CaneSpeaker: An LLM-Assisted Speaker for Generating Human-Like Navigation Instructions

YUANYU ZHENG, LIN ZHANG*, YUNDA SUN, YING SHEN, and SHENGJIE ZHAO, School of Computer Science and Technology, Tongji University, China

Navigation instruction generation aims to address data scarcity in Vision-and-Language Navigation (VLN) by generating navigation instructions for unannotated routes from data sources like simulators or online data. However, existing methods usually suffer from high reliance on panoramic views, poor cross-task generalization ability, and limited availability of training data. To address these challenges, we propose a novel speaker, CaneSpeaker, to generate human-like instructions from front-facing images for a variety of VLN tasks. Firstly, to mitigate the limited amount of speaker training data, we propose an LLM-based instruction augmentation method, LLM-IA, that utilizes an off-the-shelf LLM to create augmented instructions for training by distilling and reformulating existing instructions. This method allows us to collect an instruction-augmented dataset with human-level accuracy for speaker training, namely Rx2R. Secondly, to eliminate the dependency on panoramic views, we propose a novel Vision-Language Model (VLM)-based speaker architecture, VL-Sp. By leveraging the advanced reasoning capabilities of a pre-trained VLM, CaneSpeaker can effectively generate high-quality instructions directly from front-facing images without relying on panoramic views. Also, the prompt-based characteristic of the VLM allows us to devise a unified input representation to enable the processing of multiple VLN tasks, thus further addressing the problem of data scarcity by combining multiple datasets from different VLN tasks. Finally, we utilize CaneSpeaker to synthesize a large-scale augmented dataset, CANE, from unannotated routes in the Matterport3D Simulator. Comprehensive experiments demonstrate that CaneSpeaker generates precise instructions with diverse expressions across various VLN tasks, and the VLN agent trained on our datasets obviously outperforms its counterparts. The source codes and datasets are available at https://github.com/zheng19845/CaneSpeaker.

CCS Concepts: • **Computing methodologies** → **Natural language generation**.

Additional Key Words and Phrases: Vision-and-Language Navigation, Navigation Instruction Generation, Vision Language Model, Large Language Model.

ACM Reference Format:
Yuanyu Zheng, Lin Zhang, Yunda Sun, Ying Shen, and Shengjie Zhao. 2025. CaneSpeaker: An LLM-Assisted Speaker for Generating Human-Like Navigation Instructions. *ACM Trans. Multimedia Comput. Commun. Appl.* 1, 1, Article 1 (January 2025), 26 pages. https://doi.org/XXXXXXX.XXXXXXX

*Corresponding author: Lin Zhang.

This work was supported in part by the National Natural Science Foundation of China under Grant 62272343 and in part by the Fundamental Research Funds for the Central Universities.
Authors' Contact Information: Yuanyu Zheng, 2433271@tongji.edu.cn; Lin Zhang, cslinzhang@tongji.edu.cn; Yunda Sun, 2110850@tongji.edu.cn; Ying Shen, yingshen@tongji.edu.cn; Shengjie Zhao, shengjiezhao@tongji.edu.cn, School of Computer Science and Technology, Tongji University, Shanghai, China.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM 1551-6865/2025/1-ART1
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

To develop real-world robots with the ability to understand natural language commands and carry out navigation tasks, extensive research [8, 9, 11, 13, 14, 17, 18, 31, 32, 38, 40, 43] has been conducted in the field of Vision-and-Language Navigation (VLN) [6]. In VLN, navigation instruction generation is a widely-studied problem that focuses on generating language-based commands for unannotated routes, usually from simulators or other online data sources. Research in navigation instruction generation aims to develop generative models (referred to as speakers) to synthesize augmented datasets to perform large-scale training for VLN agents [11, 14, 18, 32, 38] (also referred to as followers) to address the difficulties in generalizing to unfamiliar environments and commands, caused by the diversity of visual observations and the complexity of natural language instructions.

Generating high-quality navigation instructions for training VLN agents is a complicated problem that involves visual comprehension, landmark selection, and spatial reasoning. Although recent studies [3, 13, 14, 30, 32, 35, 37] have explored leveraging multi-modal generative models to synthesize instructions for VLN tasks, they are still limited by three main difficulties.

First, the limited amount of annotated training data in existing datasets restricts the speakers' ability to ground languages to landmarks. As aforementioned, most current speakers [3, 13, 14, 30, 32, 35, 37] are specialized models trained on a single VLN dataset. Similar to the training of VLN agents, the training of these generative speakers also suffers from a shortage of training data. Consequently, the instructions generated by these speakers often lack the human-like attributes expected for navigation guidance, including precise landmark selection and diverse linguistic expressions.

Second, current VLN speakers heavily rely on panoramic observations to enhance spatial reasoning. Most existing models require a panoramic view composed of 36 RGB images (12 headings $\times$ 3 elevations) to perform spatial reasoning for navigation instruction generation. While effective in high-resource settings, this dependency limits their applicability in scenarios where panoramas are unavailable, such as large-scale data sources from online platforms (for example, house markets [17] and YouTube videos [23]). Additionally, the large number of images in the panoramas leads to extended input sequences. To address this, many speakers [3, 14, 32, 35, 37] extract fewer features from each image, thereby reducing the amount of detailed information that can be obtained from individual images.

Third, most current methods exhibit weak cross-task generalization capabilities. Existing research [3, 13, 14, 30, 32, 35, 37] primarily focuses on developing task-specific generative speakers for navigation instruction generation. These models are typically designed and trained for specific VLN tasks, making it challenging for them to accommodate to the varying granularity and diverse input modalities across different VLN tasks. For instance, BT-speaker [14] and EDrop-speaker [32], two most widely used speakers, are designed to generate fine-grained, step-by-step instructions leading to a destination. However, when applied to tasks like REVERIE [27], where instructions not only provide guidance to a destination but also specify a target object, these models lack a mechanism to designate the target that should be mentioned in the instruction in a manner akin to human annotation.

In this work, to address these challenges, we devise CaneSpeaker, a novel cross-task speaker that can generate human-like navigation instructions only from front-facing views (as shown in Fig. 1).

Firstly, to mitigate the limitations posed by the scarcity of speaker-training data, we propose LLM-IA, a novel instruction augmentation approach that leverages an off-the-shelf Large Language Model (LLM) to generate high-quality synthetic navigation instructions. This augmentation process enables the creation of a large-scale, high-accuracy dataset, which we refer to as Rx2R. The collected Rx2R dataset not only contains rich linguistic variety and precise grounding of language to visual

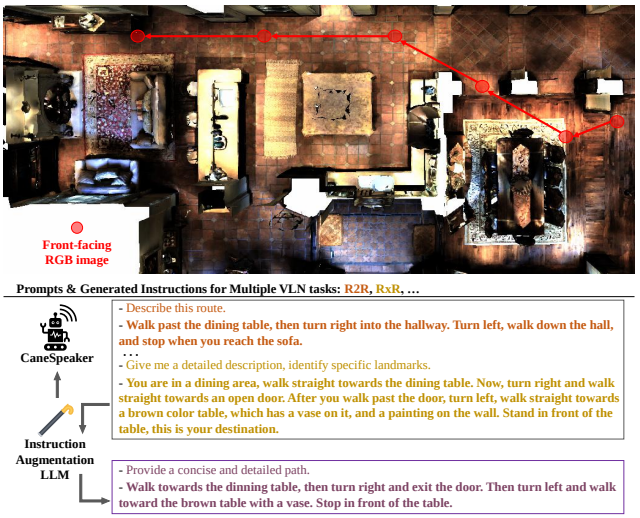


Fig. 1. Our proposed CaneSpeaker is supported by an LLM for instruction augmentation. CaneSpeaker is capable of generating VLN instructions from only front-facing RGB images along a specified path and producing task-specific instructions in response to different prompts.

landmarks that can aid speaker training, it also features distinct characteristics from existing datasets, which make it possible for it to perform as a standalone VLN evaluation dataset.

