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

Tianyi Ma[1] **Yue Zhang**[1] **Zehao Wang**[2] **Parisa Kordjamshidi**[1]
[1]Michigan State University [2]ESAT-PSI, KU Leuven
matiany3@msu.edu

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 historical 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) [2, 48] is a critical 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 [53, 40, 21, 44], VLN has seen notable progress in multimodal grounding and generalization.

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 [14, 9, 37, 38]. While data-driven methods improve VLN agents' generalization, their main limitation is reliance on black-box, end-to-end models [2, 16] that tend to memorize training ex-

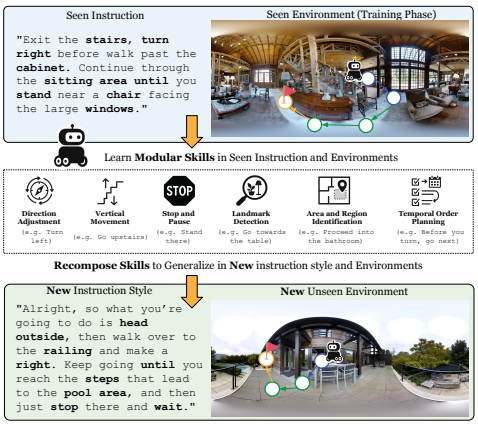


Figure 1: SkillNav decomposes complex navigation instructions into atomic skills, which can be flexibly recomposed to address new environments.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: SPACE in Vision, Language, and Embodied AI (SpaVLE).

amples. 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-driven approaches, recent work has explored zero-shot approaches leveraging Large Language Models (LLMs) for VLN tasks to improve generalization ability [54, 26, 5, 45]. 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 (approximately 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 that combine the broad generalization and compositional reasoning capabilities of LLMs with the domain-specific adaptability of fine-tuning strategies.

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 [39], 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, each proficient in its designated capability. After obtaining these agents, we then integrate them into a unified framework to perform complex navigation tasks. Moreover, we introduce a temporal reordering module to generate chronologically ordered sub-goals, facilitating effective temporal reasoning during skill selection. Finally, we integrate a VLM-based router that dynamically identifies the next relevant sub-goal and selects the most suitable skill-based agent to execute the corresponding navigation action.

SkillNav attains a strong performance on the Room-to-Room (R2R) benchmark [2], and achieves state-of-the-art (SOTA) generalization to the GSA-R2R benchmark [15] which introduces novel instructions and diverse visual environments, including both unseen residential and non-residential settings. Additionally, we evaluate individual skill-based agents using NavNuances [39], a dataset specifically designed for fine-grained skill evaluation. We provide comprehensive ablation studies 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 explicitly decomposes the navigation task into atomic, reusable skills, then recomposes them for execution, leveraging the specialization of fine-tuned VLN architectures together with the generalization capability of VLMs. This design significantly enhances generalization to novel instructions and visual environments.
2. We construct a synthetic dataset pipeline that enables skill-specific supervision without human annotation, producing diverse and linguistically natural data.
3. We demonstrate SOTA generalization on the challenging GSA-R2R dataset and provide a comprehensive analysis with ablation studies.

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 [2, 34] to Transformer-based models [8, 10, 1] and, most recently, to Large Language Model (LLM)-based agents [54, 5, 24, 55, 52, 49]. A critical challenge in VLN research is enhancing the generalization capability of agents, allowing them to navigate effectively in unfamiliar environments and handle novel instructions. To enhance generalization, most existing methods utilize data-driven augmentation

strategies, focusing either on augmenting visual observations [23, 25, 22] or synthesizing additional navigation instructions [37, 38, 14, 46, 47]. However, a fundamental limitation of purely data-driven augmentation approaches lies in their reliance on end-to-end training paradigms. Such monolithic models often memorize training examples rather than genuinely generalize, failing to fundamentally address the compositional reasoning required in novel or unseen scenarios. More recently, some approaches [54, 5, 26, 45] have explored zero-shot navigation by heavily depending on the general reasoning capabilities of LLMs without explicit training on task-specific datasets. However, their effectiveness remains constrained by the LLMs' inherent lack of detailed spatial understanding and precise grounding in real-world 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.

Skill-based MoE Systems. Mixture-of-Experts (MoE) models traditionally operate at the parameter level, distributing input across multiple expert networks to improve capacity and efficiency [17, 19, 43]. Sparsely activated MoEs [32, 20, 50, 57] further scale this idea by routing each input to a small subset of experts, making it possible to train trillion-parameter models while controlling inference cost. More recently, large language models have begun to employ skill-based MoEs at the module or LLM level, where different LLMs are specialized through fine-tuning or task profiling [31, 36, 12, 18, 41, 6, 56, 42], and expert selection is performed via prompting or routing mechanisms based on task semantics. While these skill-based MoE methods focus on video understanding [42] and visual or textual question-answering [6], they largely overlook embodied tasks such as VLN. Although a recent model, SAME [56], introduces a state-adaptive MoE framework for VLN, this approach lacks explicit skill representations and independent spatial grounding, limiting its interpretability and extensibility. In contrast, our framework explicitly defines skill-based MoE agents for VLN tasks, employing specialized skills to significantly enhance generalization, interpretability, and extensibility.

