ULN: Towards Underspecified Vision-and-Language Navigation

Weixi Feng Tsu-Jui Fu Yujie Lu William Yang Wang

UC Santa Barbara
{weixifeng, tsu-juifu, yujielu, william}@cs.ucsb.edu

Abstract

Vision-and-Language Navigation (VLN) is a task to guide an embodied agent moving to a target position using language instructions. Despite the significant performance improvement, the wide use of fine-grained instructions fails to characterize more practical linguistic variations in reality. To fill in this gap, we introduce a new setting, namely **U**nderspecified vision-and-**L**anguage **N**avigation (ULN), and associated evaluation datasets. ULN evaluates agents using multi-level underspecified instructions instead of purely fine-grained or coarse-grained, which is a more realistic and general setting. As a primary step toward ULN, we propose a VLN framework that consists of a classification module, a navigation agent, and an Exploitation-to-Exploration (E2E) module. Specifically, we propose to learn Granularity Specific Sub-networks (GSS) for the agent to ground multi-level instructions with minimal additional parameters. Then, our E2E module estimates grounding uncertainty and conducts multi-step lookahead exploration to improve the success rate further. Experimental results show that existing VLN models are still brittle to multi-level language underspecification. Our framework is more robust and outperforms the baselines on ULN by ∼10% relative success rate across all levels. [1]

1 Introduction

Vision-and-Language Navigation (VLN) allows a human user to command or instruct an embodied agent to reach target locations using verbal instructions. For this application to step out of curated datasets in real-world settings, the agents must generalize to many linguistic variations of human instructions. Despite significant progress in VLN datasets (Anderson et al., 2018b; Chen et al., 2019; Ku et al., 2020; Shridhar et al., 2020) and agent design (Fried et al., 2018; Li et al., 2021; Min et al., 2021), it remains a question whether existing models are generalized and robust enough to deal with all kinds of language variations.

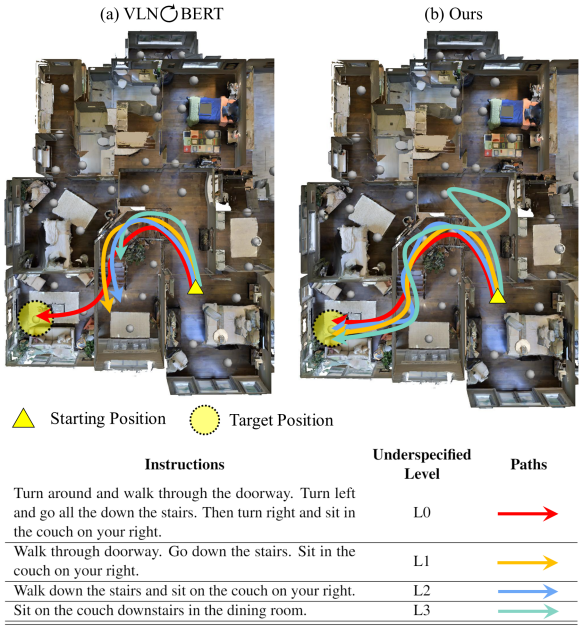

Figure 1: Navigation results of a baseline (left) and our VLN framework (right) with multi-level underspecified instructions (L_0-L_3). Trajectories are curved for demonstration. Note that the baseline stops early and fails to reach the target position with L_1-L_3. Our agent manages to reach the goal across all levels.

For the language input in an indoor environment, some datasets focus on long and detailed instructions with the route description at every step to achieve fine-grained language grounding (Anderson et al., 2018b; Ku et al., 2020) or long-horizon navigation (Jain et al., 2019; Zhu et al., 2020a). For instance, from Room-to-Room (R2R) (Anderson et al., 2018b), to Room-Across-Room (RxR) (Ku et al., 2020), the average instruction length increases from 29 to 129 words. Other datasets have coarse-grained instructions like REVERIE (Qi et al., 2020) or SOON (Zhu et al., 2021a). Agents are trained and evaluated on a single granu-

[1] Our code and data are available at https://github.com/weixi-feng/ULN.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 6394 - 6412
December 7-11, 2022

larity or one type of expression.

In contrast, we propose to evaluate VLN agents on multi-level granularity to better understand the behavior of embodied agents with respect to language variations. Our motivation is that users are inclined to give shorter instructions instead of detailed route descriptions because 1) users are not omniscient observers who follow the route and describe it step by step for the agent; 2) shorter instructions are more practical, reproducible, and efficient from a user's perspective. 3) users tend to underspecify commands in familiar environments like personal households. Therefore, we propose a new setting, namely **U**nderspecified vision-and-**L**anguage **N**avigation (ULN) and associated evaluation datasets on top of R2R, namely R2R-ULN to address these issues. R2R-ULN contains underspecified instructions where route descriptions are successively removed from the original instructions. Each long R2R instruction corresponds to three shortened and rephrased instructions belonging to different levels, which preserves partial alignment but also introduces variances.

As shown in Fig. 1, the goal of ULN is to facilitate the development of a generalized VLN design that achieves balanced performance across all granularity levels. As a primary step toward ULN, we propose a modular VLN framework that consists of an instruction classification module, a navigational agent, and an Exploitation-to-Exploration (E2E) module. The classification module first classifies the input instruction as high-level or low-level in granularity so that our agent can encode these two types accordingly. As for the agent, we propose to learn Granularity Specific Sub-networks (GSS) to handle both levels with minimally additional parameters. A sub-network, e.g., the text encoder, is trained for each level while other parameters are shared. Finally, the E2E module estimates the stepwise language grounding uncertainty and conducts multi-step lookahead exploration to rectify wrong decisions that originated from underspecified language.

Our VLN framework is model-agnostic and can be applied to many previous agents that follow a "encode-then-fuse" mechanism for language and visual inputs. We establish our framework based on two state-of-the-art (SOTA) VLN agents to demonstrate its effectiveness. We conduct extensive experiments to analyze the generalization of existing agents and our framework in ULN and the original datasets with fine-grained instructions. Our contribution is three-fold:

- We propose a novel setting named Underspecified vision-and-Language Navigation (ULN) to account for multi-level language variations for instructions. We collect a large-scale evaluation dataset R2R-ULN which consists of 9k validation and 4k testing instructions.
- We propose a VLN framework that consists of Granularity Specific Sub-networks (GSS) and an E2E module for navigation agents to handle both low-level and high-level instructions.
- Experiments show that achieving consistent performance across multi-level underspecification can be much more challenging to existing VLN agents. Furthermore, our VLN framework can improve the success rate by $\sim$10% relatively over the baselines and mitigate the performance gap across all levels.

2 Related Work

Language Variations for Multimodal Learning Natural language input has been an essential component of modern multimodal learning tasks to combine with other modalities such as vision (Antol et al., 2015; Johnson et al., 2017), speech (Alayrac et al., 2020) or gestures (Chen et al., 2021b). The effect of language variations has been studied in many vision-and-language (V&L) tasks (Bisk et al., 2016; Agrawal et al., 2018; Cirik et al., 2018; Zhu et al., 2020b; Lin et al., 2021). For instance, referring expression datasets (Kazemzadeh et al., 2014; Yu et al., 2016; Mao et al., 2016) contain multiple expressions for the same referring object. Ref-Adv (Akula et al., 2020) studies the robustness of referring expression models by switching word orders. In Visual Question Answering (VQA), Shah et al. (2019) discovers that VQA models are brittle to rephrased questions with the same meaning. As for VLN, we characterize the linguistic and compositional variations in rephrasing and dropping sub-instructions from a full instruction with complete route descriptions. We also define three different levels to formalize underspecification for navigational instructions.

VLN Datasets VLN has gained much attention (Gu et al., 2022) with emergence of various simulation environments and datasets (Chang et al., 2017;

Kolve et al., 2017; Jain et al., 2019; Nguyen and Daumé III, 2019; Koh et al., 2021). R2R (Anderson et al., 2018a) and RxR (Ku et al., 2020) provide fine-grained instructions which guide the agent in a step-wise manner. FG-R2R (Hong et al., 2020a) and Landmark-RxR (He et al., 2021) segments the instructions into action units and explicitly ground sub-instructions on visual observation. In contrast, REVERIE (Qi et al., 2020), and SOON (Zhu et al., 2021a) proposes to use referring expression with no guidance on intermediate steps that lead to the final destination. Compared to these datasets, ULN aims to build an agent that can generalize to multi-level granularity after training once, which is more practical for real-world applications.