Secondly, to eliminate the dependency on panoramic observations, we introduce a Vision-Language Model (VLM)-based [41] speaker architecture, VL-Sp, that effectively leverages the comprehension, reasoning, and generation capabilities of the pre-trained VLM. Making full use of the powerful capabilities of the VLM, our speaker can generate high-quality navigation instructions solely from front-facing images, unlike conventional methods that require full panoramic views. This enables broader applicability in real-world scenarios where panoramic data may be unavailable.

Additionally, with the help of the VLM's cross-modal understanding ability, we are able to design a unified input representation across multiple tasks to facilitate the cross-task processing ability of CaneSpeaker. Specifically, we devise an input formulation consisting of task-specific prompts, along with front-facing images and simple actions to generate instructions in a human-like fashion for a variety of VLN tasks. By incorporating the unified representation, our speaker can utilize the visual comprehension and contextual understanding capabilities of VLM to seamlessly adapt to different VLN tasks without the need for extensive task-specific modifications. Such adaptability allows us to further address data scarcity by combining multiple datasets from different VLN tasks, enhancing the diversity and quality of training data.

To evaluate the effectiveness of our approach, we apply CaneSpeaker to establish a multi-task augmented dataset, CANE, for training VLN agents, and then benchmark it against existing augmented datasets. Experimental results demonstrate that not only does the generated CANE dataset enhance agent performance, but the instruction augmentation dataset, Rx2R, synthesized for speaker training, also significantly improves agent navigation performance.

Our contributions can be summarized as:

- A meticulous LLM-assisted instruction augmentation method, LLM-IA, is designed for generating high-quality instructions that meet the high demands of speaker training. Benefiting from this method, Rx2R, an augmented dataset with human-level accuracy and distinct

characteristics from existing datasets, is proposed as not only augmented speaker-training data but also a standalone VLN evaluation dataset.
- A VLM-based speaker architecture, VL-Sp, for instruction generation in VLN is proposed. By leveraging the powerful comprehension, reasoning, and generation ability of a pre-trained VLM, CaneSpeaker can generate human-like instructions directly from front-facing images, and generalize effectively across various VLN tasks.
- CANE, a large-scale cross-task augmented VLN datasets is established using our proposed speaker, CaneSpeaker. Extensive experiments on instruction generation and agent navigation validation show that CaneSpeaker can generate accurate human-like instructions and the VLN agent trained on Rx2R and CANE performs best among its counterparts.

2 Related Work

2.1 Vision-and-Language Navigation

Vision-and-Language Navigation (VLN) focuses on enabling agents to follow language-based directions and navigate through unfamiliar environments, a skill vital for developing intelligent robots that can aid with everyday tasks. Researches in VLN usually leverage realistic 3D virtual environments, such as the Matterport3D Simulator [7] , to develop agents capable of following language-based instructions [6], retrieving objects [27, 44] and multilingual understanding [21].

Early VLN research [6] primarily focused on developing end-to-end trainable sequence-to-sequence agent that processes the natural language instruction and generates fine-grained movement commands. Later, to enhance the navigation performance, BT-speaker [14] adopted a panoramic action space that consists of high-level navigational decisions within panoramic observations. This design enables the agent to focus on high-level decision-making without the intricacies of low-level motor control. As a result, many subsequent works built upon this setting, and utilized other techniques such as mixing imitation learning and reinforcement learning [32], incorporating navigation history [10, 29], or integrating various mapping strategies, such as topological maps, [5, 11], bird-eye-view maps [4], and volumetric representation of the environment [24] to enhance navigation capabilities.

More recently, to improve the way-finding performance and generalization ability across diverse environments, more and more studies that focus on augmenting training data for VLN agents have been proposed in the field of VLN. Among them, early approaches usually collected unannotated routes within the Matterport3D environments [14, 18] and utilized the environmental dropout technique [32] to generate varied environments. More recently, additional data sources have been explored, such as online marketplaces [17], YouTube videos [23], or HM3D [38]. Upon collecting new routes from these environments, navigation instructions are automatically generated and then used for training.

2.2 Instruction Generation for VLN

Traditional instruction generation methods primarily focus on developing generative speaker models trained on human-annotated data [3, 13, 14, 32]. Subsequent research has sought to enhance these models through various strategies, such as jointly training the speaker alongside the navigation agent [35], sharing parts of the speaker's structure with the VLN agent [37], or adopting a GAN-like framework to generate refined instruction generation [30]. While they perform reasonably well in tasks such as R2R [6], where instructions are largely based on regional descriptions, most above methods exhibit significant limitations in more challenging tasks like RxR [21], where linguistic diversity is greater and landmark selection is more flexible. To address these shortcomings, Marky [36] introduced a two-stage framework that first identifies landmarks, and then generates

instructions using the selected landmarks. Although this approach enables the accurate selection of landmarks, such a two-stage design makes Marky blind to environment context, as it prevents the model from fully integrating contextual scene information during instruction generation.

Additionally, these aforementioned instruction generation methods face several fundamental challenges. First, they rely heavily on panoramic observations, either by directly utilizing panoramas or by leveraging them to construct bird's-eye-view (BEV) representations, limiting their applicability in real-world data sources where panoramas are not available. While some studies [17, 23] attempt to circumvent this dependency through template-based methods, the resulting instructions are often of low quality and lack the richness of natural human language. Second, training a generative model requires large amounts of data, yet most VLN datasets are relatively small, limiting the diversity and comprehensiveness of the training data. As a result, the generated instructions frequently lack coherence and adaptability. Third, most existing models are tailored for specific VLN tasks and struggle to generalize across different navigation instruction generation settings. In particular, they are challenged by the need to accommodate both fine-grained instructions, which provide detailed step-by-step guidance, and coarse-grained instructions, which specify the destination and a target.

In contrast to previous methods, we propose a novel speaker that generates accurate human-like instructions across various VLN tasks using a series of front-facing images. By eliminating the reliance on panoramic observations and leveraging a unified input representation across tasks, our approach enhances both the effectiveness and generalizability of instruction generation.

2.3 Foundation Models in VLN

Since the emergence of foundation models, such as LLMs and VLMs, numerous studies have explored the use of these models in the field of VLN.

Among these studies, most existing research primarily focuses on leveraging these large pre-trained foundation models to develop navigation agents. Early approaches investigate the direct application of these models without additional training, employing a zero-shot paradigm. For instance, NavGPT [43] leverages the reasoning capabilities of GPT-4 [2] to perform zero-shot sequential action predictions in complex embodied environments; LangNav [26] explores the concept of using language as a perceptual representation by converting visual observations into linguistic descriptions; MapGPT [8] introduces a map-guided prompting technique that utilizes a language-formed map to encourage global exploration; DiscussNav [25] builds an LLM-based navigation agent that discusses with several domain experts before making navigation decisions. More recently, an increasing number of studies focus on fine-tuning these pre-trained models to build agents with enhanced navigation performance. NaviLLM [40] fine-tunes a 7B LLaMA [34] model to develop a general navigation system capable of handling diverse VLN tasks; NavCoT [22] introduces a parameter-efficient fine-tuning approach using the Chain-of-Thought strategy to improve decision making; NavGPT-2 [42] designs an LLM-based navigation agent that bridges the gap between specialist models and LLM-based approaches by incorporating a hierarchical architecture consisting of a VLM and a navigation policy network. Additionally, [28] investigates the use of LLMs as copilot assistants in coarse-grained VLN tasks, leveraging their reasoning and comprehension capabilities to improve specialist performance.

