions.

The Temporal Reordering Module plays a structural role inside the routing pipeline. It converts free-form instructions into an explicitly ordered subgoal sequence and supplies this sequence **only to the action router for subgoal localization**. Importantly, it does not execute actions, nor does it provide any extra semantic supervision to the skill agents. All skill-based agents still operate solely on the original full instruction and visual observation. Therefore, temporal reordering alone cannot account for the navigation improvement.

The action router is responsible for mapping each localized subgoal to the appropriate specialized skill agent. Without the MoE routing, the system cannot translate the reordered plan into skill-conditioned execution. This is directly supported by our ablation results: when routing is random or misaligned, performance drops even with temporal reordering enabled. Conversely, when both reordering and structured MoE routing are enabled, performance becomes stable and significantly improves, especially on Test-N-Scene.

Scene-style instructions in Test-N-Scene are typically written in a travel-speaker or tour-guide style, with rich descriptive language, delayed action cues, and loosely structured narration. For example, instructions often follow a pattern such as: “Alright folks, let’s take a gentle stroll down these beautiful marble steps… keep going straight ahead… you’ll notice the stone pillars on your left… right beyond them, the large double doors… that’s where we’ll pause.”

In summary, the performance gains on Test-N-Scene cannot be attributed to the reordering module alone. Temporal reordering reduces ambiguity in planning, while the MoE router is essential for translating each subgoal into the correct expert action. The improvement comes from their tight integration, validating that the gain is fundamentally due to structured compositional reasoning via skill recomposition, rather than instruction parsing alone.

Qualitative Example: Routing on a Scene-Style Instruction. To illustrate this synergy, consider a typical narrative instruction from Test-N-Scene:

“Alright folks, let’s take a gentle stroll down these beautiful marble steps here. Keep going straight ahead, and you’ll notice those magnificent stone pillars on your left. Just keep moving forward until you see the wooden confessionals and, right beyond them, those impressive large double doors. That’s where we’ll pause.”

The Temporal Reordering module first distills this narrative into discrete subgoals (e.g., “Begin walking down the marble steps,” “Observe the

Table 15: GSA-R2R DUET agent performance (SR/SPL) on Test-R-Basic, Test-N-Basic, and Test-N-Scene. Ident. = Identification. It shows that data augmentation alone fails.

Agent	Test-R-Basic		Test-N-Basic		Test-N-Scene	
	SR	SPL	SR	SPL	SR	SPL
ScaleVLN	78.12	66.70	69.13	57.16	54.50	42.70
Mixed Skills	77.60	66.76	70.23	59.32	54.26	43.68
Direction Adjustment	78.80	68.40	70.88	59.53	53.48	42.22
Vertical Movement	78.42	67.52	70.58	59.22	54.57	43.78
Landmark Detection	78.56	67.84	70.41	59.65	54.27	44.77
Area and Region Ident.	78.22	68.24	71.27	60.81	54.60	44.93
Stop and Pause	78.38	67.93	70.03	58.77	53.44	42.02
SkillNav	**78.83**	**68.88**	**71.58**	**61.34**	**56.66**	**47.96**

stone pillars," etc.). However, SkillNav does not rigidly execute this list; instead, it dynamically routes control based on the visual environment.

I Efficiency Analysis

All experiments in efficiency analysis in Section 5.3 run on NVIDIA A6000. For the inference cost in Table 7, the number of predictions is $14,400$ for Test-R-Basic and $9,000$ for Test-N-Basic. For fairness, MapGPT is re-implemented with Qwen2.5-VL-7B-Instruct.

To isolate router overhead, we disable Temporal Reordering and compare three VLM routers on GSA-R2R. As shown in Table **??**, Qwen2.5-VL-7B-Instruct runs fastest at 18.12 seconds per sample, while GLM-4.1V-9B-Thinking and InternVL3.5-8B increase per-sample latency to 55.99 and 79.16 seconds, respectively. Heavier routers therefore incur substantial runtime cost without material gains in routing robustness.

Table 16: Runtime cost of VLM routers on GSA-R2R with Temporal Reordering disabled. Values are seconds per sample (lower is better).