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 [10] as our base VLN agent. It is a dual-scale graph transformer solution that fuses the topological map with local observations for decision-making. We formulate it 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 a framework, **SkillNav**, for VLN that coordinates a set of atomic skill-based agents to solve navigation tasks. 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 specific trajectories and enabling systematic reuse of skills across environments and instruction styles. As shown in Figure 2, the framework comprises three components: a temporal reordering module for instruction decomposition, a VLM-based router for 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. This section introduces the proposed skill taxonomy, skill-specific synthetic dataset construction, and reasoning framework for acquiring these modular skills.

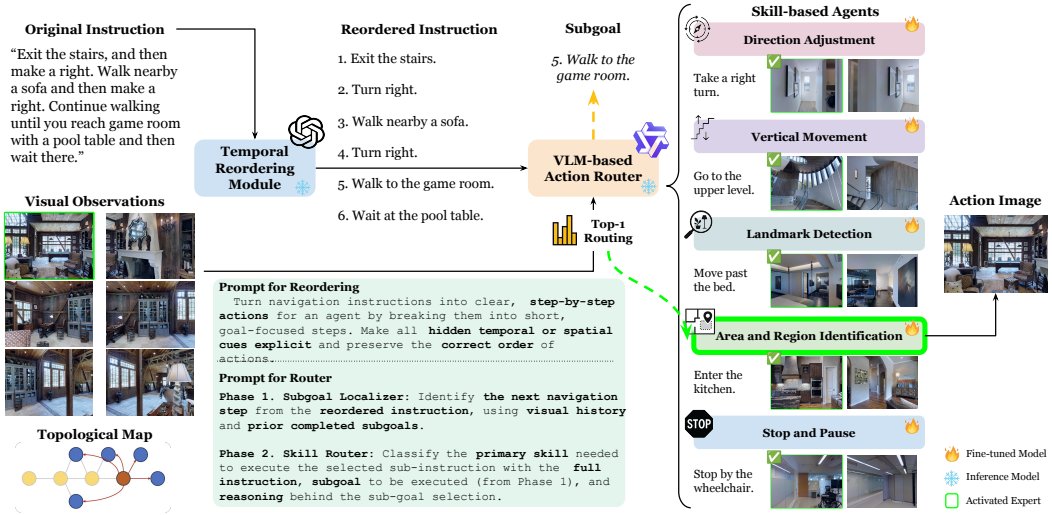


Figure 2: SkillNav Architecture. SkillNav takes visual observations, original instructions and the topological map as input. A temporal reordering module first leverages an LLM to reorder instructions into structured action goals. Subsequently, a VLM-based action router localizes the current focused sub-goal and dynamically selects the most suitable skill-based agent. For each skill, we construct specialized instruction-visual observation datasets for targeted skill learning.

4.1 Skill Taxonomy

We use the defined skills in NavNuances [39] 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 skills. In this work, we extend the initially proposed skill categories and provide solutions for acquiring them by the skill-based agents. We adopt four frequently observed atomic skills from NavNuances, **Direction Adjustment**, **Vertical Movement**, **Landmark Detection**, and **Area and Region Identification**. 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 [2], we perform a keyword-based analysis of the navigation instructions as shown in Figure 4 in Appendix A. 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

To enable the training of skill-specialized agents, we construct a set of synthetic datasets in which each trajectory–instruction pair is specifically designed to emphasize a single navigation skill.

Table 1: Statistics of skill-specific synthetic datasets and existing VLN training datasets.

Dataset	# Instr	# Vocab	Instr Len
R2R	14,039	4,597	26.28
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

We begin with a random starting node in the Matterport3D [4] environment and sample diverse navigation paths through graph traversal. For each skill, we define filtering heuristics to select trajectories where this skill is the primary factor for successful navigation. For instance, we emphasize frequent orientation changes or non-trivial turning sequences for the Direction Adjustment category. We explain detailed primary factors of skill-based trajectory generation in Appendix A. Each selected trajectory consists of a sequence of panoramic observations. Besides, we constrain trajectory length to 4–7 steps to keep the difficulty and temporal context comparable to human-annotated VLN data. The analysis of path length constraints used during trajectory generation to ensure temporal consistency and alignment with existing VLN datasets are provided in Figure 5 in Appendix B.

To generate skill-focused instruction, we feed the observation sequence of each candidate trajectory into GPT-4o [27] with a structured prompt. We design the prompts such that the generated instructions preserve the general linguistic quality of real VLN datasets, including comparable sentence length, vocabulary diversity, and fluency, while emphasizing the content toward the targeted skill. This is achieved by providing GPT-4o with explicit skill-focused cues during generation, encouraging, for example, frequent references to orientation change for the Direction Adjustment skill or strong emphasis on landmark description for the Landmark Detection skill. For each skill, we synthesize N such trajectory–instruction pairs, forming six separate datasets. A summary of dataset statistics is provided in Table 1.

The training of each skill-based agent is conducted in two stages. In the first stage, we fine-tune the pre-trained DUET model using the original R2R training dataset, the ScaleVLN augmentation data [37], and our Temporal Synthetic dataset to obtain a strong, skill-agnostic backbone. We provide the analysis of the effectiveness of the Temporal Order Planning agent in Appendix C. In the second stage, this backbone is further fine-tuned on a skill-specific synthetic dataset to obtain a specialized agent in the targeted skill. Following this process, we obtain five specialized skill-based agents: the Direction Adjustment agent (π_{da}), Vertical Movement agent (π_{vm}), Stop and Pause agent (π_{sp}), Landmark Detection agent (π_{ld}), and Area and Region Identification agent (π_{ar}). We denotes the predefined set of five skill-based agents as $\mathcal{S} = \{\pi_{\text{da}}, \pi_{\text{vm}}, \pi_{\text{sp}}, \pi_{\text{ld}}, \pi_{\text{ar}}\}$.