However, due to the scarcity of high-quality data for fine-tuning large generative models, the use of foundation models for navigation instruction generation has received much less attention. BEVInstructor [13] constructs BEV features from panoramas, integrates them into a LLaMA [34] model, and then fine-tunes the LLM for instruction generation. It is worth noting that BEVInstructor only exploits the generative capabilities of the LLM to produce coherent instructions. It does not fully leverage the reasoning or generalization ability of the LLM to handle multiple VLN tasks. Instead, it fine-tunes the model on separate VLN tasks, similar to conventional methods.

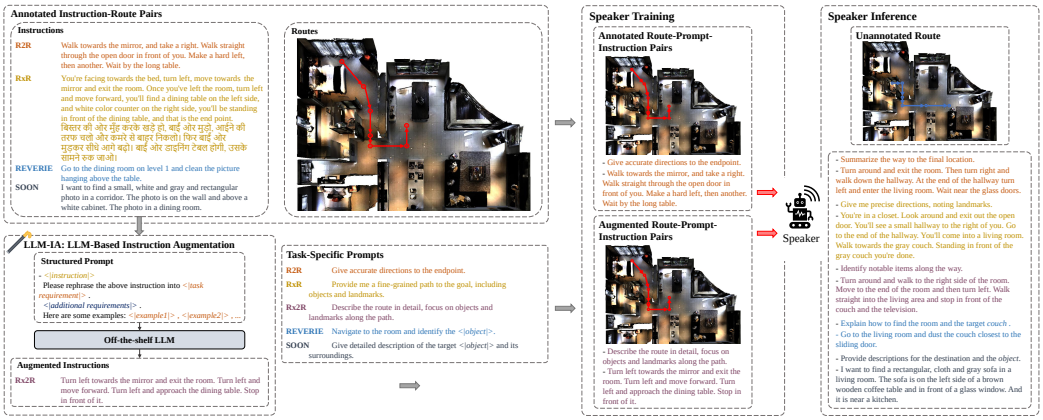


Fig. 2. Overview of CaneSpeaker. We first perform data augmentation through LLM-IA, and collect a large-scale augmented dataset Rx2R. Then we devise a VLM-based architecture, VL-Sp, and train CaneSpeaker on a variety of VLN tasks, along with the augmented data. The speaker is capable of generating human-like navigation instructions for multiple VLN tasks at inference.

3 Methodology

To address the aforementioned challenges in Sec. 2, CaneSpeaker first performs data augmentation based on an LLM and then devises a cross-task speaker architecture for various VLN tasks. The overview of our method is shown in Fig. 2.

3.1 Problem Formulation

The task of navigation instruction generation aims to produce a natural language command $\mathbf{Y}$ that describes and guides a navigation route $\mathbf{R}$.

To achieve this, the speaker model collects the visual observation Obs_t at each discrete step t along the route, as well as the corresponding action Act_t taken between successive steps. The specific content of the observations and actions differs depending on the setting. For instance, in the traditional panoramic setting, the observations consist of panoramic views, while the actions correspond to the selected viewing angles. The entire route can be represented as,

$$\mathbf{R} = (Obs_1, Act_1, \ldots, Obs_T, Act_T). \tag{1}$$

The model then processes this sequence of observations and actions to generate the instruction, as follows,

$$\mathbf{Y} \sim P(\mathbf{Y} \mid \mathbf{R}; \mathbf{S}), \tag{2}$$

where $\mathbf{S}$ denotes the speaker model utilized to generate the navigation instruction.

3.2 LLM-Assisted Data Augmentation

LLM-IA: LLM-Based Instruction Augmentation. The lack of high-quality training data has long posed a significant challenge for instruction generation, where training a speaker requires large volumes of well-annotated data. Although current studies have explored training generative speakers on VLN datasets, the experimental results demonstrate that existing methods cannot generate highly-accurate instructions that resemble human-annotated data [11, 14, 38].

With this in mind, an intuitive idea to tackle this issue is to perform data augmentation on existing annotated instructions through a non-training based approach. With the appearance of LLMs and

their demonstrated effectiveness in tasks that require nuanced reasoning and text generation, applying an LLM for augmenting navigation instructions via a fully text-based prompting approach appears promising. Driven by this idea, LLM-IA, a method that leverages an off-the-shelf LLM to generate new instructions is proposed.

LLM-IA aims to generate high-accuracy augmented instructions from existing human-annotated instructions by a prompting-based approach. Empirically, we observe that instruction augmentation is most effective when transforming a detailed source instruction into a concise directive across different tasks. This can be attributed to several factors: (a) when augmenting instructions within the same task domain, LLMs struggle to generate truly novel instructions, resulting in reduced diversity and increased semantic overlap, and (b) LLMs excel at reasoning and extracting key information from detailed, free-form instructions. However, when tasked with generating detailed instructions, they are more prone to hallucination. As a result, we design the following prompt structure to systematically guide the LLM in augmenting instructions across tasks.

-<|instruction|>
Please rephrase the above instruction into <|Task Reformulation|>.
<|Semantic and Structural Guidelines|>.
Here are some examples: <|Reference|>, ...
-[augmented instruction],

where *<|instruction|>* represents the original human-annotated directive, serving as the source text to be reformulated; *<|Task Reformulation|>* specifies the intended transformation, defining how the instruction should be restructured, for example, converting a detailed route description into a concise navigational command; *<|Semantic and Structural Guidelines|>* establish the desired linguistic and formatting constraints, such as selecting important landmarks or ensuring clarity, natural phrasing, and coherence; and *<|Reference|>* provides representative examples demonstrating the expected output style, offering concrete illustrations of properly reformulated instructions that adhere to the specified guidelines. LLM-IA represents the first attempt to augment instructions for speaker training, and provides an effective approach to mitigate the issue of data scarcity.

Rx2R. Using LLM-IA, an augmented dataset for speaker-training, Rx2R, with human-level accuracy is created with the RxR [21] dataset as the source dataset and R2R [6] as the target task.

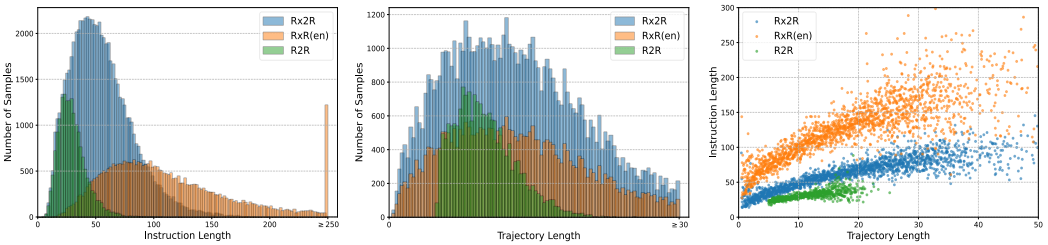
For better assessment of the Rx2R dataset, we collected 400 human wayfinding trajectories on the val_unseen split, involving 10 human annotators. In addition to performing the navigation task, participants were asked to rate the following aspects on a 1–5 scale: (a) the overall quality of the instructions, (b) whether key information was missing compared to the original RxR instruction, and (c) whether the instruction contained errors or misinterpreted information. The results are summarized in Table 1. Compared to the original RxR instructions, the success rate is nearly identical, with only a marginal difference of 0.5%. Furthermore, the ratings for missing information and error content indicate that Rx2R instructions rarely omit essential information and exhibit minimal error. These results confirm that the generated instructions maintain an accuracy level comparable to the original human-annotated RxR instructions.