Router	Throughput (s/sample)
Qwen2.5-VL-7B-Instruct	18.12
GLM-4.1V-9B-Thinking	55.99
InternVL3.5-8B	79.16

I.1 Why the skill-based agents are not VLM-based models

The SkillNav framework itself is model-agnostic and only assumes the availability of specialized skill executors. In principle, these skill executors could be instantiated by any policy model. In our implementation, however, we deliberately use pretrained supervised VLN backbones rather than VLMs for skill execution, based on both effectiveness and efficiency considerations.

From an effectiveness perspective, existing LLM or VLM-based VLN agents such as MapGPT, NavGPT-2, VLN-R1, and DiscussNav consistently lag behind supervised VLN models on R2R, and more importantly, they exhibit unstable generalization on GSA-R2R. This shows that directly using MLLMs as low-level action policies is still unreliable for precise and robust navigation, especially in unseen environments.

From an efficiency perspective, our analysis in Section 5.3 already shows that using a single MLLM as the router makes SkillNav about $50\times$ slower than a fully supervised VLN model. The throughput ablation in Table 16 further quantifies this cost across different routers: even among zero-shot VLMs, per-sample inference ranges from 18.12 to 79.16 seconds, indicating that the router alone is already the dominant computational bottleneck. If we further replace each low-level skill executor with an MLLM or VLM, the system would require multiple MLLM inferences per navigation step, which would introduce another order-of-magnitude slowdown. This would make the system impractical for real-time or large-scale deployment in real-world embodied settings.

In summary, SkillNav is a general, model-agnostic, skill-based framework that is fully compatible with the MLLM paradigm at the routing and planning level. However, for executing atomic navigation skills, we intentionally prefer pretrained, compact VLN backbones over VLMs, because they

provide significantly better accuracy, efficiency, and spatial control reliability for low-level navigation.

J Evidence of Supervised Agents' Memorization

While recent methods like ScaleVLN and SRDF have indeed narrowed the performance gap on standard splits (val-seen vs. val-unseen), this improvement does not reflect true generalization. Instead, our analysis suggests these methods rely on linguistic memorization (due to high distribution overlap) and task-specific overfitting (via augmentation construction), causing them to fail in rigorous generalization benchmarks like GSA-R2R.

Linguistic Memorization due to Distributional Homogeneity. The "unseen" environments in R2R benchmarks are not linguistically novel. Our analysis of the R2R instruction vocabulary reveals that the distribution of instructions in validation splits is statistically indistinguishable from the training distribution. There is a Jaccard **similarity score of 0.9992** between instruction sets, with effectively perfect token coverage. The Cosine similarity of token frequency vectors is 1.0, and the Jensen–Shannon divergence is negligible (8.2×10^{-6}).This implies that the "narrowing gap" the reviewer observes is likely due to models memorizing these static linguistic patterns, which remain constant across splits, rather than learning to ground instructions in truly unseen visual environments.

Failure in Robust Generalization Benchmarks (GSA-R2R). To test if the performance gap reduction translates to real-world robustness, we evaluated these methods on GSA-R2R, a benchmark designed to test generalization beyond standard splits.

Despite high scores on standard R2R, both ScaleVLN and SRDF exhibit significant performance drops in GSA-R2R testing.

Crucially, this failure persists even when the augmentation data includes scenes that were technically "seen" during training. This indicates that their success on R2R is brittle and relies on specific dataset biases that do not hold in broader generalization scenarios.

Methodological Memorization via Data Construction. Finally, we argue that the definition of "memorization" extends to the training methodology.

Methods like SRDF require constructing specific augmentation data tailored for each new task.

This reliance on constructing task-specific priors acts as a form of structural memorization. The model is not generalizing to a new task zero-shot; it is overfitting to the constraints provided by the engineered augmentation data.

K Limitations

First, SkillNav is evaluated only in discrete VLN simulator environments (R2R, GSA-R2R, and NavNuances), leaving open the challenge of extending to continuous or real-world robotic navigation.