4.3 SkillNav Framework

After training specialized agents for different navigation skills, we build our SkillNav framework. SkillNav first employs a temporal reordering module to generate chronologically ordered execution plans. Then, we introduce a VLM-based action router to accurately identify the current subgoal and dynamically select the corresponding skill-based agent to choose the appropriate action.

4.3.1 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

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

Table 2: Performance comparison on R2R and GSA-R2R benchmarks. † indicates large-scale data augmentation. SRDF performs best on R2R due to extensive pretraining on data that mimics R2R-style instructions; however, it struggles to generalize effectively to the GSA-R2R dataset.

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) [5]	1	5.63	58	44	35	–	–	–	–	34	30	25	23	25	23
NavCoT (LLaMA2) [24]	2	6.26	42	34	29	–	–	–	–	37	35	29	26	29	26
NavGPT-2 (FlanT5-5B) [55]	3	3.13	81	72	61	3.33	80	72	60	58	45	48	35	**57**	<u>43</u>
NaviLLM (Vicuna-7B) [52]	4	3.51	–	67	59	3.71	–	68	60	–	–	–	–	–	–
Supervised VLN															
HAMT [8]	5	2.29	–	66	61	3.93	72	65	60	48	44	42	38	34	30
DUET [10]	6	3.31	81	72	60	3.65	76	69	59	58	47	48	37	40	30
BEVBERT [1]	7	2.81	84	75	64	3.13	81	73	62	58	45	46	35	39	27
GR-DUET [15]	8	–	–	–	–	–	–	–	–	69	64	57	52	48	<u>43</u>
ScaleVLN [37] †	9	2.34	87	79	70	2.73	<u>84</u>	77	68	<u>78</u>	<u>67</u>	<u>69</u>	<u>57</u>	<u>55</u>	<u>43</u>
SRDF [38] †	10	1.83	**89**	**84**	**78**	1.88	**88**	**84**	**77**	71	63	59	49	52	<u>43</u>
Mixture of Skill-based VLN															
SAME† [56]	11	2.73	–	76	66	3.03	–	74	64	–	–	–	–	–	–
SkillNav† (ours)	12	1.97	**89**	<u>83</u>	<u>77</u>	2.53	83	<u>78</u>	<u>70</u>	**79**	**69**	**72**	**61**	**57**	**48**

4.3.2 VLM-based Action Router

To coordinate skill-based agents during navigation, we introduce an Action Router that dynamically selects the most suitable agent at each time step. Inspired by LLM-based planning systems such as LLM-Planner [33], Mic [28], and A2Nav [7], our router leverages a large VLM model (e.g., GPT-4o [27], Qwen2.5-VL-7B-Instruct [3]) in a zero-shot in-context fashion. We structure the routing process into two distinct reasoning phases:

Phase 1: 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 2: 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^*. Besides, 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 1 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 Equation 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 [2], especially the unseen split of validation (Val Unseen) and test (Test Unseen) splits. R2R is a commonly-used benchmark in VLN consisting of panoramic RGB-D scans from the Matterport3D [4] simulator and providing crowd-sourced instructions paired with navigation paths. Moreover, we evaluate the generalization ability of SkillNav on GSA-R2R [15] which includes residential (R) and non-residential (N) scenes (*e.g.*, shops, restaurants, and museums) from Habitat-Matterport3D [30], and diverse instruction styles with role-specific dialogues (*e.g.*, travel guides (Scene) beyond the basic style of R2R (Basic).

Table 3: 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.

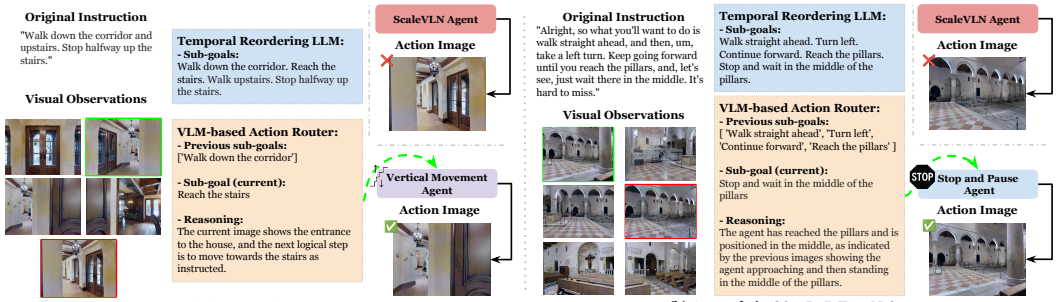
	Methods	DC	VM			LR	RR	
		SR	SR	OSR	SPL	SR	SR	OSR
VLN Agents	ScaleVLN [37]	68.39	81.76	88.82	76.34	28.32	82.91	95.27
	SRDF [38]	59.93	82.94	**91.18**	80.98	26.28	77.09	94.55
Skill-based Agents	Direction Adjustment	**70.81**	81.76	**91.18**	76.28	31.39	81.82	94.91
	Vertical Movement	70.68	**87.65**	89.41	**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**

Evaluation Metrics. We use the standard metrics to evaluate the navigation performance [2, 51]: (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 utilize CLIP-B/16 [29] as the visual backbone and BERT-base-uncased [13] as the language backbone within our DUET-based skill agents. During the skill training, we fine-tune the DUET pre-trained 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. In our SkillNav LLM-based architecture, we adopt GPT-4o [27] as the Temporal Reordering module due to its superior instruction-following capabilities and employ Qwen2.5-VL-7B-Instruct [3] as the action router because of its strong multi-modal alignment and reasoning abilities. All inferences with the action router are performed using in-context prompting.