In addition to its high accuracy characteristics, the generated instructions also incorporate structured language (similar to R2R), alongside the varied landmarks and route lengths (typical of RxR). While initially developed to serve as a high-quality augmented dataset for speaker training, such high quality and distinct characteristics from existing dataset suggest that it can also be used as a standalone evaluation dataset. Therefore we assign these instructions to a new task, Rx2R, instead of directly including them to the R2R task.

Fig. 3 presents a visualization comparing the distribution of Rx2R instructions to those in R2R and RxR. The generated Rx2R dataset comprises 67,977 instructions corresponding to 12,591 paths.

Table 1. Human evaluation of the Rx2R instructions. "Avg. Len." stands for the average length of the instructions, "Miss. Sc." denotes the score for whether key information is missing compared to the original RxR instruction, and "Err. Sc." denotes the score for whether the instruction contains errors or misinterpreted information. SR serves as the primary metric.

Datasets	Avg. Len.	val_seen				Quality	Miss. Sc.	Err. Sc.
		TL	NE ↓	**SR** ↑	SPL ↑			
RxR	110.5	25.79	1.95	87.00	80.97	4.79	-	-
Rx2R	54.4	25.84	2.07	86.50	79.69	4.72	4.81	4.94



(a) Instruction Length (b) Trajectory Length (c) Trajectory Length vs. Instruction Length

Fig. 3. Distribution of R2R, Rx2R and RxR route-instruction pairs.

Of these route-instruction pairs, 53,038 pairs are allocated for training. In comparison, the R2R [6] dataset contains 14,039 training pairs, while RxR includes 26,464 English-language pairs. It can be seen in Fig. 3(c) that the distribution of Rx2R instructions closely resembles that of R2R, but with a greater variety of trajectory samples. For Rx2R, the average instruction length per trajectory is 3.9 words per meter, which is lower than the 8.1 words per meter in RxR but still higher than the 3.0 words per meter in R2R. This difference between Rx2R and R2R can be attributed to the different preference for landmark selection, as it would often take more instances and words to specify a route when using detailed objects (for example, "glass partition", "kitchen counter") as in RxR and Rx2R, than when using regional descriptions (for example, "bathroom", "hallway") as in R2R.

The Rx2R dataset plays a crucial role in speaker training by: (a) significantly expanding the volume of training data, and (b) addressing the challenge of grounding the diverse and free-form RxR instructions to routes through using an LLM to distill essential information from instructions.

Additional VL-based Augmentation. As LLM-IA generates additional high-quality training data, it also increases the possibility of the speaker overfitting to the training scenes. To tackle this issue, we follow previous work [13, 14, 32, 35, 37] and freeze the parameters of the pre-trained image encoder during all training. However, the significant amount of augmented data makes maintaining generalization ability even more difficult. As a result, except for the aforementioned VLN tasks, an additional Vision-Language (VL) task is also incorporated to prevent the model from over-fitting to Matterport3D environments. Specifically, we use the image captioning task and the COCO Captions dataset [12].

In summary, our data augmentation method includes: (a) training on a combined dataset from four VLN tasks, (b) expanding the training dataset with LLM-augmented instructions, and (c) incorporating a VL task to enhance the generalization capability of the image feature extractor.

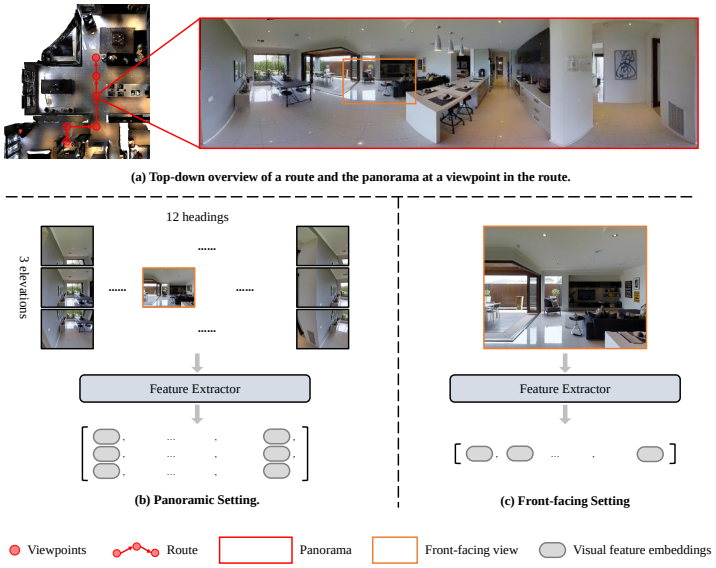


Fig. 4. Illustration of the panoramic setting and the front-facing setting. In the panoramic setting, the speaker captures a panoramic view consisting of 12 headings × 3 elevations RGB images. Then the visual feature extractor generates a feature embedding for each images, resulting in $K = 36$ embeddings for each panorama. In the front-facing setting, the speaker only takes the front-facing view at each viewpoint, and then extracts a sequence of visual patch features for the front-facing image, thus allowing for more detailed visual information.

3.3 Route Representation with Front-Facing Setting

In traditional panoramic settings, speakers generate the instruction from a route consisting of a series of panoramic observations. For a route of T steps, the visual observation $\mathbf{O}_t^{pano}$ at each step $t \in [1, T]$ consists of K discrete RGB images covering K different view angles, which can be given as,

$$\mathbf{O}_t^{pano} = (\mathbf{o}_t^1, \mathbf{o}_t^2, \ldots, \mathbf{o}_t^K), \tag{3}$$

where $\mathbf{o}_t^k$ denotes the the RGB image for view angle $k \in [1, K]$ at step t. Additionally, speakers are provided with an action a_t^{pano} from a panoramic action space $\mathcal{A}^{pano}$, given as,

$$a_t^{pano} \in \mathcal{A}^{pano} = \{\text{view}^1, \text{view}^2, \ldots, \text{view}^K, \text{Stop}\}, \tag{4}$$

where view^k stands for the k-th view angle (relative to the agent's current pose), and Stop represents the completion of the episode. Therefore, the complete route $\mathbf{R}_T^{pano}$ can be denoted by,

$$\mathbf{R}_{1:T}^{pano} = (\mathbf{O}_1^{pano}, a_1^{pano}, \ldots, \mathbf{O}_{T-1}^{pano}, a_{T-1}^{pano}, \mathbf{O}_T^{pano}, a_T^{pano}), \tag{5}$$

In most implementations, K is set as 36, covering 12 headings × 3 elevations. The speakers then generate a navigation instruction $\mathbf{Y}$ from the route information $\mathbf{R}_{1:T}^{pano}$.

While this setting facilitates spatial reasoning for speakers by providing panoramic observations at each step of the route, it has several main drawbacks. First, it imposes strict requirements on input data, limiting its applicability to sources such as videos, where panoramas cannot be obtained. Second, extracting features from all images in a panorama results in long sequences. A common approach to mitigate this is to represent each image with shorter feature embedding, which restricts the amount of detailed information that can be extracted from the visual input. Such an approach

hinders performance in tasks requiring detailed landmarks or precise environmental descriptions. Third, speakers adopting this setting cannot accommodate the need to specify the target object when generating instructions for some VLN tasks.

Nevertheless, it is important to note that the panoramic setting is not a prerequisite for instruction generation due to several factors: a) the distance between most steps is less than three meters, ensuring sufficient visual overlap, and b) the speaker receives information about the action taken at each step, aiding in understanding its movements. In fact, human annotators can understand a route using only front-facing visual observations. Given advancements in machine learning and the strong reasoning capabilities of modern large-scale pre-trained models, it appears feasible to generate navigation instructions in a human-like manner without relying on panoramic observations.