Embodied Navigation Agents Learning to ground instructions on visual observations is one major problem for an agent to generalize to an unseen environment (Wang et al., 2019; Deng et al., 2020; Fu et al., 2020). Previous studies demonstrate significant improvement by data augmentations (Fried et al., 2018; Tan et al., 2019; Zhu et al., 2021b; Fang et al., 2022; Li et al., 2022), designing pre-training tasks (Hao et al., 2020; Chen et al., 2021a; Qiao et al., 2022) and decoding algorithms (Ma et al., 2019a; Ke et al., 2019; Ma et al., 2019b; Chen et al., 2022). For exploration-based methods, FAST (Ke et al., 2019) proposes a searching algorithm that allows the agent to backtrack to the most promising trajectory. SSM (Wang et al., 2021) memorizes local and global action spaces and estimates multiple scores for candidate nodes in the frontier of trajectories. Compared to E2E, Active VLN (Wang et al., 2020) is the most relevant work where they learn an additional policy for multi-step exploration. However, they define the reward function based on distances to target locations, while our uncertainty estimation is based on step-wise grounding mismatch. Our E2E module is also more efficient that has fewer parameters and low training complexity.

3 Underspecification in VLN

Our dataset construction is three-fold: We first obtain underspecified instructions by asking workers to simplify and rephrase the R2R instructions. Then, we validate that the goals are still reachable with underspecified instructions. Finally, we verify that instructions from R2R-ULN are preferred to R2R ones from a user's perspective, which proves the necessity of the ULN setting. We briefly describe definitions and our ULN dataset in this section with more details in Appendix A.

Level	Instructions
L_0	Turn around and go down the stairs. At the bottom turn slightly right and enter the room with the TV on the wall and a green table. Walk to the right past the TV. Stop at the door to the right facing into the bathroom. (*from R2R*)
L_1	Take the stairs to the bottom and enter the room with the TV on the wall and a green table. Walk past the TV. Stop at the door to facing into the bathroom. (*Redundancy Removed*)
L_2	Take the stairs to the bottom and enter the room a green table. Walk past the TV. Stop at the bathroom door. (*Partial Route Description*)
L_3	Go to the door of the bathroom next to the room with a green table. (*No Route Description*)

Table 1: Instruction examples from the R2R-ULN validation set. We mark removed words in red and rephrased words in blue in the next level.

3.1 Instruction Simplification

We formalize the instruction collection as a sentence simplification task and ask human annotators to remove details from the instructions progressively. Denoting the original R2R instructions as **Level 0** (L_0), annotators rewrite each L_0 into three different levels of underspecified instructions. We discover that some components in L_0 can be redundant or rarely used in indoor environments, such as "turn 45 degrees". Therefore, to obtain **Level 1** (L_1) from each L_0 instruction, annotators rewrite L_0 by removing any redundant part but keep most of the route description unchanged. Redundant components include but are not limited to repetition, excessive details, and directional phrases (See Table 1). As for **Level 2** (L_2), annotators remove one or two sub-instructions from L_1, creating a scenario where the users omit some details in commonplaces. We collect **Level 3** (L_3) instructions by giving destination information such as region label and floor level and ask annotators to write one sentence directly referring to the object or location of the destination point.

3.2 Instruction Verification

To ensure that the underspecified instructions provide a feasible performance upper bound for VLN agents, we have another group of annotators navigate in an interactive interface from R2R (Anderson

Level	R2R-ULN Val-Unseen			
	Instr. Following		Instr. Preference	
	SR↑	SPL↑	Practicality	Efficiency
L_0	86	72	-	-
L_1	82	68	55%	57%
L_2	82	65	63%	59%
L_3	75	58	68%	66%

Table 2: Human performance on R2R-ULN validation unseen in terms of Success Rate (SR) and SR weighted by Path Length (SPL), and human preference assessment results. The percentage denotes the ratio of participants selecting L_i over L_0.

et al., 2018b). As is shown in Table 2, annotators achieve a slightly degraded but promising success rate (SR) with L_3. SPL is a metric that normalizes SR over the path length. Therefore, the trade-off for maintaining high SR is to have more exploration steps, resulting in a much lower SPL value. We also verify that $L_i, i \in \{1, 2, 3\}$ are more practical and efficient choices than L_0. Table 2 shows that people prefer underspecified instructions over full instructions in both aspects, with an increasing trend as i increases to 3.

4 Method

4.1 Overview

In this section, we present our VLN framework for handling multi-level underspecified language inputs, which mainly consists of three modules (see Figure 2). Given a natural language instruction in a sequence of tokens, $\mathcal{W} = (w_1, \dots w_n)$,the classification module first categorize language input as low-level (L_0, L_1, L_2) or high-level (L_3) instructions. To handle these two types accordingly, GSS learns a sub-network, e.g., the text encoder, for each type while the other parameters are shared. At each step t, we denote the visual observation $\mathcal{O}_t = ([v_1; a_1], \dots, [v_N, a_N])$ with visual feature v_i and angle feature a_i of i-th view among all N views. The history contains a concatenation of all observations before tm $H_t = (\mathcal{O}_1, \dots, \mathcal{Q}_{t-1})$. Given $\mathcal{W}_t, \mathcal{H}_t, \mathcal{O}_t$, the GSS-based agent predicts a an action a_t by choosing a navigable viewpoint from $\mathcal{O}_t$. To overcome the reference misalignment issue, the E2E module predicts a sequence of uncertainty score $\mathcal{S} = (s_1, ..., s_T)$ and conducts multi-step exploration to collect future visual information.

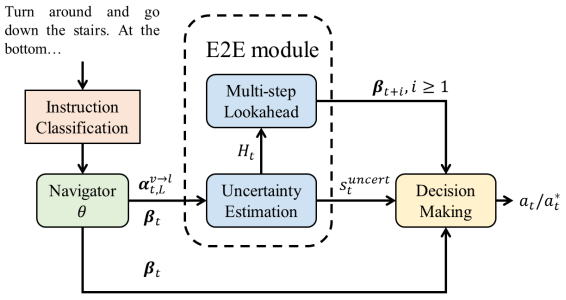

Figure 2: Our VLN framework with classification module, navigation agent, and E2E module.

4.2 Instruction Classification

VLN agents can operate in two different modes, fidelity-oriented or goal-oriented, depending on reward functions (Jain et al., 2019) or text inputs (Zhu et al., 2022). Agents trained on low-level granularity encounter performance degradation when applied to high-level ones, and vice versa. As is shown in Figure 2, we propose first to classify the text inputs into two granularities and then encode them independently in downstream modules. Our classification module contains an embedding layer, average pooling, and a fully-connected layer to output binary class predictions.

4.3 Navigation Agent

Base Agent We summarize the high-level framework of many transformer-based agents (Hao et al., 2020; Guhur et al., 2021; Moudgil et al., 2021) paramterized as θ as shown in Figure 2. Given the history $\mathcal{H}_t$, text $\mathcal{X}$, visual observation $\mathcal{O}_t$, the agent first encodes each modality input with encoders $f_{\text{hist}}, f_{\text{text}}, f_{\text{img}}$:

$$X = f_{\text{text}}(\mathcal{W}), \quad H_t = f_{\text{hist}}(\mathcal{H}_t), \\ O_t = f_{\text{img}}(\mathcal{O}_t) \tag{1}$$

HAMT (Chen et al., 2021a) applies ViT (Dosovitskiy et al., 2020) and a Panoramic Transformer to hierarchically encode $\mathcal{H}_t$ as a sequence of embeddings $H_t = (h_1, \dots, h_{t-1})$ while VLN↻BERT (Hong et al., 2021) encodes $\mathcal{H}_t$ as a state vector $H_t = h_t$. The embedding from each modality is then fed into a L-layer cross-modal transformer f_{cm}, and passed through a cross-attention first in each layer l:

$$\boldsymbol{\alpha}_{t,l}^{\text{v}\to\text{t}} = \frac{([H_{t,l}; O_{t,l}]W_l^{\text{query}})(X_{t,l}W_l^{\text{key}})^T}{\sqrt{d_h}} \tag{2}$$

where $\alpha_{t,l}^{\text{v}\to\text{t}}$ denotes the attention weights of history-visual concatenation on the language embeddings, d_h is the hidden dimension. We omit

the attention head index for simplicity. For VLN↻BERT, it concatenated state H_t with X_t instead. The prediction of a_t relies on either a two-layer FC network f_{action}, or the summation of attention weights of H_t on O_t over all heads:

$$\boldsymbol{\beta}_t^{\text{HAMT}} = f_{\text{action}}(O'_t \odot x'_{t,1}) \quad (3)$$

$$\boldsymbol{\beta}_t^{\text{VLN}\circlearrowright\text{BERT}} = \sum_{\text{head}} \frac{(H_{t,L}W_L^Q)(O_{t,L}W_L^K)^T}{\sqrt{d_h}} \quad (4)$$

$$a_t = \arg\max_c(\beta_{t,c}) \quad (5)$$

where O'_t, X'_t are the observation and language tokens output from f_{cm}.