5.1 Main Results

As shown in Table 2, SkillNav achieves strong overall performance across both R2R datasets and demonstrates robust generalization on GSA-R2R, outperforming most fine-tuned and LLM-based agents. On the R2R unseen environments, SkillNav (Method #12) achieves 83% SR and 77% SPL, ranking second highest after SRDF (Method #10). While SRDF achieves the highest performance on R2R Test-Unseen, this can be largely attributed to its pretraining on large-scale data that closely follows R2R-style instruction patterns. However, this reliance weakens its generalization ability, leading to a 13% and 5% SR drop on GSA-R2R Test-N-Basic and Test-N-Scene, respectively. SRDF requires additional tuning to remain competitive when transferred to new environments or novel instruction styles. In contrast, SkillNav is trained only on R2R and synthetic skill-specific data, yet achieves strong cross-dataset generalization without any retraining. Additionally, SkillNav also demonstrates SOTA generalization performance in GSA-R2R, ranking 1st in SPL across all GSA-R2R splits and demonstrating its ability to predict more efficient and precise navigation trajectories. Notably, on Test-N-Scene, which combines non-residential environments with more complex and role-specific instructions, SkillNav matches the best SR tied with NavGPT-2 (Method #3), while significantly outperforming it in SPL. NavGPT-2 benefits from fine-tuning on FlanT5-XXL [11], which likely enhances its ability to interpret stylized instructions. However, its lower SPL reveals inefficiencies in path planning and execution. While LLMs can help parse diverse instructions, they often introduce noise or lose critical spatial details when translating, limiting their effectiveness in downstream navigation tasks. This highlights the need for tightly integrated skill reasoning and grounded visual understanding, beyond language interpretation alone.



(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 4: Ablation results on GSA-R2R across residential (R) and non-residential (N) scenarios with varying instruction styles (Basic and Scene). Reorder: ✗ = LLM-guided Temporal Reordering disabled, ✔ = enabled. Router: Random = randomly select skill-based agents without utilizing action router; 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
✗	Random	1	78.39	67.46	70.93	59.71	54.61	43.17
✗	Qwen	2	78.42	67.80	71.01	59.62	55.46	45.43
✔	GLM	3	78.60	67.93	71.13	59.73	**56.80**	46.51
✔	Qwen	4	**78.83**	**68.88**	**71.58**	**61.34**	56.66	**47.96**

5.2 Ablation Study

Skill Evaluation. To further probe the capabilities of our skill-based agents, we have a fine-grained evaluation on the NavNuances, which categorizes navigation instructions into four atomic skills: (1) Direction Change (DC), (2) Vertical Movement (VM), (3) Landmark Recognition (LR), and (4) Region Recognition (RR). These subsets isolate specific reasoning capabilities and allow us to assess each agent's specialization. As shown in Table 3, each skill-based agent in SkillNav excels in its corresponding category. The Vertical Movement agent achieves the highest SR (87.65%) and SPL (83.83%) on VM, while the Direction Adjustment agent leads in DC with an SR of 70.81%. The Landmark Detection agent performs best in LR with 31.53% SR, and the Area and Region Identification agent reaches 85.09% SR on RR. We report the effectiveness of the Stop and Pause agent in Appendix D. These results validate our skill-based training and data augmentation strategy, confirming that targeted supervision fosters functional specialization that outperforms generalist VLN baselines in isolated skill settings.

Temporal Reordering Module. We conduct an ablation study to evaluate SkillNav's two key components: the LLM-guided Temporal Reordering module and the VLM-based action router. The results, shown in Table 4, are reported across GSA-R2R splits, covering both residential (R) and non-residential (N) environments with varying instruction styles. First, we evaluate the effectiveness of the temporal reordering module. As shown in rows #2 and #4, when using the same router (Qwen2.5-VL-7B-Instruct), incorporating the reordering module consistently improves performance across all benchmarks. Notably, in Test-N-Basic, SPL increases $+1.72\%$, demonstrating that temporally structured subgoals offer clearer guidance for effective skill selection.

Action Router. To evaluate the effectiveness of our action router, we compare the performance of randomly selected skills without a router (row #1) against our proposed Qwen router. The observed improvements in both SR and SPL metrics clearly indicate the router's effectiveness: specifically, Test-N-Scene SR increases from 54.61% to 55.46%, and SPL rises notably from 43.17% to 45.43%. These results confirm that our VLM-based router effectively selects appropriate skills even in the absence of temporal structuring. We further examine the significance of router selection by comparing rows #3 and #4, where the instruction reordering is fixed, and only the router model varies. Qwen2.5-VL-7B-Instruct consistently achieves superior SPL across all splits, particularly notable in Test-N-Scene (47.96% vs. 46.51%), underscoring its enhanced visual grounding capabilities

compared to GLM-4.1V-9B-Thinking [35]. This emphasizes that high-quality vision-language representations are essential for effective skill routing, and the primary driver of success in VLN appears to be the skill-based agents.