As a result, for application in panorama-unavailable scenarios, we propose to generate instructions only from front-facing images, as shown in Figure 4. In this front-facing setting, the route representations for different types of VLN tasks are as follows.

Fine-grained Instructions. The instructions in R2R [6] and RxR [21] provide step-by-step instructions that guide the agent to a destination. To generate instructions for these two navigation tasks, the speaker needs the information of the complete T-step route $\mathbf{R}_{1:T}$.

Denoting the front-facing observation at time step t by $\mathbf{o}_t^1$, the route can be given as,

$$\mathbf{R}_{1:T} = (\mathbf{o}_1^1, a_1, \ldots, \mathbf{o}_{T-1}^1, a_{T-1}, \mathbf{o}_T^1, a_T), \tag{6}$$

where a_t is the corresponding high-level action at step t, which is given as,

$$a_t \in \mathcal{A} = \{\text{Forward, Left, Right, Up, Down, Stop}\}, \tag{7}$$

Coarse-grained Instructions. REVERIE [27] and SOON [44] provide high-level instructions by specifying the target object and location. Given the target-oriented characteristic of these instructions, the speaker doesn't need the entire route but only the sub-route $\mathbf{R}_{S:T}$ from step $S \in [1, T)$ to step T. Also, these coarse-grained tasks require locating the target object within the final destination, and thus, we also provide the model with the object's class *</object/>*, making the generation process more alike to human annotation process. Therefore, the standard input for these tasks is given as,

$$(\mathbf{R}_{S:T}, \textit{</object/>}) = ((\mathbf{o}_S^1, a_S, \ldots, \mathbf{o}_T^1, a_T), \textit{</object/>}). \tag{8}$$

Additionally, these coarse-grained instructions often include detailed descriptions of the target and destination, thus requiring the speaker to locate the target object, which may not always be visible in the front-facing view. Therefore, for better training efficacy, only during training, the speaker is allowed to capture a surround view $\mathbf{O}_T^{surr} = (\mathbf{o}_T^1, \ldots, \mathbf{o}_t^P)$ of 4 to 6 images upon arrival to locate the target object in the environment. It is worth noting that the surround view is only leveraged during training for efficacy, and during inference, our model only needs front-facing images. The sub-route with surround view $\mathbf{R}_{S:T}^{coarse}$ is given as,

$$\mathbf{R}_{S:T}^{coarse} = (\mathbf{o}_S^1, a_S, \ldots, \mathbf{o}_{T-1}^1, a_{T-1}, \mathbf{O}_T^{surr}, a_T). \tag{9}$$

For better generalization when generating coarse-grained instructions, during training, we randomly select: (a) the value of S, (b) whether the model receives the surround view at the destination, and (c) whether the agent receives *</object/>*.

As a result, the route representation can take any one of the following four combinations: $\{\mathbf{R}_{S:T}, (\mathbf{R}_{S:T}, \textit{</object/>}), \mathbf{R}_{S:T}^{coarse}, (\mathbf{R}_{S:T}^{coarse}, \textit{</object/>})\}$.

3.4 VL-Sp: VLM-Based Speaker Architecture

Inspired by the powerful reasoning and understanding ability of current large-scale multi-modal models, we propose VL-Sp, an architecture that leverages a pre-trained VLM to build a speaker

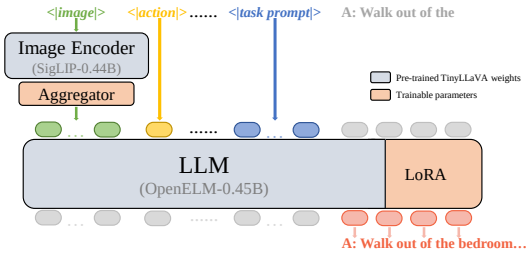


Fig. 5. Model architecture: VL-Sp. VL-Sp leverages a pre-trained VLM (TinyLLaVA [41]) and incorporates an aggregator to condense image features. The base LLM is fine-tuned in a parameter-efficient manner using LoRA [19]. All parameters in the original VLM are frozen, and only the additional parameters from the aggregator and LoRA are trainable.

capable of generating instructions from a series of single images in a human-like fashion. Specifically, our proposed architecture is based on a multi-modal TinyLLaVA [41] model, as illustrated in Fig. 5. Such an architecture allows us to make full use of the pre-trained VLM's reasoning ability to reason about the spatial location and movements in the routes without requiring panoramas.

Another benefit of exploiting the VLM-based architecture is that it enables CaneSpeaker to exploit the textual comprehension ability of the VLM to generate instructions for various tasks based on different prompts, thereby enabling training on multiple VLN tasks and paving the way for the data augmentation method in Sec. 3.2.

To enable the speaker to attend to the inputs of both fine- and coarse-grained tasks, we formulate a unified input representation consisting of two parts: a) the route information introduced in Sec. 3.3, b) and a task-specific prompt Prompt^{task} tailored to each task to enable training on multiple VLN tasks.

Under this setting, an example input $(\mathbf{R}, \text{Prompt}^{\text{REVERIE}}(<|object|>))$ for the REVERIE task is as follows,

> -$\mathbf{R}_{S:T}$, *Provide directions to the room and identify the <|object|>.*
> *-[generated instruction].*

Also, thanks to the cross-task nature of the input structure, we can effortlessly incorporate the additional VL-based augmentation introduced in Section. 3.2 in our model by adjusting the input prompt structure. For an image-caption pair, $(\mathbf{o}^{caption}, \textit{[caption]})$, the speaker is required to generate the caption in the following manner:

> -$\mathbf{o}^{caption}$, *Describe the above image in one sentence.*
> *-[caption].*

In such inputs, the prompt is directly tokenized and turned into embeddings as below,

$$\begin{aligned}(tok_1^{task}, \ldots, tok_L^{task}) &= \textbf{Tokenizer}(\text{Prompt}^{task}),\\ \mathbf{E}^{task} = (\mathbf{e}_1^{task}, \ldots, \mathbf{e}_L^{task}) &= \textbf{Emb}(tok_1^{task}, \ldots, tok_L^{task}),\end{aligned} \tag{10}$$

where **Tokenizer** and **Emb** are the tokenizer and embedding layer of the base LLM, respectively, L is the length of the sequence of the tokenized prompt, tok_l^{task} is the l-th token in the tokenized sequence, and $\mathbf{e}_l^{task}$ is the corresponding token embedding of tok_l^{task}.

Also, the actions in the route are processed similarly as follows,

$$\begin{aligned}tok_t^{action} &= \textbf{Tokenizer}(a_t),\\ \mathbf{e}_t^{action} &= \textbf{Emb}(tok_t^{action}),\end{aligned} \tag{11}$$

where tok_t^{action} is the tokenized result of action a_t, and $\mathbf{e}_t^{action}$ is the token embedding of tok_t^{action}.

And for the observation images, it is not practical to directly use the extracted features, since VLN routes involve multiple images and directly incorporating them would produce overly lengthy context. Therefore, a learnable feature aggregation layer is introduced to condense the image features. The feature extraction is performed as follows,

$$\begin{aligned} \mathbf{E}^{*image}_t &= \textbf{Image\_encoder}(\mathbf{o}_t^1) \in \mathbb{R}^{M\times D}, \\ \mathbf{E}_t^{image} = (\mathbf{e}_{t,1}^{image}, \ldots, \mathbf{e}_{t,N}^{image}) &= \textbf{Aggregator}(\mathbf{E}^{*image}_t) \in \mathbb{R}^{N\times D}, \end{aligned} \tag{12}$$

where D is the dimension of the image features extracted by the image encoder (after the projector), the same as the dimension of the text embedding of the LLM in TinyLLaVA, M is the length of the original sequence of visual patch features $\mathbf{E}^{*image}_t$, and N is the length of the aggregated sequence $\mathbf{E}_t^{image}$.