Second, we evaluate our method on widely used VLN benchmarks that are commonly used for both supervised agents and LLM-based methods. These benchmarks enable controlled and fair comparison with prior work, but they indeed remain limited in reflecting real-world deployment settings. More recent datasets, such as GSA-R2R, introduce more diverse scenes and novel instruction styles, and our strong performance on this benchmark already provides evidence for improved generalization beyond VLN settings. We note that DiscussNav (Long et al., 2024) includes a real-world demonstration setting, and extending our framework to such real-world embodied platforms is an important direction for future work. We also discuss other continuous-environment VLN settings such as VLN-CE, which bridge perception and control, but still share similar structure and instruction styles with simulated benchmarks. Beyond embodied navigation, our skill decomposition framework is closely related to recent advances in computer use and GUI agents. For example, Mirage-1 (Xie et al., 2025) introduces a Hierarchical Multimodal Skills (HMS) module that explicitly abstracts trajectories into multiple levels, including execution skills, core skills, and meta-skills. This represents an even more explicit form of skill-level abstraction. Our method follows a similar principle by decomposing complex behaviors into reusable atomic skills and dynamically coordinating them through a high-level router. This close alignment with hierarchical skill-based GUI agents further supports that our approach captures general, transferable principles of multimodal decision-making, and is naturally extensible across different embodied and computer-use agent set-

tings. Our method follows a similar principle by decomposing complex behaviors into reusable atomic skills and dynamically coordinating them through a high-level router. This close alignment with hierarchical skill-based GUI agents further supports that our approach captures general, transferable principles of multimodal decision-making, and is naturally extensible across different embodied and computer-use agent settings.

L LLM Usage

We used LLM-based tools for polishing grammar and aiding in writing. In addition, we utilize LLM to generate synthetic instructions for skill-specific datasets as described in Section 4.2. Moreover, LLMs and VLMs serve as our temporal reordering module and action router in Section 4.3.

M LLM and VLM Prompts

In this section, we provide the prompts used in data construction and all components of SkillNav.

M.1 Prompts for Skill-specific Data Synthesis

To generate skill-focused instruction, we feed the observation sequence of each candidate trajectory into GPT-4o with the structured prompt, in Listing 1 and Listing 2. Both of prompts are tailored for GPT-4o.

Temporal Order Planning Skill Data Construction. The detailed prompt for Temporal Order Planning Skill data construction can be seen in Listing 1.

Atomic Skills Data Construction. The 5 atomic skills in VLN share the same prompt (in Listing 2) for their skill-specific data synthesis. .

M.2 Prompt for Temporal Reordering Module

The Temporal Order Module only takes the original natural language instruction as input. It applies the instruction reordering prompt to turn navigation instructions into subgoals I_{reorder}. The prompt is shown in Listing 3, utilizing GPT-4o as the generation model.

M.3 Prompts for Action Router

The Action Router dynamically selects the most suitable agent at each time step, which can be structured into two distinct reasoning phases: Phase 1 Subgoal Localizer and Phase 2 Skill Router. We provide the detailed prompt for the two phases, respectively. They can be used for either Qwen2.5-VL-7B-Instruct or GLM-4.1V-Thinking-9B .

Subgoal Localizer. The Subgoal Localizer identifies the next subgoal to be executed for the current time step and outputs the corresponding reasoning trace. Listing 4 contains the prompt for the subgoal localizer, which takes all reorder subgoals, the previously executed subgoals, and the prior selected viewpoints as input.

Skill Router. The skill router determines which skill-based agent is most appropriate for executing the selected subgoal among the 5 skill-based agents. Besides, it receives the original instruction as contextual input to capture additional linguistic cues such as verbs and spatial references. It also uses the reasoning trace from the subgoal localizer to enhance its understanding of the current subgoal. The whole process is displayed in Listing 5.

Execution begins when the router detects the stair structure and assigns the current subgoal to the Vertical Movement skill. Crucially, this agent receives the original full instruction. Upon activation, it parses the complete narrative to locate and attend to the specific phrase “down these beautiful marble steps,” grounding this text to the visual descent while ignoring later landmark descriptions.

Once the agent descends and aligns with the corridor, the router shifts control to the Direction Adjustment skill. This agent processes the full instruction to isolate the directive “keep going straight ahead,” executing the forward movement. Subsequently, when the router activates the Landmark Detection skill for the "stone pillars" subgoal, the agent again references the original text. It attends to the detailed description—“magnificent stone pillars on your left”—to perform precise visual grounding, ensuring the agent identifies the specific object mentioned in the tour.

Finally, the router tracks the destination using the Stop and Pause skill. This agent verifies the visual presence of the doors against the instruction’s final clause (“that’s where we’ll pause”) before terminating the episode. Throughout the trajectory, the router dictates which expert is active, but the experts themselves derive their visual-linguistic understanding directly from the rich, original instruction.