Granularity Specific Sub-network Training an ensemble of agents or one agent with a mixture of levels can be inefficient or sub-optimal for ULN. Instead, we find a sub-network that influences the agent's navigation mode, as shown in Figure 3. We identify such sub-network by the following steps:

1. Train an agent θ_{l} with full instructions L_0 and a separate agent θ_{h} with the last sentence of L_0. Denote θ_{l} performance on L_3 as metric value $m_{\text{l}\to\text{h}}$.

2. For each of the sub-network, f_{text}, f_{img}, f_{hist}, and f_{cm} in θ_{h}, load its weights to θ_{l} and denote the performance on L_3 as $m^x_{\text{l}\to\text{h}}$ where $x \in \{\text{text, img, hist, cm}\}$ indicates the sub-network replaced.

3. Find the sub-network with the maximum gain in metric value on L_3 after replacement, i.e. $x^* = \arg\max_x(m^x_{\text{l}\to\text{h}} - m_{\text{l}\to\text{h}})$.

After identifying the critical sub-network f_{x*}, we train a new f_{x*} from scratch with the rest of the model parameters loaded from θ_l and kept frozen.

4.4 Exploitation to Exploration

Multi-level inputs introduces **Temporal Reference Misalignment (TRM)**. As the agent gradually shifts its attention to sub-instructions, it lacks a mechanism to ensure the attended text segments align with the visual observation transition. Consequently, after several steps, agents cannot correctly ground sub-instructions to visual features. To mitigate this issue, we propose an E2E module to estimate step-wise uncertainty and perform multi-step lookahead to skip the dilemma.

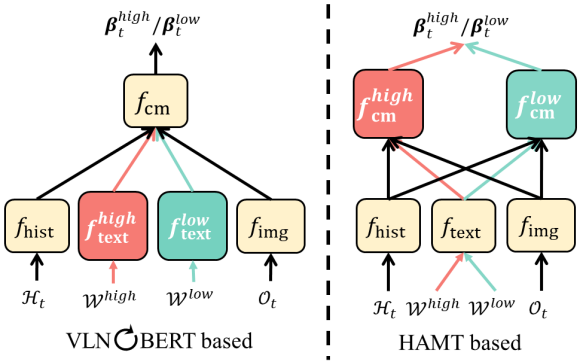

Figure 3: Granularity Specific Sub-networks of two different agents. We discover that the critical sub-network for VLN↻BERT is f_{text}, while f_{cm} for HAMT.

Uncertainty Estimation We evaluate the decision uncertainty at each step based on the **attention score distribution**. TRM changes the distribution of $\boldsymbol{\alpha}_{t,L}^{\text{v}\to\text{t}}, \boldsymbol{\beta}_t$ and makes them different from the distribution when full instructions are given. Therefore, the joint distribution of $\boldsymbol{\alpha}_{t,L}^{\text{v}\to\text{t}}, \boldsymbol{\beta}_t$ implies the degree of grounding uncertainty. We simply input the concatenation $\boldsymbol{\alpha}_{t,L}^{\text{v}\to\text{t}}, \boldsymbol{\beta}_t$ to a two-layer MLP f_{uncert} to learn the uncertainty score:

$$s_t^{\text{uncert}} = f_{\text{uncert}}([\boldsymbol{\alpha}_{t,L}^{\text{v}\to\text{t}}; \boldsymbol{\beta}_t]). \quad (6)$$

$s_t^{\text{uncert}} \in [0, 1]$ indicates whether the agent is confident for the decision. If s^{uncert} is greater than a threshold, the agent first explores the environment before the next step decisions.

Multi-Step Lookahead When score s_t^{uncert} indicates an uncertain decision, our system calculates the likelihood of success by exploration and re-evaluates the action logits. Specifically, the explorer moves K steps forward for each of the top C candidate actions at step t. Since unnormalized logits incorporate alignment between actions and instructions, we adopt the attention weights on visual candidates, i.e., $\boldsymbol{\beta}_t$. The new action probability estimation accounts for a weighted sum of the future logits sequence with a hyperparameter γ:

$$a_t^* = \arg\max_c \left[\beta_{t,c} + \sum_{i=1}^{K} \gamma^i \max_{c'}(\beta_{t+i,c'})\right]. \quad (7)$$

$\beta_{t,c}$ is the logit value for candidate c, and $\boldsymbol{\beta}_{t+i}$ are the logits after executing greedy action at step $t+i-1$. For parameter efficiency, we utilize the trained agent as the explorer. Our lookahead heuristic differentiates from Active VLN (Wang et al., 2020) as we explicitly quantify the misalignment

	Models	Training Set	R2R Val-Unseen				R2R-ULN Val-Unseen Level 3 (L_3)			
			TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
1	VLN↻BERT	R2R	12.09	4.10	61.4	55.6	13.20	7.41	32.7	29.1
2	VLN↻BERT	R2R-last	12.34	5.01	53.8	47.7	13.31	6.96	36.5	32.0
3	VLN↻BERT	R2R+R2R-last	11.78	4.28	59.2	53.5	12.81	7.12	35.7	31.2
4	Ours (w/o E2E)	R2R+R2R-last	12.09	**4.08**	**61.6**	**55.8**	13.30	**6.91**	**37.8**	**33.6**
5	HAMT	R2R	11.46	**3.62**	**66.2**	**61.5**	13.36	7.18	35.1	31.1
6	HAMT	R2R-last	11.57	4.55	57.1	52.4	13.51	6.84	37.6	33.9
7	HAMT	R2R+R2R-last	11.12	3.90	63.8	30.1	13.20	7.13	36.3	32.4
8	Ours (w/o E2E)	R2R, R2R-last	11.54	3.71	65.6	60.7	13.26	**6.75**	**38.8**	**34.9**

Table 3: Comparison of two baselines and our VLN framework with a classification module and GSS. We bold the best values and underline the second-best values.

by uncertainty estimation, while the latter depends on goal-based reward functions with implicit supervision. Our E2E module is also more efficient and stable as we spare the need to train a separate policy for exploration.

State Freeze The history encoding h_t serves as an important query to attend to on both visual and language domains. Due to the underspecified language input, the history-attended instruction advances the transition in the visual scenarios. Therefore, to calculate $\beta_{t+1,c'}$ in Equation 7, we shall utilize h_{t-1} instead of h_t to maintain the attention on the pending sub-instruction until alignment recovers.

5 Experiments

Dataset R2R (Anderson et al., 2018b) contains over 14k instructions for training, 1k for validation seen environments (Val-seen), 2.3k for validation unseen (Val-unseen). As for R2R-ULN, we select around 1600 longest instructions from the R2R validation set as L_0. We assign three different annotators to simplify each L_0 and filter low-quality samples. R2R-ULN includes 3132 instructions for Val-seen, 6714 for Val-unseen, and 4198 test unseen. We train the agent and other modules on R2R (train split) only and evaluate our system on R2R and R2R-ULN without re-training. To maintain the ratio between the training and validation set, we use a subset of 2k Val-seen and 4.5k Val-unseen from the full R2R-ULN for evaluation as default if not specified. We randomly sample 30% L1, 70% L2 and 100% L3 from the full set based on the preference results (Table 2). We also report the evaluation results in the full set in Appendix.