These embeddings are then processed by the base LLM to generate a response in an autoregressive manner as follows,

$$\begin{aligned} &P(y_s \mid \mathbf{E}_S^{image}, \mathbf{e}_S^{action}, \ldots, \mathbf{E}_T^{image}, \mathbf{e}_T^{action}, \mathbf{E}^{task}, y_{<s}) \\ &= \textbf{LLM}(\mathbf{E}_S^{image}, \mathbf{e}_S^{action}, \ldots, \mathbf{E}_T^{image}, \mathbf{e}_T^{action}, \mathbf{E}^{task}, y_{<s}), \end{aligned} \tag{13}$$

where y_s represents the s-th token in the generated instruction, $y_{<s}$ denotes the previously generated tokens, and $P(y_s \mid \mathbf{E}_S^{image}, \mathbf{e}_S^{action}, \ldots, \mathbf{E}_T^{image}, \mathbf{e}_T^{action}, \mathbf{E}^{task}, y_{<s})$ represents the probability assigned to the next token by the LLM.

CaneSpeaker is trained to minimize the negative log-likelihood of the reference instructions autoregressively. For parameter-efficient updating, the base LLM in the VLM is fine-tuned using LoRA [19]. The optimization objective can be expressed as,

$$\min_{\theta^*} -\sum_{s=1}^{S} \log P(y_s \mid \mathbf{E}_S^{image}, \mathbf{e}_S^{action}, \ldots, \mathbf{E}_T^{image}, \mathbf{e}_T^{action}, \mathbf{E}^{task}, y_{<s}; \theta \cup \theta^*), \tag{14}$$

where θ represents the frozen parameters of the base LLM, and θ^* represents the additional parameters for parameter-efficient updating.

3.5 CANE: A Large-Scale Cross-Task Augmented Datasets

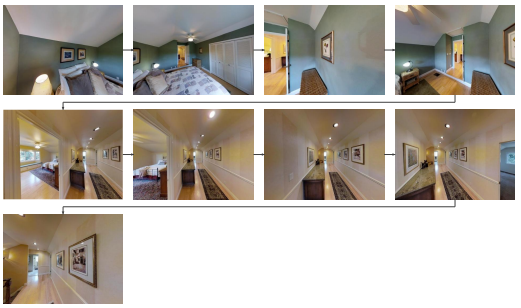
CANE To better evaluate the impact of the instructions generated by CaneSpeaker on VLN agent training, a cross-task dataset, CANE, is also synthesized by adopting CaneSpeaker. CANE leverages 178k unannotated routes from the Matterport3D Simulator [7], similar to PREV [18]. To generate fine-grained instructions, CaneSpeaker is leveraged to generate six R2R-style, six Rx2R-style, and five RxR-style instructions for each unannotated route. To generate coarse-grained instructions, we first randomly select a visible object at the destination using Matterport3D's semantic annotations. Then the route information, along with the object category, is processed by CaneSpeaker to generate REVERIE instructions. Since the REVERIE instructions are relatively concise and only include destination and target information, we only generate one instruction per route.

Table 2 summarizes the basic statistics of the proposed CANE dataset, including the number of instructions, the average instruction length, and the total token count. Compared to the widely-used PREV dataset, which focuses solely on the R2R task, CANE covers four distinct navigation tasks. Overall, CANE contains more than four times the total number of tokens compared to PREV.

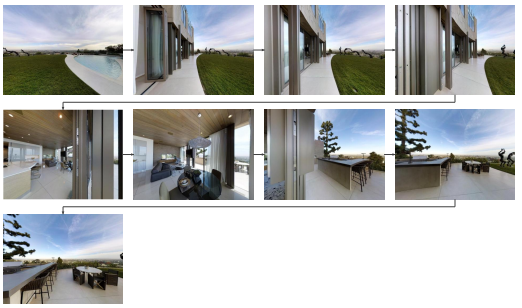
Additionally, examples of CANE routes and the corresponding synthesized instructions are presented in Figure 6. It can be seen that while the synthesized instructions inevitably contain some errors, due to imperfections in instruction generation, such as incorrect landmark selection,

Table 2. Comparison of Prev [18] and CANE. "Num. Instr." stands for number of instructions per route, "Avg. Len." stands for the average length of the instructions, and "Total Tok." stands for the total number of tokens in the dataset.

Datasets	R2R		Rx2R		RxR		REVERIE		Total Tok.
	Num. Instr.	Avg. Len.	Num. Instr.	Avg. Len.	Num. Instr.	Avg. Len.	Num. Instr.	Avg. Len.	
PREV [18]	6	24.05	-	-	-	-	-	-	25,729k
CANE	6	24.99	6	32.14	5	62.40	1	17.68	119,851k



R2R	Walk past the bed and out the door. Turn left and go into the hallway. Stop near the wall with three pictures. Exit the bedroom, turn left, then turn right and wait near the stairs.
Rx2R	Walk out of the room and turn right to exit. Move to the brown cupboard, then turn left and exit through the door. Take a slight right and proceed straight, stopping at the photo frames on the wall. Walk out of the room and turn right to the wall. Turn left and move forward to the entrance. Take a right and proceed straight to the photo frames. Stop at the first photo frame.
RxR	You are in the bedroom, facing towards the bed, turn right and walk straight, you can see a door which is open, walk through the door and move towards the carpet, turn right and move forward, you can see an open door, walk towards the open door and stand in front of the portrait on the wall, you have reached your destination. You are facing towards the wall, turn slightly right and move forward. Turn slightly left and exit the room through the open door. You are facing towards an open door, turn slightly right. You can see a narrow passage in front of you, move towards the passage. You are facing a photo frame, turn slightly right, move towards the photo frame. You are facing the photo frame, which is your final destination.
REVERIE (Obj: Photo)	Go to the hallway on level 2 near the stairs and take down the photo on the wall to the right of the stairs.



R2R	Go left past the open door and walk straight until you reach the bar. Go straight, turn left and go past the pool. Then go past the dining room table and stop in the doorway to the outside. "
Rx2R	Turn slightly left and move forward to the open glass door. Enter the room and walk straight to the dining table. Turn right and move to the chairs. Stop in front of the chairs. Walk through the open door to the left, then proceed to the glass door. Exit the room and head towards the round dining table. Turn right and walk to the three bar stools.
RxR	You're facing towards the pool, turn left and move forward, you'll find a glass door, move forward and enter into the room, turn right and you can find two chairs, move forward, you can find a dining table on your right side, stand in between the two chairs, this is your destination. Right now you're facing towards the green grass. Now turn left and move towards the glass door. Now slightly turn left and move forward. Now you can see an open door to your left. Enter into the room. Now turn right and move towards the dining table. Now you're facing a dining table and that is the end point.
REVERIE (Obj: Chair)	Go to the outdoor dining area and pull out the chair closest to the door.

Fig. 6. Examples of CANE routes and instructions.

or orientation mismatches, they still capture the key landmarks and maintain reasonable spatial consistency.

Different Characteristics and Positioning of Rx2R and CANE. The relationship between Rx2R and CANE reflects their complementary roles in training both VLN speakers and VLN agents.

Rx2R is a high-quality augmented dataset with human-level accuracy, primarily designed for speaker training. It is generated by leveraging an LLM to augment existing free-form, multilingual instructions across different VLN tasks. Such a generation method allows it serve as a high-quality augmented dataset with human-level accuracy and can be used for speaker training. This high quality along with the distinct characteristics from existing datasets (in terms of landmark selection and language formulation) also allows it to be used as a standalone evaluation dataset.