5.3 Efficiency Analysis

Training Cost. Fine-tuning five skills on the Temporal Order Planning agent with R2R and synthetic skill-specific datasets requires approximately $3,329$ minutes ($\sim$ 55.5 hours) in total. For comparison, SRDF training on R2R with larger data augmentation takes $2,521$ minutes ($\sim$ 42 hours), suggesting that SkillNav's 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 5. SkillNav introduces overhead due to its Temporal Reordering LLM and VLM-based action router, reaching 0.49 throughput on Test-N-Basic of GSA-R2R, which is roughly $50\times$ slower than ScaleVLN but still nearly $20\times$ faster than MapGPT. The Random variant, despite sharing the DUET as the backbone and selecting only one DUET for action prediction, is $4.3\times$ slower than ScaleVLN due to the per-observation skill selection overhead that prevents batch inference. 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.

Table 5: 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			
Random (ours)	Test-R-Basic	$2,223.4$	6.48
	Test-N-Basic	$1,507.9$	5.97
SkillNav (ours)	Test-R-Basic	$\sim 27,000$	0.54
	Test-N-Basic	$\sim 18,360$	0.49

5.4 Qualitative Examples

Figure 3 shows two qualitative examples highlighting SkillNav's capability to dynamically select the appropriate skill at each navigation step. These examples illustrate the effectiveness of our approach in reordering temporal action plans, accurately identifying the currently focused subgoal via the router, and subsequently selecting the correct action. Specifically, in Figure 3 (a), the router correctly reasons that the agent has reached the target pillars and decides it is time to stop, resulting in the agent appropriately choosing the stop action at the view containing the pillars. Similarly, in Figure 3 (b), the router identifies the need to move toward the stairs and accordingly selects the vertical movement skill. Overall, SkillNav successfully interprets diverse instruction styles and performs robustly across both residential and non-residential scenes.

6 Conclusion

We introduce SkillNav, a VLN agent that combines skill-based learning with VLM-based routing to dynamically select the most suitable actions based on the decision of the most relevant expert. We evaluate SkillNav on R2R to show strong navigation performance and demonstrate its generalization capabilities on the GSA-R2R dataset. While the utilization of LLM for temporal reordering and VLM for routing introduces computational overhead, SkillNav is more efficient than relying solely on LLMs or VLMs for navigation and achieves stronger performance than supervised VLN agents by exploiting both paradigms. Our framework provides a novel and interpretable approach that advances compositional reasoning and generalization for the VLN research community.

References

[1] D. An, Y. Qi, Y. Li, Y. Huang, L. Wang, T. Tan, and J. Shao. Bevbert: Multimodal map pre-training for language-guided navigation. Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023.

[2] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel. 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, 2018.

[3] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, H. Zhong, Y. Zhu, M. Yang, Z. Li, J. Wan, P. Wang, W. Ding, Z. Fu, Y. Xu, J. Ye, X. Zhang, T. Xie, Z. Cheng, H. Zhang, Z. Yang, H. Xu, and J. Lin. Qwen2.5-vl technical report. arXiv preprint arXiv:2502.13923, 2025.

[4] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. International Conference on 3D Vision (3DV), 2017.

[5] J. Chen, B. Lin, R. Xu, Z. Chai, X. Liang, and K.-Y. K. Wong. MapGPT: Map-Guided Prompting with Adaptive Path Planning for Vision-and-Language Navigation, June 2024. URL http://arxiv.org/abs/2401.07314. arXiv:2401.07314 [cs].

[6] J. C.-Y. Chen, S. Yun, E. Stengel-Eskin, T. Chen, and M. Bansal. Symbolic Mixture-of-Experts: Adaptive Skill-based Routing for Heterogeneous Reasoning, Mar. 2025. URL http://arxiv.org/abs/2503.05641. arXiv:2503.05641 [cs].

[7] P. Chen, X. Sun, H. Zhi, R. Zeng, T. H. Li, G. Liu, M. Tan, and C. Gan. A2nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. CoRR, abs/2308.07997, 2023. URL https://arxiv.org/abs/2308.07997.

[8] S. Chen, P.-L. Guhur, C. Schmid, and I. Laptev. History aware multimodal transformer for vision-and-language navigation. Advances in neural information processing systems, 34:5834–5847, 2021.

[9] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In ECCV, 2022.

[10] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev. Think Global, Act Local: Dual-scale Graph Transformer for Vision-and-Language Navigation, Feb. 2022. URL http://arxiv.org/abs/2202.11742. arXiv:2202.11742 [cs].

[11] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Zhao, Y. Huang, A. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. Scaling instruction-finetuned language models, 2022. URL https://arxiv.org/abs/2210.11416.

[12] D. Dai, C. Deng, C. Zhao, R. Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. arXiv preprint arXiv:2401.06066, 2024.

[13] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. CoRR, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

[14] W. Hao, C. Li, X. Li, L. Carin, and J. Gao. 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, 2020.