Evaluation Metrics We evaluate the navigation performance using the standard metrics of R2R: Trajectory Length (TL): the agent's navigation path in meters; Navigation Error (NE ↓): the average distance between the goal and agent's final location; Success Rate (SR ↑): the ratio of trials that end within 3 meters to the overall target trials; Success weighted by inverse Path Length (SPL ↑) (Anderson et al., 2018a).

Implementation Details We adopt full instructions as low-level samples (R2R) and the last sentences of instructions as high-level samples (R2R-last). We train the classification module and the agents with these two training sets. For uncertainty estimation training, we applied the chunking function (Hong et al., 2020a) to randomly drop sub-instructions from R2R and create pseudo-underspecified instructions as inputs to a trained agent. At each step, if the agent's action is different from the teacher's action, the uncertainty ground truth label is 1, else 0. During inference, an uncertainty score over 0.5 will initiate multi-step lookahead. We also limit the lookahead to at most three times for performance benefit. We explain this choice in Sec. 5.2.3.

We train the classification and the agent with a low-level text encoder for 300,000 iterations with a learning rate of 1e-2 and 1e-5. The agent is trained on a mix of imitation learning, and A2C (Mnih et al., 2016), the same as the baselines. Then we train the high-level text encoder with other parameters fixed for 10,000 iterations. Finally, for the E2E module, we train the uncertainty estimation network with a learning rate 1e-4 for 10 epochs. We directly adopt the trained agent as the explorer

	Methods	R2R-ULN Val-Seen				R2R-ULN Val-Unseen			
		TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
0	Human	-	-	-	-	14.97	2.94	77.4	61.7
	Greedy-Decoding Agents								
1	Speaker-Follower (Fried et al., 2018)	12.26	6.17	43.4	39.4	14.86	8.43	22.0	17.4
2	SMNA (Ma et al., 2019a)	12.07	5.95	46.5	41.0	15.42	7.98	23.1	16.3
3	EnvDrop (Tan et al., 2019)	9.14	6.96	37.7	36.8	8.74	8.26	24.6	23.3
4	PREVALENT (Hao et al., 2020)	10.24	6.32	45.8	44.2	11.91	7.28	33.8	31.1
5	RelGraph (Hong et al., 2020b)	9.19	7.10	36.6	35.3	9.26	7.79	29.0	27.4
	Exploration-based Agents								
6	FAST-Short (Ke et al., 2019)	14.70	5.52	52.3	46.3	22.89	6.78	36.8	26.4
7	Active VLN (Wang et al., 2020)	24.35	6.48	43.1	29.3	19.40	7.08	32.2	21.2
8	SSM (Wang et al., 2021)	20.13	6.14	49.4	40.9	26.64	6.70	39.8	26.1
9	VLN↻BERT (Hong et al., 2021)	12.29	5.80	47.9	44.2	13.00	6.47	39.3	35.0
10	Ours (VLN↻BERT-based, w/o E2E)	11.92	5.60	49.1	**45.7**	12.95	6.19	42.3	**37.7**
11	Ours (VLN↻BERT-based, w/ E2E)	19.28	**5.56**	**50.7**	38.1	23.02	**6.13**	**44.7**	29.7
12	HAMT (Chen et al., 2021a)	11.79	5.65	49.1	46.2	12.98	6.33	41.7	37.6
13	Ours (HAMT-based, w/o E2E)	12.44	5.36	52.3	**49.3**	12.91	6.10	43.5	**39.5**
14	Ours (HAMT-based, w/ E2E)	27.08	**5.21**	**54.2**	34.7	28.31	**6.05**	**44.6**	25.9

Table 4: Comparison of VLN agents on R2R-ULN validation set, including all three levels L_1, L_2, L_3. Rows 1-5 are results from greedy-decoding agents without back-tracking. Rows 6-8 are results from agents with back-tracking or graph-based search. Note that the metric values shown here are averaged across three levels of underspecification.

Methods	R2R-ULN Val-Seen			R2R-ULN Val-Unseen		
	L1	L2	L3	L1	L2	L3
	SR↑	SR↑	SR↑	SR↑	SR↑	SR↑
VLN↻BERT	62.4	55.6	38.5	50.2	44.2	32.7
Ours (w/o E2E)	62.4	54.8	41.3	50.5	45.4	37.8
Ours (w/ E2E)	**62.7**	**56.6**	**43.2**	**55.7**	**47.6**	**39.5**
HAMT	**64.7**	**57.6**	38.7	**56.4**	44.6	35.1
Ours (w/o E2E)	64.4	55.9	**46.3**	56.0	44.9	38.8
Ours (w/ E2E)	62.4	56.8	43.6	55.5	**47.7**	**39.5**

Table 5: Performance breakdown by different levels of underspecification on R2R-ULN.

for efficiency. All training stages are done on a single GPU with AdamW optimizer (Loshchilov and Hutter, 2018). For E2E, we set γ as 1.2 and K as 1 for the best performance and balance between SR and SPL.

5.1 Main Results

Comparison with SOTA Agents We first compare our GSS method with the state-of-the-art (SOTA) model HAMT (Chen et al., 2021a), and a strong baseline VLN↻BERT. We mainly consider three training sets for the baselines: R2R only, R2R-last only, and R2R+R2R-last. Table 3 shows that training on a single granularity inevitably degrades the agent's performance on the other level. A mixture of these two levels shows a compromised performance across R2R and R2R-ULN validation sets. In contrast, our method can achieve 3.8% absolute SR improvement for R2R-ULN L_3 while maintaining the performance on R2R. Our GSS only requires 30% additional training iterations, 25% - 38% extra parameters, and can achieve slightly better performance.

Greedy-Decoding Agents As is shown in the top section (Row 1-5) of Table 4, greedy-decoding agents generally struggle with R2R-ULN instructions. These agents' SR and SPL values generally drop relatively by 40-50%, indicating a potential performance degradation and risk due to language variations when we deploy these models in more realistic household environments. Note that VLN↻BERT and HAMT are more robust with only a relatively 30% decrease in SR. This may attribute to better transformer architectures and initialization from large-scale pre-trained models.

Exploration-based Agents We select three exploration-based agents: FAST-Short (Ke et al., 2019), Active VLN (Wang et al., 2020), SSM (Wang et al., 2021). Table 4 shows that FAST and SSM navigate longer trajectories and achieve better SR. They are even better than VLN↻BERT and HAMT on Val-seen though they underperform these SOTA agents on R2R.

Our system outperforms the corresponding base-

Components				R2R-ULN Val-Unseen		
Classify Instr.	GSS	E2E		L1	L2	L3
		Lookahead	State Freeze	SR↑	SR↑	SR↑
				50.8	44.1	32.7
	✓			50.8	44.1	37.8
✓	✓			51.0	45.1	37.8
		✓		51.8	46.7	34.5
		✓	✓	**52.6**	46.6	35.9
✓	✓	✓		52.0	46.8	37.8
✓	✓	✓	✓	**52.7**	**47.4**	**39.5**

Table 6: Ablation study for our framework components. We use VLN↻BERT as the baseline and run experiments on full R2R-ULN Val-unseen. We use the ground truth class for GSS without a classification module.

lines by around 2% in SR and 1.6% in SPL without exploration. With the E2E module, our system gains additional improvement by sacrificing the trajectory length, resulting in lower SPL values. The improvement from GSS and E2E is consistent, as shown in Table 5. Since the target level of GSS is L_3, only L_3 SR is improved without E2E. On the other hand, E2E can improve performance across all levels by estimating grounding uncertainty and looking ahead of frontiers.

Comparison on R2R Table 3 shows the evaluation results on R2R Val-unseen. Our classification module and GSS can achieve similar or better performance than the baselines. We disable E2E since ULN does not encourage exploration for L_0, but we also show the performance with E2E in Appendix B Table 11. E2E improves the SR by 1.4% by sacrificing path length for VLN↻BERT while decreasing SR by 1.8% for HAMT. This is potentially due to inaccurate uncertainty estimation and excessive exploration for full instructions. Considering the overall performance in R2R and R2R-ULN, our method still improves the SR by an absolute 1-3 % (see Appendix B.2).