CANE is a large-scale, cross-task dataset generated using CaneSpeaker, suited for agent training. It is generated by first collecting previously unannotated routes and then using a speaker to generate instructions. Compared to Rx2R, it inevitably contains some errors, which arise from imperfections

Table 3. Comparison of the characteristics and positioning of Rx2R and CANE. "Num. Tasks" stands for the number of VLN tasks covered in the dataset, and "Total Instr." stands for the total number of instructions in the dataset.

Datasets	Characteristic			Function			
	Num. Tasks	Total Instr.	Accuracy	Speaker Training	Agent Pre-training	Agent Fine-tuning	Stanalone Task
Rx2R	1	68k	Human-level	✓	✓	✓	✓
CANE	4	3,207k	Noisy	×	✓	×	×

in instruction generation, incorrect landmark selection, or orientation inconsistencies. However, due to its scale and visual diversity, it is fit for agent pre-training.

The characteristics and positioning of both datasets are summarized in Table 3.

4 Experiments

4.1 Datasets

In the following experiments, six existing datasets were leveraged:

- **R2R** [6] contains approximately 22k instructions for routes within the Matterport3D Simulator, each paired with step-by-step navigation paths. The instructions are concise, with an average length of 29 words, and are designed to guide agents along predefined shortest-path trajectories.
- **RxR** [21] features diverse, free-form instructions in three languages: English, Hindi, and Telugu. These multilingual instructions are intended to reduce path biases, emphasize visible entities, and enhance the role of language in navigation. In the following experiments, unless otherwise mentioned, only the English instructions in RxR were used.
- **REVERIE** [27] provides 22k high-level instructions with an average length of 18 words. These instructions primarily describe target locations and objects, requiring agents to navigate without detailed guidance and select the correct object bounding box at the destination.
- **SOON** [44] offers longer, more detailed high-level instructions, with an average length of 47 words. Unlike REVERIE, SOON does not include ground-truth bounding boxes. Instead, agents are tasked with predicting the center location of the target object within the panorama.
- **PREV** [18] (short for Prevalent) is an existing augmented dataset that enhances the original R2R instructions. It aims to improve the generalization and robustness of VLN models by generating instructions for unannotated routes in the Matterport3D Simulator.
- **COCO Captions** [12] is a large-scale image captioning dataset containing over 300k images, designed to facilitate tasks such as image captioning.

Except for the above datasets, our proposed datasets, **Rx2R** and **CANE**, were also utilized.

4.2 Evaluation Metrics

Instruction Generation. In line with previous studies, the generated instructions were evaluated using the following five metrics:

- **SPICE** measures semantic accuracy by utilizing semantic graphs. It served as the primary evaluation metric for instruction generation, as it is the only metric that demonstrates a strong correlation with human way-finding performance [39].

Table 4. Instruction generation results on tasks with fine-grained instructions: R2R and RxR. SPICE serves as the primary metric. Best in bold.

Methods	R2R val_seen					R2R val_unseen					Fluency
	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	
BT-speaker [14]	0.203	0.121	0.233	0.350	0.155	0.188	0.114	0.228	0.346	0.142	4.06
EDrop-speaker [32]	0.202	0.493	0.228	0.467	0.245	0.181	0.422	0.225	0.458	0.237	4.34
VLS [3]	0.214	0.137	0.228	0.352	0.157	0.197	0.132	0.231	0.357	0.159	-
CCC-speaker [35]	0.231	0.543	0.236	0.493	0.287	0.214	0.461	0.231	0.477	0.272	-
LANA-speaker [37]	0.256	0.533	0.245	**0.503**	**0.314**	0.226	0.457	0.238	**0.498**	0.298	-
BEVInstructor [13]	0.220	0.549	0.238	0.480	0.285	0.208	0.449	0.230	0.467	0.264	-
CaneSpeaker (Ours)	**0.294**	**0.562**	**0.254**	0.502	**0.314**	**0.266**	**0.469**	**0.245**	0.496	**0.300**	**4.91**
Methods	**RxR** val_seen					**RxR** val_unseen					Fluency
	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	
BT-speaker [14]	0.157	0.175	0.184	0.314	0.130	0.149	0.160	0.184	0.314	0.129	2.51
EDrop-speaker [32]	0.154	0.180	0.176	0.309	0.130	0.145	0.170	0.178	0.312	0.136	3.19
Marky [36]	0.178	0.350	-	-	-	0.164	0.317	-	-	-	-
CaneSpeaker (Ours)	**0.219**	**0.462**	**0.251**	**0.412**	**0.266**	**0.220**	**0.437**	**0.248**	**0.413**	**0.267**	**4.14**

- **CIDEr** measures consensus between generated and reference instructions based on term frequency using TF-IDF weighting.
- **METEOR** computes the semantic similarity between generated and reference instructions, accounting for synonymy, stemming, and word order.
- **ROUGE** assesses recall-oriented similarity between generated and reference instructions.
- **BLEU** evaluates the precision of n-gram overlap in the generated and reference instructions.

Way-finding. For the evaluation of navigation performance, the following metrics were adopted:

- **Trajectory Length (TL)** is the total length of the path traveled during the navigation task.
- **Navigation Error (NE)** is the Euclidean distance between the final position and destination.
- **Success Rate (SR)** is the proportion of successful navigation episodes, where a successful episode is defined as reaching the destination within 3 meters. This metric was selected as the primary evaluation criterion.
- **Success Penalized by Path Length (SPL)** adjusts the success rate by incorporating the length of the path taken relative to the length of the ground-truth trajectory.
- **Oracle Success Rate (OSR)** measures success rate when given an oracle stop policy.

4.3 Implementation Details

Rx2R. For generating Rx2R dataset, a widely-used LLM, LLaMA3 8B [16], was utilized. Since LLaMA3 is multilingual, the English (en-US, en-IN) and Hindi (hi-IN) instructions from RxR were used directly. The dataset generation took approximately 16 hours on two NVIDIA L40 GPUs. The collected dataset contains 67,977 instructions for 12,591 paths.

Speaker. The speaker model uses the pre-trained weights of a 0.89B TinyLLaVA [41] model. The visual feature aggregator is implemented as an MLP with $M = 728$ and $N = 14$. The training of the model was done in two stages, first on the image captioning task for fast convergence, and then on a combination of all tasks. Both stages were conducted on one single NVIDIA Tesla A100 GPU.

4.4 Experiments on Instruction Generation

CaneSpeaker was compared against several baselines: BT-speaker [14], EDrop-speaker [32], VLS [3], CCC-speaker [35], LANA-speaker [37], BEVInstructor [13], Marky [36], and AIGeN [30]. These methods were evaluated across four different VLN tasks: two tasks with fine-grained instructions, R2R and RxR, and two with coarse-grained instructions, REVERIE and SOON. Additionally, an LLM [16] was employed to rate the fluency of the generated instructions from 1 to 5.

Table 5. Instruction generation results on tasks with coarse-grained instructions: REVERIE and SOON. SPICE serves as the primary metric. Best in bold.