[15] H. Hong, Y. Qiao, S. Wang, J. Liu, and Q. Wu. General scene adaptation for vision-and-language navigation. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=2oKkQTyfz7.

[16] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould. A Recurrent Vision-and-Language BERT for Navigation, Mar. 2021. URL http://arxiv.org/abs/2011.13922. arXiv:2011.13922 [cs].

[17] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. Neural computation, 3(1):79–87, 1991.

[18] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. Bou Hanna, F. Bressand, et al. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024.

[19] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. Neural computation, 6(2):181–214, 1994.

[20] D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In 9th International Conference on Learning Representations, ICLR 2021, 2021. URL: https://openreview.net/forum?id=qrwe7XHTmYb.

[21] C. Li, Z. Gan, Z. Yang, J. Yang, L. Li, L. Wang, J. Gao, et al. Multimodal foundation models: From specialists to general-purpose assistants. Foundations and Trends® in Computer Graphics and Vision, 16(1-2):1–214, 2024.

[22] J. Li and M. Bansal. Panogen: Text-conditioned panoramic environment generation for vision-and-language navigation. In Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS), 2023. URL https://openreview.net/forum?id=cNOb16QQEH.

[23] J. Li, H. Tan, and M. Bansal. 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, 2022.

[24] B. Lin, Y. Nie, Z. Wei, J. Chen, S. Ma, J. Han, H. Xu, X. Chang, and X. Liang. NavCoT: Boosting LLM-Based Vision-and-Language Navigation via Learning Disentangled Reasoning, Mar. 2024. URL https://arxiv.org/abs/2403.07376v1.

[25] C. Liu, F. Zhu, X. Chang, X. Liang, Z. Ge, and Y.-D. Shen. Vision-language navigation with random environmental mixup. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 1644–1654, 2021.

[26] Y. Long, X. Li, W. Cai, and H. Dong. Discuss before moving: Visual language navigation via multi-expert discussions. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 17380–17387. IEEE, 2024.

[27] OpenAI. Hello gpt-4o, 2024. URL https://openai.com/index/hello-gpt-4o/.

[28] Y. Qiao, Y. Qi, Z. Yu, J. Liu, and Q. Wu. March in chat: Interactive prompting for remote embodied referring expression. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15758–15767, 2023.

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In International conference on machine learning, pages 8748–8763. PmLR, 2021.

[30] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. M. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra. 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, 2021. URL https://arxiv.org/abs/2109.08238.

[31] C. Riquelme, J. Puigcerver, B. Mustafa, M. Neumann, R. Jenatton, A. Susano Pinto, D. Keysers, and N. Houlsby. Scaling vision with sparse mixture of experts. In Advances in Neural Information Processing Systems 34 (NeurIPS), pages 8583–8595, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/48237d9f2dea8c74c2a72126cf63d933-Abstract.html.

[32] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538, 2017.

[33] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su. 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), October 2023.

[34] H. Tan, L. Yu, and M. Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. arXiv preprint arXiv:1904.04195, 2019.

[35] G.-V. Team, W. Hong, W. Yu, X. Gu, G. Wang, G. Gan, H. Tang, J. Cheng, J. Qi, J. Ji, L. Pan, S. Duan, W. Wang, Y. Wang, Y. Cheng, Z. He, Z. Su, Z. Yang, Z. Pan, A. Zeng, B. Wang, B. Shi, C. Pang, C. Zhang, D. Yin, F. Yang, G. Chen, J. Xu, J. Chen, J. Chen, J. Chen, J. Lin, J. Wang, J. Chen, L. Lei, L. Gong, L. Pan, M. Zhang, Q. Zheng, S. Yang, S. Zhong, S. Huang, S. Zhao, S. Xue, S. Tu, S. Meng, T. Zhang, T. Luo, T. Hao, W. Li, W. Jia, X. Lyu, X. Huang, Y. Wang, Y. Xue, Y. Wang, Y. An, Y. Du, Y. Shi, Y. Huang, Y. Niu, Y. Wang, Y. Yue, Y. Li, Y. Zhang, Y. Zhang, Z. Du, Z. Hou, Z. Xue, Z. Du, Z. Wang, P. Zhang, D. Liu, B. Xu, J. Li, M. Huang, Y. Dong, and J. Tang. Glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning, 2025. URL https://arxiv.org/abs/2507.01006.

[36] J. Wang, J. Wang, B. Athiwaratkun, C. Zhang, and J. Zou. Mixture-of-agents enhances large language model capabilities. arXiv preprint arXiv:2406.04692, 2024.

[37] Z. Wang, J. Li, Y. Hong, Y. Wang, Q. Wu, M. Bansal, S. Gould, H. Tan, and Y. Qiao. Scaling data generation in vision-and-language navigation. In Proceedings of the IEEE/CVF international conference on computer vision, pages 12009–12020, 2023.

[38] Z. Wang, J. Li, Y. Hong, S. Li, K. Li, S. Yu, Y. Wang, Y. Qiao, Y. Wang, M. Bansal, and L. Wang. Bootstrapping Language-Guided Navigation Learning with Self-Refining Data Flywheel, Dec. 2024. URL http://arxiv.org/abs/2412.08467. arXiv:2412.08467 [cs].