In summary, the improvements on Test-N-Scene are not attributable to better instruction parsing alone. While reordering provides necessary struc-

```
You are an expert in Vision-and-Language Navigation (VLN) and Language.

<Task>
Your task is to write natural, human-like navigation instructions based on a sequence of visual observations from an indoor
      environment.

<Instruction Guidelines>
- Do not use explicit temporal markers like ``then'', ``next'', ``before'', or ``after''.
- Imply sequence using spatial or contextual phrasing instead.
- Include only essential visual cues, such as layout, furniture, doorways, or notable landmarks that help clarify the path.
- Avoid over-descriptive or decorative language (e.g., ``intricate stonework'', ``high ceiling'').
- Keep the instruction fluent, intuitive, and helpful, like someone casually guiding a friend through a space.
- Keep it concise and comparable in length to a temporal-based instruction.

<Visual Reasoning Process>
Analyze each frame in the visual sequence. Focus on:
- Movement across spaces
- Transitions (e.g., turns, room entries)
- Orientation shifts
- Key visible cues needed to navigate the path

<Instruction Output>
Once you've analyzed the path:
- Write a fluent, natural-sounding instruction describing the full trajectory.
- Do **not** include reasoning steps.
- Output **only** the final instruction.

<Example Chain-of-Thought>
- 1st Frame:
    - The agent is inside a narrow wooden hallway-like space.
    - The doorway directly ahead leads to a brighter area.

- 2nd Frame:
    - The agent is almost at the threshold of the doorway.
    - You can see the hallway plant and the open area outside.

- 3rd Frame:
    - The agent is now fully outside the room, looking into a wide open space.
    - There's a visible bedroom to the left, and the plant in the yellow pot is to the right, indicating the agent has made a
          hard left turn.

- 4th Frame:
    - The agent is now facing a doorway to a bedroom on the left side.
    - The bed is partially visible inside.

- 5th Frame:
    - The agent has entered the room and is facing a window.
    - The position suggests the agent took one step inside and then stopped.

---

<Trajectory Images>
``{path_images}''
```

Listing 1: Prompt used for Temporal Order Planning Skill-specific Data Synthesis

ture to long narratives, the MoE routing mechanism provides the essential visual grounding required to execute that structure. The advantage of SkillNav lies in this tight integration of planning and compositional execution.

```
You are an expert in Vision-and-Language Navigation (VLN) and Language.

<Task>
- Generate a **single** natural-language instruction that guides an agent through the scene.

<Input>
- A visual sequence (an ordered list of images)
- A specific navigation skill to emphasize

<Requirements>
- The instruction should describe what the agent does across the image sequence (e.g., move, climb, pause).
- Ground the instruction in **visible cues**, such as layout, objects, stairs, doorways, lighting, or orientation.
- Emphasize the given **target skill** (e.g., "Direction Adjustment", "Vertical Movement", etc.), while naturally
      incorporating other relevant details as needed.
- The output must be a **single sentence**, written in fluent, natural language (no lists, quotes, or symbols).
- Instruction length should be **20-30 words** (aim for ~25).
- Do **not** include explanations, reasoning steps, or metadata output only the instruction itself.

<Available Skills>
{Direction Adjustment, Vertical Movement, Stop and Pause, Landmark Detection, Area and Region Identification}

<Skill Definitions>
- **Direction Adjustment**: Involves turning or changing heading. Look for instructions like ``turn left'', ``go back'', or
      ``face the hallway''. Used when the agent needs to rotate or reorient without necessarily changing position.

- **Vertical Movement**: Involves moving across floors or elevation changes. Triggered by terms like ``go upstairs'', ``down
      the stairs'', or ``take the elevator''. Watch for floor changes in visuals or references to vertical navigation.

- **Stop and Pause**: Involves coming to a full stop at a defined point. Use lighter-weight verbs such as pause, wait, and
      stand, when the stop happens in the middle of sequence (e.g., ``pause by the red sofa''). Use stronger, more terminal
      verbs like stop and come to a stop for the final action or true endpoint (e.g., ``stop at the glass doors''). This
      distinction helps the agent decide whether to hold briefly or end its navigation.

- **Landmark Detection**: Requires identifying and responding to specific objects or features in the environment. Triggered by
       mentions of visible items like ``lamp'', ``chair'', ``red sofa'', ``painting''. Used when object recognition is
      necessary to proceed or confirm position.

- **Area and Region Identification**: Involves recognizing or transitioning between distinct spaces or rooms. Triggered by
      mentions like ``enter the kitchen'', ``in the bedroom'', ``exit hallway''. Requires understanding of semantic regions
      based on context or appearance.


<Output Format>
Return only the instruction sentence. Do not include tags, labels, or formatting.

---

<Trajectory Images>
``{path_images}''

<Focused Skill>
``{skill_name}''
```