5.2 Ablation Studies

5.2.1 Component Analysis

We demonstrate the effectiveness of the GSS and E2E module in our framework in Table 6. Adding GSS brings the most significant SR gain in L_3 by 5.1% percent. This gain is intuitive as the high-level sub-network is trained on coarse-grained instructions only. Adding the classification module harms the performance for L_1 but improves it for L_2. That is because of the 85% accuracy and potentially some L_2 instructions being too short. The high-level sub-network is thus more suitable for these L_2 instructions. Secondly, adding the lookahead heuristic improves the SR by 1-2% consistently across all levels. The state freeze trick is also beneficial as it further improves the SR by 1% for L_1, L_3. Finally, we verify that gains from GSS and E2E are supplementary to each other. The last two rows show that adding lookahead on top of GSS can improve 1-2% SR on L_1, L_2 but cannot help with L_3. With state freeze, we observe an additional 2% SR gain in L_3. The potential reason is that the lookahead heuristic has overlapping benefits with GSS, but state freeze is a complementary trick for high-level sub-network.

Base	Uncertainty Threshold	R2R-ULN Val-Unseen					
		L1		L2		L3	
		SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
VLN↻BERT	0.00	53.2	13.7	46.5	12.1	38.0	9.2
	0.25	53.9	25.7	48.5	23.8	39.9	18.2
	0.50	52.6	36.9	47.4	32.5	39.5	25.6
	0.75	51.5	44.0	45.7	38.6	38.5	31.3
	1.00	51.0	45.6	45.1	40.4	37.8	33.5

Table 7: Ablation study on uncertainty threshold. We mark the value with the best SR-SPL trade-off in red.

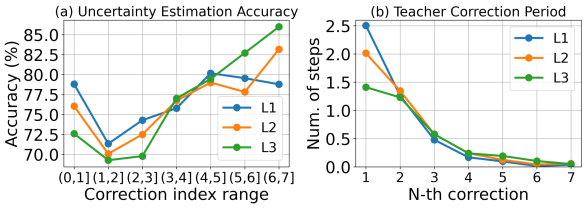

Figure 4: (a) Average estimation accuracy for steps between two consecutive corrections. (b) The average number of steps before taking N-th correction.

5.2.2 Exploration Threshold

As is mentioned in Section 4.4, the E2E module initiates an exploration when the uncertainty score s^{uncert} exceeds a threshold. We investigate the effect of this threshold value on the evaluation results in Table 7. A threshold value of 1.0 indicates no exploration at all. Decreasing the threshold imposes more explorations and improves SR by 2-3% across all levels at the expense of lower SPL. When the value goes below 0.5, the improvement in SR is marginal, while the reduction in SPL is significant. Therefore, we select 0.5 as our threshold for the best SR-SPL trade-off in practice.

5.2.3 Exploration Accuracy

Uncertainty Estimation Beyond success-based metrics, the accuracy of uncertainty estimation is an important indicator of whether the exploration steps are necessary. Given an underspecified in-

struction, we measure the accuracy with teacher corrections, i.e., correcting the agent's action when it deviates from the teacher's action. Figure 4(a) shows the accuracy for steps between two consecutive corrections. Interestingly, the accuracy drops to a minimum between the first and second correction and increases as the agent moves forward. Despite that the estimation becomes most inaccurate when the agent makes its second or third false decision, we will show that these steps are critical ones to be adjusted.

Early Exploration One interesting question that arises from the above observation is: *should we seize explorations for the first several uncertainties and encourage explorations in later steps?* The answer can be revealed by investigating the number of steps between two teacher corrections. Surprisingly, Figure 4(b) indicates that Ti deviates from T_0 at almost every step after the third correction. Assuming that our one-step lookahead is perfect as the teacher correction, the phenomenon implies that our agent has to make an exploration for every single step after the third exploration. However, such behavior is undesirable as it excessively harms navigation efficiency. Fig. 4 together implies that agents must rely on early explorations instead of later ones. Otherwise, the system error accumulates exponentially and becomes intractable eventually. Therefore, our agent relies on early explorations even though the uncertainty estimation accuracy is relatively defective.

6 Conclusion

In this work, we consider a new setting, Underspecified vision-and-Language Navigation (ULN). We collected a large-scale evaluation set, R2R-ULN, with multi-level underspecified instructions. We show that ULN is a reasonable and practical setting. ULN presents a novel direction where explorations are necessary and justifiable. As a first step towards ULN, we propose two novel components to build a VLN framework, Granularity Specific Sub-network (GSS) and Exploitation-to-Exploration (E2E). Experimental results show that GSS and E2E effectively mitigate the performance gap across all levels of instruction. Finally, we believe that ULN is a more challenging setting for future VLN development, and our framework can be further improved with more sophisticated policy design or language grounding models.

7 Limitations

In this study, we only cover Vision-Language Navigation datasets with English instructions. Instructions in other languages may characterize different types of ambiguity or underspecification. Thus, expanding the datasets to multi-lingual ULN based on datasets like RxR is essential. Secondly, we only consider indoor environments where the instructions are generally shorter than outdoor ones due to shorter path lengths. However, the phenomenon of underspecification can also be expected outdoors, accompanied by other modalities such as hand gestures or hand sketches. We simply assumed that underspecification is a more ubiquitous phenomenon in the indoor than the outdoor environment, which may be overturned from additional surveys or experiments. In the future, we hope to expand our work to multi-lingual instructions and outdoor environments and combine it with more modalities.

8 Ethical Considerations

For data collection and verification on Amazon Mechanical Turk, we select annotators from English-speaking countries, including the US, CA, UK, AU, and NZ. Each HIT for instruction simplification takes around 1.5 minutes on average to accomplish, and we pay each submitted HIT with 0.4 US dollars, resulting in an hourly payment of 16 US dollars. As for the instruction following, each HIT takes around 2 minutes to accomplish, and we pay each HIT 0.5 US dollars per HIT, resulting in an hourly payment of 15 US dollars. In addition, we award each successful navigation attempt with 0.3 US dollars for high-quality verification. As for the preference assessment, each HIT takes around 1 minute to accomplish, and we pay each HIT 0.3 US dollars, resulting in an hourly payment of 18 US dollars.

9 Acknowledgment

We would like to thank the Robert N. Noyce Trust for their generous gift to the University of California via the Noyce Initiative. The work is also partially funded by an unrestricted gift from Google. The writers' opinions and conclusions in this publication are their own and should not be construed as representing the sponsors' official policy, expressed or inferred.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4971–4980.

Arjun Akula, Spandana Gella, Yaser Al-Onaizan, Song-chun Zhu, and Siva Reddy. 2020. Words aren't enough, their order matters: On the robustness of grounding visual referring expressions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6555–6565.

Jean-Baptiste Alayrac, Adria Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. 2020. Self-supervised multimodal versatile networks. *Advances in Neural Information Processing Systems*, 33:25–37.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018a. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018b. 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.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Yonatan Bisk, Deniz Yuret, and Daniel Marcu. 2016. Natural language communication with robots. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 751–761.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. In *2017 International Conference on 3D Vision (3DV)*, pages 667–676. IEEE.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12538–12547.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021a. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. 2022. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16537–16547.

Yixin Chen, Qing Li, Deqian Kong, Yik Lun Kei, Song-Chun Zhu, Tao Gao, Yixin Zhu, and Siyuan Huang. 2021b. Yourefit: Embodied reference understanding with language and gesture. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1385–1395.

Volkan Cirik, Louis-Philippe Morency, and Taylor Berg-Kirkpatrick. 2018. Visual referring expression recognition: What do systems actually learn? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 781–787.

Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. 2020. Evolving graphical planner: Contextual global planning for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 33:20660–20672.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Qingkai Fang, Rong Ye, Lei Li, Yang Feng, and Mingxuan Wang. 2022. Stemm: Self-learning with speech-text manifold mixup for speech translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7050–7062.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3318–3329.

Tsu-Jui Fu, Xin Eric Wang, Matthew F Peterson, Scott T Grafton, Miguel P Eckstein, and William Yang Wang. 2020. Counterfactual vision-and-language navigation via adversarial path sampler. In *European Conference on Computer Vision*, pages 71–86. Springer.

Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Wang. 2022. Vision-and-language navigation:

A survey of tasks, methods, and future directions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7606–7623.

Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. 2021. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1634–1643.

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

Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. 2021. Landmark-rxr: Solving vision-and-language navigation with fine-grained alignment supervision. *Advances in Neural Information Processing Systems*, 34:652–663.

Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. 2020a. Sub-instruction aware vision-and-language navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3360–3376.

Yicong Hong, Cristian Rodriguez-Opazo, Yuankai Qi, Qi Wu, and Stephen Gould. 2020b. Language and visual entity relationship graph for agent navigation. *arXiv preprint arXiv:2010.09304*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*.

Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6741–6749.

Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. 2021. Pathdreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14738–14748.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412.

Jialu Li, Hao Tan, and Mohit Bansal. 2021. Improving cross-modal alignment in vision language navigation via syntactic information. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1041–1050.

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

Bingqian Lin, Yi Zhu, Yanxin Long, Xiaodan Liang, Qixiang Ye, and Liang Lin. 2021. Adversarial reinforced instruction attacker for robust vision-language navigation. *arXiv preprint arXiv:2107.11252*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6732–6740.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20.

So Yeon Min, Devendra Singh Chaplot, Pradeep Kumar Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. 2021. Film: Following instructions in language with modular methods. In *International Conference on Learning Representations*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR.

Abhinav Moudgil, Arjun Majumdar, Harsh Agrawal, Stefan Lee, and Dhruv Batra. 2021. Soat: A scene-and object-aware transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34:7357–7367.

Khanh Nguyen and Hal Daumé III. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 684–695.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991.

Yanyuan Qiao, Yuankai Qi, Yicong Hong, Zheng Yu, Peng Wang, and Qi Wu. 2022. Hop: History-and-order aware pre-training for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15418–15427.

Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. 2019. Cycle-consistency for robust visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6649–6658.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2610–2621.

Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. 2021. Structured scene memory for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8455–8464.

Hanqing Wang, Wenguan Wang, Tianmin Shu, Wei Liang, and Jianbing Shen. 2020. Active visual information gathering for vision-language navigation. In *European Conference on Computer Vision*, pages 307–322. Springer.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6629–6638.

Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. 2016. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85. Springer.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. 2021a. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699.

Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. 2020a. Babywalk: Going farther in vision-and-language navigation by taking baby steps. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2539–2556.

Wanrong Zhu, Yuankai Qi, Pradyumna Narayana, Kazoo Sone, Sugato Basu, Xin Eric Wang, Qi Wu, Miguel Eckstein, and William Yang Wang. 2022. Diagnosing vision-and-language navigation: What really matters. *NAACL 2022*.

Wanrong Zhu, Xin Wang, Tsu-Jui Fu, An Yan, Pradyumna Narayana, Kazoo Sone, Sugato Basu, and William Yang Wang. 2021b. Multimodal text style transfer for outdoor vision-and-language navigation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1207–1221.

Wanrong Zhu, Xin Wang, Pradyumna Narayana, Kazoo Sone, Sugato Basu, and William Yang Wang. 2020b. Towards understanding sample variance in visually grounded language generation: Evaluations and observations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8806–8811.

Number of:	R2R	R2R-ULN		
	L0	L1	L2	L3
Instructions	1622	3282	3282	3282
Paths	917	917	917	917
Tokens	38.6	27.3	18.9	8.7
Direction Tokens	2.5	1.1	0.7	0.2
Object Tokens	8.5	6.6	4.6	2.6

Table 8: R2R validation v.s. our R2R-ULN validation. R2R-ULN preserves most of the paths from R2R. It is large-scale, human-centric, and strictly aligns all levels of underspecification.

Level	Instructions
L_0	Go straight then slightly right to continue going straight. Exit the room then turn left and go into the room next door. Turn right and go past the wall with the holes in it. Wait near the lockers.
L_1	Exit the room and go into the room next door. Go past the wall with holes in it. Wait near the lockers.
L_2	Exit the room and go into the room next door. Wait near the lockers.
L_3	Go into the room next door and wait near the lockers.
L_0	Turn around, go through the kitchen and up the stairs to the right. When at the top of the steps, turn to the left and go down the hallway. Stop in front of the painting on the right wall.
L_1	Go through the kitchen and go up the stairs. Go down the hallway. Stop in front of the painting on the wall.
L_2	Go up the stairs and go down the hallway. Stop in front of the painting on the wall.
L_3	Stop in front of the painting on the wall down the hallway.
L_0	Turn and walk towards the open brown wooden door that leads to an office with a large desk. Exit the room through the door. Walk around the left side of the table and go through the double open doors that lead to a hallway. Walk out into the hallway until you reach the first door on the right. Turn tight and take two steps into the room, stopping in the doorway to the room next to the carpet.
L_1	Walk towards the open brown wooden door that leads to an office with a large desk. Exit the room. Walk around the table and go through the double open doors that leads to a hallway. Walk out into the hallway until you reach the first door. Take two steps into the room, stopping in the doorway to the room next to the carpet.
L_2	Take the open brown wooden door that leads into the office. Walk around the table and go into the hallway. Stop in the doorway of the room next to the carpet.
L_3	Go to the doorway of the room next to the carpet in the hallway.

Table 9: More instruction examples from the R2R-ULN validation set. Words marked in red are removed, and words marked in blue are rephrased in the next level.

A ULN Datasets

A.1 Data Collection

Denote the original R2R instructions as Level 0 (L_0). We define Level 1 (L_1) instructions as ones where the redundant part is removed from L_1. Level 2 (L_2) instructions are partial route description with some sub-instructions removed from L_1. Level 3 (L_3) instructions directly refer to the goal destination without any intermediate route information.

Level 1: We form the data collection stage as a sentence simplification task and ask human workers to progressively omit details from the instruction. For the first step, workers remove redundant parts from L_0 to obtain L_1. Redundancy includes but is not limited to repetition, excessive details, and directional phrases. To make it more human-centric, we allow the workers to determine the degree of redundancy. For example, some may rewrite "turn 180 degrees" as "turn around", while others may delete the whole phrase assuming that "turn around" is still redundant.

Level 2: Then, one or two phrases containing objects are removed from L_1, resulting in partial route descriptions (L_2). L_2 assumes that humans tend to ignore intermediate references when the route is partially visible, or they are familiar with the environment.

Level 3: Finally, the third step requires workers to write one sentence directly referring to the goals by combining L_0 and providing information like region label and floor level (L_3). L_3 resembles instructions in REVERIE (Qi et al., 2020) but is restricted to pure navigational instructions while REVERIE commands the agent to interact with target objects. As a result, we obtain one triplet (L_1, L_2, L_3) per L_0 instruction per annotator.

Such progressive simplification design enables us to control the **inter-level alignment** as workers inject no external objects/directions or substitute existing ones with external ones for each L_0. It also preserves **intra-level sample variance** since workers simplify the sentences based on subjective judgments on the degree of redundancy, yet circumscribed by the definition of levels.

Instruction Following We ask workers to reach the goal by following instructions and operating in an interactive WebGL environment. We randomly sample 250 triplets plus L_0 instructions from the validation set. We follow a similar setup as in (Anderson et al., 2018b). As is shown in Table 2, workers can still achieve over 80% success rate (SR), the most suitable metric for ULN, on L_1 and L_2 and maintain a high-quality performance on L_3. Hence, ULN is a feasible setting where agents should actively make more explorations as the instructions become less specific.