[39] Z. Wang, M. Wu, Y. Cao, Y. Ma, M. Chen, and T. Tuytelaars. Navigating the nuances: A fine-grained evaluation of vision-language navigation. In Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, editors, Findings of the Association for Computational Linguistics: EMNLP 2024, pages 4681–4704, Miami, Florida, USA, Nov. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.269. URL https://aclanthology.org/2024.findings-emnlp.269/.

[40] T. Xiao and J. Zhu. Foundations of large language models. arXiv preprint arXiv:2501.09223, 2025.

[41] F. Xue, Z. Zheng, Y. Fu, J. Ni, Z. Zheng, W. Zhou, and Y. You. Openmoe: An early effort on open mixture-of-experts language models. arXiv preprint arXiv:2402.01739, 2024.

[42] S. Yu, Y. Zhang, Z. Wang, J. Yoon, and M. Bansal. Mexa: Towards general multimodal reasoning with dynamic multi-expert aggregation. In Findings of the Association for Computational Linguistics: EMNLP 2025. Association for Computational Linguistics, 2025.

[43] S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. IEEE Transactions on Neural Networks and Learning Systems, 23(8):1177–1193, 2012. doi: 10.1109/TNNLS.2012.2200299.

[44] J. Zhang, J. Huang, S. Jin, and S. Lu. Vision-language models for vision tasks: A survey. IEEE transactions on pattern analysis and machine intelligence, 46(8):5625–5644, 2024.

[45] S. Zhang, Y. Qiao, Q. Wang, L. Guo, Z. Wei, and J. Liu. Flexvln: Flexible adaptation for diverse vision-and-language navigation tasks. arXiv preprint arXiv:2503.13966, 2025.

[46] Y. Zhang and P. Kordjamshidi. Vln-trans: Translator for the vision and language navigation agent. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 13219–13233, Toronto, Canada, jul 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.737. URL https://aclanthology.org/2023.acl-long.737/.

[47] Y. Zhang, Q. Guo, and P. Kordjamshidi. Navhint: Vision and language navigation agent with a hint generator. In Y. Graham and M. Purver, editors, Findings of the Association for Computational Linguistics: EACL 2024, pages 92–103, St. Julian's, Malta, mar 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.findings-eacl.7/.

[48] Y. Zhang, Z. Ma, J. Li, Y. Qiao, Z. Wang, J. Chai, Q. Wu, M. Bansal, and P. Kordjamshidi. Vision-and-Language Navigation Today and Tomorrow: A Survey in the Era of Foundation Models, July 2024. URL http://arxiv.org/abs/2407.07035. arXiv:2407.07035 [cs].

[49] Y. Zhang, Z. Xu, Y. Shen, P. Kordjamshidi, and L. Huang. SPARTUN3d: Situated spatial understanding of 3d world in large language model. In The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=FGMkSL8NR0.

[50] Z. Zhang, Y. Lin, Z. Liu, P. Li, M. Sun, and J. Zhou. Moefication: Conditional computation of transformer models for efficient inference. arXiv preprint arXiv:2110.01786, 2021.

[51] C. Zhao, Y. Qi, and Q. Wu. Mind the Gap: Improving Success Rate of Vision-and-Language Navigation by Revisiting Oracle Success Routes, Aug. 2023. URL http://arxiv.org/abs/2308.03244. arXiv:2308.03244 [cs].

[52] D. Zheng, S. Huang, L. Zhao, Y. Zhong, and L. Wang. Towards learning a generalist model for embodied navigation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13624–13634, 2024.

[53] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, et al. A comprehensive survey on pretrained foundation models: A history from bert to chatgpt. International Journal of Machine Learning and Cybernetics, pages 1–65, 2024.

[54] G. Zhou, Y. Hong, and Q. Wu. NavGPT: Explicit Reasoning in Vision-and-Language Navigation with Large Language Models, Oct. 2023. URL http://arxiv.org/abs/2305.16986. arXiv:2305.16986 [cs].

[55] G. Zhou, Y. Hong, Z. Wang, X. E. Wang, and Q. Wu. NavGPT-2: Unleashing Navigational Reasoning Capability for Large Vision-Language Models, July 2024. URL http://arxiv.org/abs/2407.12366. arXiv:2407.12366 [cs].

[56] G. Zhou, Y. Hong, Z. Wang, C. Zhao, M. Bansal, and Q. Wu. Same: Learning generic language-guided visual navigation with state-adaptive mixture of experts. arXiv preprint arXiv:2412.05552, 2024.

[57] S. Zuo, X. Liu, J. Jiao, Y. J. Kim, H. Hassan, R. Zhang, J. Gao, and T. Zhao. Taming sparsely activated transformer with stochastic experts. In International Conference on Learning Representations, 2022. URL: https://openreview.net/forum?id=B72HXs80q4.

Appendix

A 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 → 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.

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.

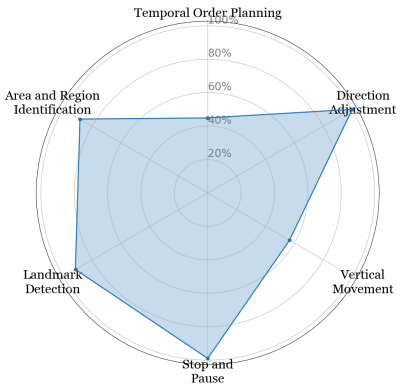


Figure 4: Distribution of instructions in the R2R dataset categorized by the proposed skill taxonomy.