Listing 2: Prompt used for Atomic Skill-specific Data Synthesis

```
You are an expert at converting natural language navigation instructions into detailed, logically ordered sub-instructions for
       agents.

<Task>
- Break down instructions into a sequence of minimal, goal-directed steps.
- Make all implicit temporal or spatial relationships explicit.
- Preserve execution order by reconstructing intermediate actions that are implied, not directly stated.

<Logic Rules>
- (A) --> [after / then / once / as soon as] --> (B): Do A fully, then B.
- (B) --> [before] --> (A): Move toward A, then perform B at a point prior.
- (A) --> [until] --> (B): Continue A until B is reached.
- Avoid ``then'', ``before'', ``until'', ``once'' etc. in the output.

<Formatting Rules>
- Single sentence, steps separated by periods.
- Each step must be minimal, concrete, and goal-focused.

<Examples>
**Example 1:**
Instruction: ``Turn around and walk down the stairs. Stop once you get down them.''
Output:
Turn around. Walk down the stairs. Stop at the bottom of the stairs.

**Example 2:**
Instruction: ``Walk toward the dining room but turn left before entering it and go into the open area.''
Output:
Walk toward the dining room. Stop at the entrance. Turn left. Enter the open area.

**Example 3:**
Instruction: ``After you leave the laundry room, make a left in the hallway, and go to the bedroom straight ahead. When you
      are in the doorway of the room go to the doorway of the closet on the left and wait.''
Output:
Exit the laundry room. Turn left in the hallway. Walk to the bedroom straight ahead. Enter the doorway of the bedroom. Go to
      the doorway of the closet on the left. Wait there.

**Example 4:**
Instruction: ``Start moving forward down the corridor. You will pass offices on your left and right. Keep going down the
      hallway until you get to an exit sign on your right and what looks like some lockers in front of you. There will also be
       a brown door with an exit sign above it in front of you.''
Output:
Start moving forward down the corridor. Pass the offices on your left and right. Continue walking down the hallway. Reach the
      exit sign on your right and the lockers in front of you. Stop in front of the brown door with the exit sign above it.

---

<Original Instruction>:
``{instruction}''
```

Listing 3: Prompt used for Temporal Reordering

```
You are a visual reasoning assistant for indoor navigation.
<Task>:
Your task is to analyze a list of previously observed images and a natural language instruction.
Determine which parts of the instruction have already been completed, and return the next step to be executed.
<Response Rules>
Your response must:
- Return the next action using *exact phrasing* from the reordered instruction (no paraphrasing).
- Match the sub-instruction to the visual context from previous images.
- If the goal (e.g., pool table) has clearly been reached, return the final sub-instruction.
- If *all* sub-instructions have been completed based on the visual path, do not return anything further. Stop reasoning.
- If the final destination has been reached and the last step is a positional or waiting action (e.g., ``wait there'', ``step
      to the left''), return that as the next step.
- You must reason about whether the agent is already at the destination.
- If the current image shows the goal destination (e.g., inside the room with the pool table, or inside the open doorway), and
      the instruction contains a final step like ``wait'' or ``adjust your position'', that is the next sub-instruction.
---
Use the following reasoning strategy to determine what to do next:
<Step-by-Step Reasoning Instructions>:
1. Decompose the instruction into sub-instructions.
- Break the full instruction into smaller steps. Each sentence or clause typically represents one step.
- Example:
    - Original: ``At the bottom of the stairs, go through the nearest archway to your left. Head straight until you enter the
          room with a pool table. Step slightly to the left to get out of the way.''
    - Decomposed:
        - ``At the bottom of the stairs, go through the nearest archway to your left.''
        - ``Head straight until you enter the room with a pool table.''
        - ``Step slightly to the left to get out of the way.''
2. Use the previous sub-instruction list to identify completed steps.
- Do not reissue any previously executed sub-instructions.
- Compare upcoming steps against what may have been visually completed, even if not explicitly executed one-by-one.
3. Analyze the sequence of previous viewpoint images.
- Use visual context to infer if *multiple* sub-instructions have been completed in a single transition.
- If image progression clearly shows the agent has already bypassed an intermediate area or reached a later goal, mark those
      steps as implicitly complete.
4. Evaluate remaining sub-instructions for completion.
- If the current image shows the agent at or beyond the target of a sub-instruction, that step can be considered completed.
- If the current image shows the agent inside the goal location and only a final positional instruction remains (e.g., ``Step
      slightly to the left''), return that final instruction.
5. Select the next uncompleted sub-instruction that is visually and contextually justified.
- Use exact wording from the original instruction.
- Do not return instructions that the agent already visually fulfilled, even if they were skipped.
6. Output the result in the following JSON format:
{
"Sub-instruction to be executed": "<exact next instruction clause>",
"Reasoning": "<why this is the next step based on image sequence>"
}
CHECKPOINT:
If multiple sub-instructions were completed based on a single or continuous image segment, skip them and jump to the next
      logical, visually unfulfilled step.
---

Now, using the instruction and the visual history, identify the next step.
IMPORTANT: Your response must be a valid JSON object without any surrounding text, code blocks, or explanations.
Do not include markdown formatting like ```json or ```.

<Original Whole Instruction>:
``{instruction}''
<Previous Sub-Instructions>:
``{previous_sub_instructions}''
<Previous Viewpoint Images>:
```