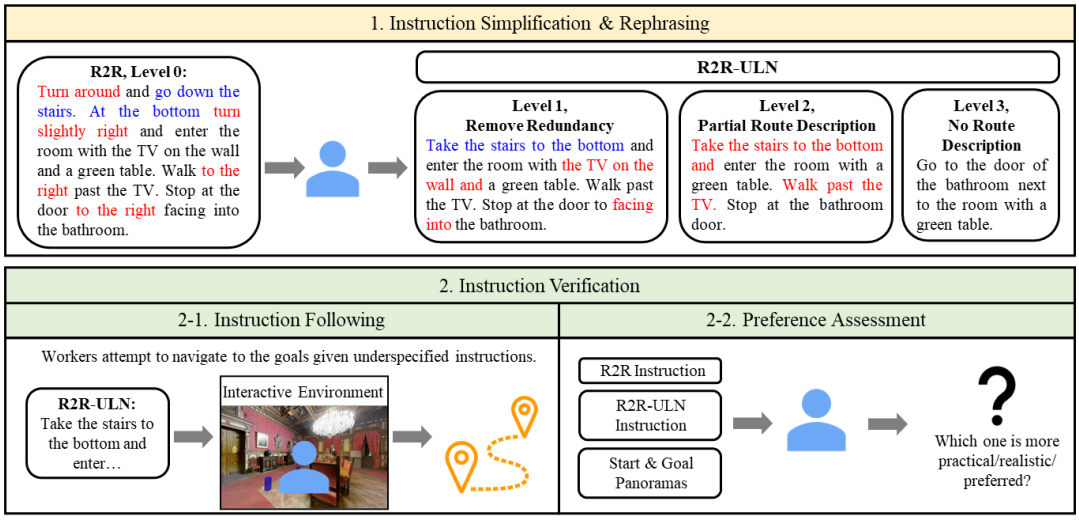

Figure 5: Dataset collection and verification. Text in red is removed in the next level and text in blue is paraphrased in the next level.

Preference Assessment As is shown in the bottom right of Fig. 5, we investigate human preference on whether the full instructions or our collected ones are more practical and realistic. We sample 600 L_1-L_3 instructions and form each one with its corresponding L_0 instruction as a pair (Lx, L_0), x $\in [1, 3]$. Given (Lx, L_0) and starting and goal viewpoint panoramas, workers are asked "Q1. Which one is a more practical expression in daily life?" and "Q2. Which one do you prefer to speak to command the robot, considering applicability and efficiency?". As shown in Table 2, the results demonstrate an increasing trend in choosing shorter and less specific instructions to reflect more practical expressions and benefit applicability and efficiency.

A.2 Dataset Statistics

As illustrated in Table 8, R2R-ULN preserves most of the trajectories from the R2R validation set. It addresses the shortage of underspecified instructions, which is essential for evaluating the generalization of embodied agents to various language expressions. Instructions of different lengths are better aligned, making it a better reference to investigate the correlation between instruction length and agent performance.

B Supplementary Experiments

B.1 Identifying GSS

Intuitively, this sub-network incorporates into the cross-modal transformer. The conjecture is that the cross-attention layers shift the vision-to-text attention from the first token to the end for low-level instructions. On the other hand, for high-level instructions, the visual features may persistently attend to the goal object tokens. To verify this intuition,

Replaced part	R2R-ULN Val-Unseen Level 3 (L_3)			
	TL	NE↓	SR↑	SPL↑
VLN↻BERT				
low-level	13.20	7.41	32.71	29.07
f_{text}	14.23	7.18	36.64	31.25
f_{text}, f_{emb}	14.26	7.26	36.37	31.18
f_{img}	12.31	7.38	33.07	29.27
f_{cm}	13.50	7.15	34.94	31.22
high-level	13.31	6.96	36.45	31.99
HAMT				
low-level	13.26	7.18	35.12	31.13
f_{text}	13.26	7.18	35.12	31.13
f_{hist}	13.26	7.18	35.12	31.13
f_{img}	13.22	7.28	34.23	30.38
f_{cm}	13.26	6.75	38.74	34.86
high-level	13.34	6.95	37.18	33.40

Table 10: Low-level baseline performance after replacing sub-network from another high-level agent with the same architecture. The low-level agent is one trained with full R2R instructions. The high-level agent is one trained with last sentence of R2R instructions.

We show three additional experimental results here for supplementary. First, as is shown in Table 10, we identify the critical sub-network by replacing part of a low-level agent with that from a high-level agent and observe the performance change. For VLN↻BERT, the most crucial sub-

Methods	R2R Val-Seen		R2R Val-Unseen	
	SR↑	SPL↑	SR↑	SPL↑
EnvDrop	62.3	59.4	50.0	46.8
PREVALENT	69.5	65.4	58.3	53.6
FAST-Short	-	-	55.8	43.3
SSM	71.1	61.8	62.4	45.3
VLN↻BERT	**74.2**	**69.8**	61.4	55.6
Ours (w/o E2E)	73.9	69.3	61.6	**55.8**
Ours (w/ E2E)	73.0	57.3	**62.8**	45.0
HAMT	75.6	72.2	66.2	61.5
Ours (w/o E2E)	73.6	70.2	65.6	60.7
Ours (w/ E2E)	73.3	51.9	64.4	41.3

Table 11: Performance on R2R validation set in the single-run setting. We disable E2E since ULN does not encourage exploration with L_0 instructions.

Training Set	# of training instr.	R2R-ULN Val-Unseen L3	
		SR↑	SPL↑
R2R-Last+Speaker	24505	**34.8**	**31.5**
R2R-Last+REVERIE	24505	32.3	28.3
R2R-Last+SOON	16718	34.0	29.3

Table 12: Agents trained with a combination of R2R-Last and one of the existing high-level datasets, REVERIE or SOON.

network that impacts its navigation mode is the text encoder, as replacing it achieves the highest performance. Note that replacing the embedding layer downgrades the performance. We hypothesize that the high-level agent is trained with a smaller vocabulary due to shorter instructions. As for HAMT, the critical sub-network is the cross-modal encoder, which aligns with our hypothesis.

B.2 Ablation Studies

R2R Evaluation We also show full performance results on R2R in Table 11. Our framework maintains a comparable performance on R2R for VLN↻BERT and even slightly improves the SR with E2E. However, as for HAMT, our framework downgrades the SR by 0.6%. This is due to the imperfect classifier that misclassifies some short R2R instructions as high-level instructions. We empirically find that the HAMT is more sensitive to classifier error, indicating better robustness of HAMT in handling short R2R instructions. It spares the need to run the high-level sub-network to deal with those exceptions in R2R.

Components				R2R-ULN Val-Unseen					
Classify Instr.	GSS	E2E		L1		L2		L3	
		Lookahead	State Freeze	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
				50.8	**45.4**	44.1	39.5	32.7	29.1
	✓			50.8	**45.4**	44.1	39.5	37.8	**33.5**
✓	✓			51.0	45.6	45.1	**40.4**	37.8	**33.5**
		✓		51.8	37.1	46.7	31.9	34.5	21.3
		✓	✓	**52.6**	36.7	46.6	31.3	35.9	21.5
✓	✓	✓		52.0	37.3	46.8	32.4	37.8	24.6
✓	✓	✓	✓	**52.7**	36.9	**47.4**	32.5	**39.5**	25.6

Table 13: Full ablation study with SPL reported.

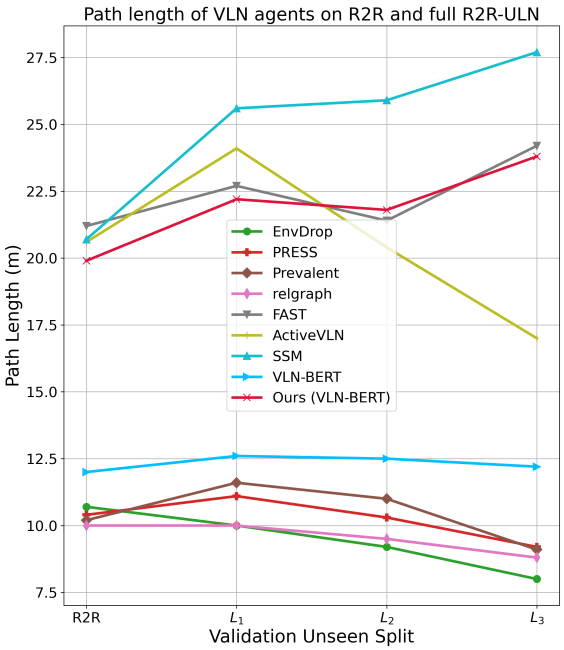

Figure 6: Path length of various agents on R2R and R2R-ULN.