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.

B 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 quite less instructions with a 4-step trajectory.

C 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.

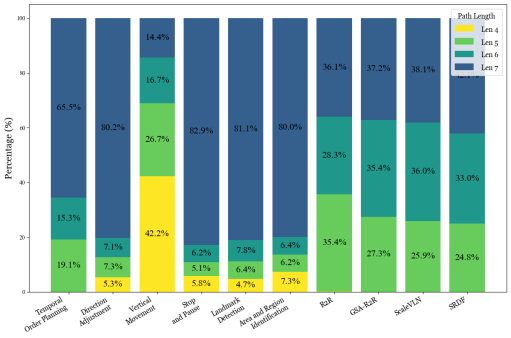


Figure 5: The statistics of the path length of our synthetic datasets compared with existing VLN datasets. The R2R, ScaleVLN, SRDF datasets, and our 6 skill-specific datasets are all for training, while only GSA-R2R is for evaluation.

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 6. 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 6, our Temporal Synthetic Data improves navigation in failure cases where prior methods such as ScaleVLN struggle.

Table 6: 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.

Environment	Metric	ScaleVLN	Temporal DUET
Conditional Immediacy	SR	**84.29**	**88.57**
	SPL	**76.29**	**82.18**
Bounded Duration	SR	76.27	**84.18**
	SPL	67.45	74.90
Forward Sequential	SR	**79.53**	**85.83**
	SPL	68.92	**76.93**
Backward Sequential	SR	74.29	**88.57**
	SPL	66.97	**81.72**

D 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 3), suggesting that effective stopping is a foundational capability that influences the success of diverse navigation behaviors.

E Efficiency Analysis

All experiments in efficiency analysis in Section 5.3 run on NVIDIA A6000. For the inference cost in Table 5, 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.

F LLM Usage

We used LLM-based tools for polishing grammar and aiding the writing. In addition, we uti-

lize 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 4.3.1 and 4.3.2.

G 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, the approach depends on synthetic, skill-specific datasets generated via prompting, which may introduce distributional biases compared to human-authored instructions. We do a human evaluation on 20 cases with action routing, and the result shows 100% accuracy. This means with high confidence, the true accuracy is at least 84% on R2R Val Unseen.

H LLM and VLM Prompts

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

H.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 the two 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. .

H.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.

H.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 claims 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.

```
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 Synthsis

```
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 Synthsis

```
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

NeurIPS Paper Checklist

1. **Claims**
   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: See the last part in Section 1.
   Guidelines:
   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.
2. **Limitations**
   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: Section G
   Guidelines:
   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed cap-

tions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**
   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?
   Answer: [NA]
   Justification: The paper does not present formal theorems or proofs. Its contributions are methodological and empirical. While the methodology is carefully described and justified with ablations and qualitative examples, there are no explicit theoretical results requiring assumptions or formal proofs.
   Guidelines:
   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**
   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?
   Answer: [Yes]
   Justification: Section 5, A and B
   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**
   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?
   Answer: [Yes]
   Justification: The data and code are included in supplemental material.
   Guidelines:
   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.
6. **Experimental setting/details**
   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?
   Answer: [Yes]
   Justification: Section 5
   Guidelines:
   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.
7. **Experiment statistical significance**
   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?
   Answer: [No]
   Justification: Instead, it supports its main claims through extensive ablation studies and qualitative examples, which demonstrate the contribution of individ-

ual components and the robustness of SkillNav across benchmarks. While these analyses provide useful insights into performance variability, they do not substitute for error bars or formal statistical measures. Including such statistical reporting would have strengthened the empirical rigor and reproducibility of the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 5

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: It does not involve sensitive human subjects, private data, or practices that raise ethical concerns. The work appears to conform fully to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not explicitly include a Broader Impacts section. However, the contributions could have positive societal impact by enabling safer, more generalizable, and interpretable vision-and-language nav-

igation systems, which in turn may benefit assistive robotics and accessibility applications.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper introduces SkillNav and skill-specific synthetic datasets, but these do not carry the same high-risk misuse potential as large pretrained language models, generative image models, or scraped internet-scale datasets. The datasets are task-focused, controlled, and primarily intended for navigation research. Since the work poses no significant risks of malicious dual-use, safeguards are not discussed and are not required in this case.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: These assets are properly cited in the text and referenced in the bibliography, ensuring that the original creators receive credit. While the paper does not spell out individual license names, the datasets used are public benchmarks distributed under permis-

sive terms of use, and no evidence of license violations is present.
Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are repackaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**
    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?
    Answer: [Yes]
    Justification: The paper introduces SkillNav and accompanying synthetic datasets designed for skill-specific VLN training. These new assets are described in detail within Section A and Section B, including how they are generated, the types of skills covered, and how they are used for training and evaluation.
    Guidelines:
    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/-model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.
14. **Crowdsourcing and research with human subjects**
    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?
    Answer: [NA]
    Justification: The paper does not involve crowdsourcing or experiments with human subjects. All datasets used are synthetic or standard publicly available VLN datasets.
    Guidelines:
    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.
15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**
    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?
    Answer: [NA]
    Justification: The paper does not involve crowdsourcing nor research with human subjects.
    Guidelines:
    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB ap-

proval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: The LLMs are used as central components, which can be seen in Section 4.3.1 and Section 4.3.2.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.