Listing 4: Prompt used for Subgoal Localizer in Action Router

```
You are a visual reasoning assistant for indoor navigation.

<Available Skills>:
[``Direction Adjustment'', ``Vertical Movement'', ``Stop and Pause'', ``Landmark Detection'', ``Area and Region
      Identification'']

<Skills Explanation>:
- Direction Adjustment:
Involves turning or changing heading. Look for instructions like ``turn left'', ``go back'', or ``face the hallway''. Used
      when the agent needs to rotate or reorient without necessarily changing position.
- Vertical Movement:
Involves moving across floors or elevation changes. Triggered by terms like ``go upstairs'', ``down the stairs'', or ``take
      the elevator''. Watch for floor changes in visuals or references to vertical navigation.
- Stop and Pause:
Involves stopping at a specific location. Triggered by instructions like ``stop'', ``wait'', or ``stand in front of''. Used
      when the endpoint or a mid-action pause is important.
- Landmark Detection:
Requires identifying and responding to specific objects or features in the environment. Triggered by mentions of visible items
       like ``lamp'', ``chair'', ``red sofa'', ``painting''. Used when object recognition is necessary to proceed or confirm
      position.
- Area and Region Identification:
Involves recognizing or transitioning between distinct spaces or rooms. Triggered by mentions like ``enter the kitchen'', ``
      in the bedroom'', ``exit hallway''. Requires understanding of semantic regions based on context or appearance.

<Task>:
1. Read and understand the sub-instruction to be executed.
2. Use the reasoning explanation to infer what skills are likely required to carry out that sub-instruction.
3. Choose the top 1 skill that is most relevant to the sub-instruction.

<Input>:
You will be given:
- The original full navigation instruction.
- The sub-instruction that should be executed next, based on reasoning.
- A reasoning explanation derived from the visual history and instruction.

Output exactly **one skill name** from the above list.
Do not provide explanations or additional text.

<Output Format>:
*****SKILL_NAME*****

<Example>
Original Whole Instruction: ``At the bottom of the stairs, go through the nearest archway to your left. Head straight until
      you enter the room with a pool table. Step slightly to the left to get out of the way.''

Sub-instruction to be executed for next step: ``Head straight until you enter the room with a pool table.''

Reasoning based on previous viewpoints path and original instruction: The agent appears to be just outside the archway. The
      next step is likely to involve entering the archway and preparing to head straight.

Expected Output:
*****Landmark Detection*****

---

<Reordered Whole Instruction>:
``{full_instruction}''

Sub-instruction to be executed for next step:
``{sub_instruction}''

<Reasoning based on previous viewpoints path and original instruction>:
``{reasoning}''
```

Listing 5: Prompt used for Skill Router in Action Router