Combining Multiple Datasets Since there are existing high-level datasets like REVERIE (Qi et al., 2020) and SOON (Zhu et al., 2021a), there is also a natural motivation to combine these datasets during training since these instructions can be seen as goal instructions (similar to L_3). We trained three agents with a combination of R2R-Last and one of the high-level datasets. For comparison, we also train an agent with R2R-Last, and the same amount of speaker-augmented instructions. Note that the speaker is trained with last sentences as well, so it is a goal instruction speaker. As is shown in Table 12, existing datasets have limited benefit on L_3 evaluation. REVERIE and SOON instructions contain both navigation and localization orders, making these datasets noisier for pure navigation training and evaluation. We leave it as future work to consider combining VLN datasets of different settings for one unified, general navigational agent.

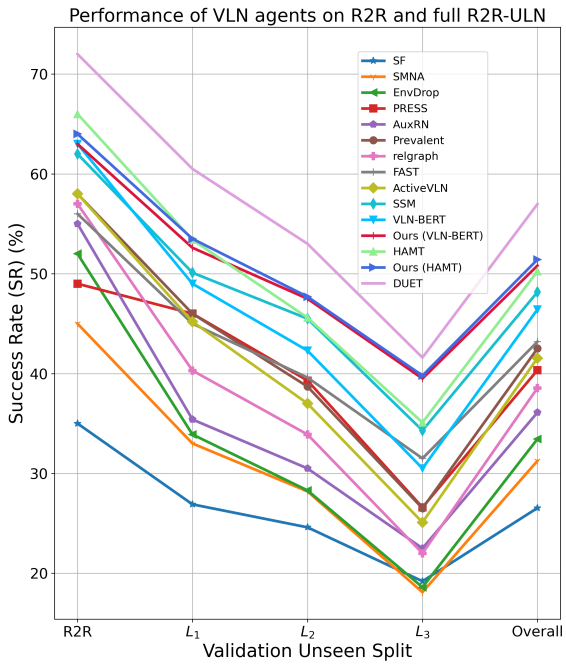

Figure 7: Evaluation results of many base agents on R2R and full R2R-ULN.

Path Length As is shown in Fig. 6, most greedy-decoding agents learn a spurious correlation between instruction length and trajectory length (TL), reflected as a decreasing trend in TL from L_1 to L_3. However, SOTA agents are more robust as they navigate slightly longer for L3. Exploration-based agents showed significantly longer TL and increasing trends, which is more robust (in SR percentage drop) and interpretable.

Component Analysis Last but not least, we show the full ablation table with SPL compared to Table 6. Our GSS-only framework achieves the highest SPL while adding the E2E module decreases the SPL value. This is intuitive as the purpose of E2E is to improve the success rate by making more exploration steps, and will result in a smaller SPL value.

B.3 Trajectory Visualization

We provide visualizations of a set of underspecified instruction (L_1, L_2, L_3) in add to Fig. 1. As is shown in Fig. 8, the base agent fails to go downstairs from the beginning and keeps moving around on the initial floor. Our agent also navigates on the initial floor for a long time. However, it starts moving downstairs from step 14 thanks to uncertainty estimation and active exploration at the previous step. Finally, it reaches the lounge downstairs. Fig. 9 shows the trajectories of an L_2 instruction. Similarly, the base agent keeps moving around on the initial floor and fails to go downstairs. However, our agent classifies the instruction as high-level and adopts the high-level GSS to guide the navigation decisions. Therefore, it reaches the goal with only seven steps. As for L_3, the base agent manages to go downstairs at step 7 but keeps moving downstairs and completely misses out on the lounge, resulting in a long path. In contrast, our agent with the high-level GSS reaches the target location with only seven steps. There is no exploration because no uncertain steps are identified.

Original Instruction: Exit the room and turn right to go down the stairs. After descending two flights of stairs, continue forward to enter the room ahead of you and slightly to the left.

Instruction (L_3): Exit the room and go down the stairs. Continue forward to enter the room ahead of you.

Figure 8: Visualization of trajectories following an L_1 instruction.

Original Instruction: Exit the room and turn right to go down the stairs. After descending two flights of stairs, continue forward to enter the room ahead of you and slightly to the left.

Instruction (L_2): Go downstairs, and enter the room ahead on the left.

Figure 9: Visualization of trajectories following an L_2 instruction.

Original Instruction: Exit the room and turn right to go down the stairs. After descending two flights of stairs, continue forward to enter the room ahead of you and slightly to the left.

Instruction (L_3): Go downstairs to the lounge.

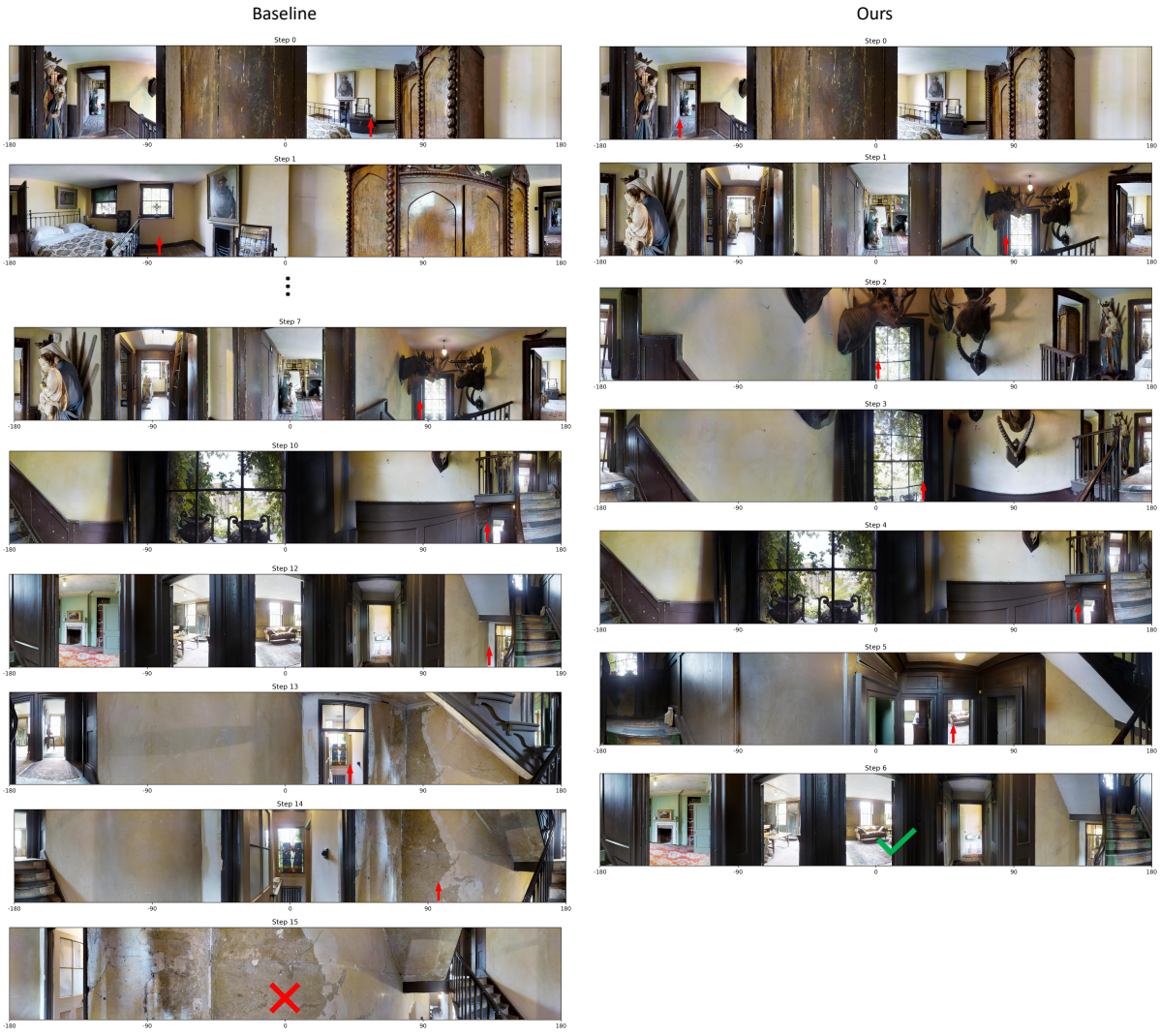

Figure 10: Visualization of trajectories following an L_3 instruction.