Methods	**REVERIE** val_seen					**REVERIE** val_unseen					Fluency
	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	**SPICE**↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	
BT-speaker [14]	0.108	0.336	0.261	0.564	0.318	0.082	0.162	0.213	0.508	0.239	3.75
EDrop-speaker [32]	0.117	0.468	0.262	0.611	0.409	0.106	0.363	0.241	0.578	0.198	4.50
BEVInstructor [13]	0.208	0.745	**0.324**	0.635	**0.425**	0.159	0.489	0.267	0.560	0.335	-
AIGeN [30]	0.329	0.890	0.228	0.465	-	0.228	0.486	0.179	0.393	-	-
CaneSpeaker (Ours)	**0.346**	**1.540**	0.318	**0.658**	0.398	**0.328**	**1.267**	**0.280**	**0.605**	**0.369**	**4.98**
Methods	**SOON** val_seen					**SOON** val_unseen					Fluency
	SPICE↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	**SPICE**↑	CIDEr↑	Meteor↑	Rouge↑	Bleu-4↑	
BT-speaker [14]	0.115	0.246	0.162	**0.343**	**0.126**	0.121	0.220	0.182	0.344	**0.148**	2.86
EDrop-speaker [32]	0.130	0.259	0.160	0.332	0.119	0.116	0.178	0.173	0.342	0.131	3.27
CaneSpeaker (Ours)	**0.224**	**0.541**	**0.174**	0.318	0.101	**0.232**	**0.573**	**0.184**	**0.361**	0.118	**4.11**

According to the results in Table 4 and Table 5, CaneSpeaker achieves significantly superior performance compared to existing methods, particularly in unseen environments. Notably, CaneSpeaker attains an average improvement of 0.070 on SPICE on the val_unseen splits, highlighting its ability to generate instructions that closely align with human-annotated references in terms of semantic fidelity. This strong performance on SPICE, especially in unseen environments, coupled with its high Fluency score, indicates that CaneSpeaker produces natural-sounding instructions that not only contain accurate semantic information but also generalize effectively to previously unseen scenarios.

Furthermore, CaneSpeaker outperforms other methods on CIDEr, which emphasizes term frequency based relevance, while its performance on BLEU and ROUGE is sometimes comparatively lower. This suggests that CaneSpeaker is adept at selecting appropriate lexical choices across different navigation routes, even though it may not consistently match reference instructions at an exact n-gram level. Such discrepancies often arise from minor linguistic variations (for example, "move to" and "go to the"), which have minimal semantic importance but can influence n-gram based metrics. This can be attributed to the extensive training data used in our method, which enables our speaker to learn from a large corpus of text and generate diverse expressions. Although such a large amount of training data may reduce performance on metrics that rely on n-gram overlap, it enhances the diversity of generated instructions, which is crucial for training VLN agents.

These findings underscore the effectiveness of the pre-trained VLM based architecture and the large-scale training data employed in our approach. By leveraging both, CaneSpeaker is able to generate navigation instructions that are not only accurate and semantically rich but also lexically diverse, in other words, human-like.

4.5 Qualitative Experiments

Qualitative Results of LLM-IA. The qualitative examples of the collected Rx2R instructions are presented. The Rx2R instructions are derived from the English (en-US and en-IN) and Hindi (hi-IN) instructions in RxR. Table 6 presents examples for each of the en-US, en-IN, and hi-IN instruction pairs.

As illustrated by the examples of RxR and their corresponding Rx2R instructions, it is clear that RxR instructions are more free-form and sometimes contain significant redundancy, in the form of unnecessary storytelling or corrections of previous statements. In contrast, the generated Rx2R instructions effectively capture the essential route information and are more concise.

Qualitative Results of CaneSpeaker. Qualitative examples illustrating how our speaker model generates varied instructions for different tasks in comparison to previous methods are presented.

Table 6. Qualitative examples of Rx2R instructions along with the original RxR instructions.

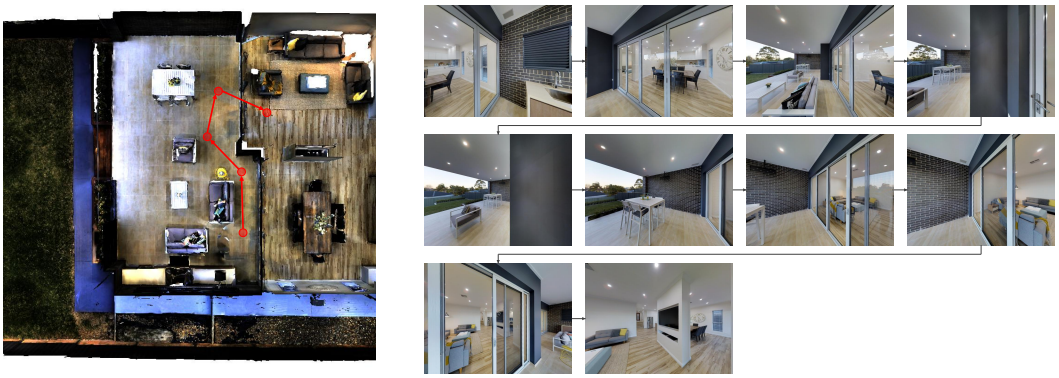
Source language	RxR	Rx2R
en-US	You begin in what I hope is a room under construction, and not one that has simply been stripped bare by robbers, but y'never know. Could be a good burgling. You could have taken part in it. Maybe you're on the lam,running from the law. Who knows? Turn all the way around and exit through the door you've- you've been standing in. Just move straight down this hallway, into the one actually carpeted room that seems to be spreading its carpet out nicely, slowly growing towards the cemented floors that you've just walked across. You should see four armchairs that are lined up to look at the nothing that is currently mounted on the walls. The nothing with surround sound, it looks like! Yeah, speakers... Speakers up front and in the back. That is some good quality nothing. Anyways, continue to the very center of this room so that you feel like you are standing trial in front of the four slouchy armchairs, and once you are there, pleading your case to the armchairs and hoping that they show pity upon your soul for all the burgling you did, you are done.	Walk out of the room, move down the hallway, and enter the carpeted room. Proceed to the center of the room where four armchairs are lined up, and stop in front of them.
en-IN	You are currently facing towards a bathtub which is in the centre of a room, turn to your left, take a few steps forward, now turn to your right and you can see a couple of steps, climb them and move towards the white bathtub on the left. Now you can see a couple of more stairs, climb up. Move a few steps forward and you can see a white bathtub on your left and you are facing a glass window, that is your end point.	Walk to the left of the bathtub and move forward a few steps. Turn right and climb the couple of steps. Head towards the white bathtub on the left and climb the couple of stairs. Move forward and stop in front of the white bathtub near the glass window.
hi-IN	अभी सामने दिवार है. हल्का बाए होकर बिलकुल सामने कमान में जाए. दाहिने मुड़े दरवाज़े तक जाए. हल्का दाहिने होकर दरवाज़े के अंदर चित्र के पास रुक जाए. अभी उलटा घूमने पर दाहिने तरफ आइना है. बहार दीपक और सोफा है.	Turn left and walk to the center. Turn right, go to the door and enter. Move right, stop in front of the painting. Turn around and face the right to see the mirror.

The instructions generated by our model were compared with those from human annotators and two most widely-adopted speaker models (BT-speaker [14] and EDrop-speaker [32]) in Fig. 7, and color coding were employed to highlight specific types of errors.

According to the results, the instructions generated by our approach exhibit fewer errors in guiding information, such as incorrect orientation and landmarks, compared to previous methods. Additionally, our approach generates instructions with less ambiguity, reducing the likelihood of misinterpretation. Furthermore, the instructions produced by our model sound more natural and fluent, similar to human instructions, while also incorporating a greater variety of expressions, which aids the agents in better understanding diverse instructions when used for training. These comparisons highlight that the instructions generated by our model are not only more accurate but also more human-like, underscoring the effectiveness of our approach. Also, it is important to note that CaneSpeaker uses a single model for all tasks, whereas BT-speaker and EDrop-speaker use separate models for each task.

4.6 Evaluation of the Augmented Datasets

Assessment of Rx2R. Experiments were conducted to evaluate the accuracy of the generated Rx2R instructions. To this end, four baseline VLN agents were selected: EDrop-follower [32], Hamt [10], Duet [11], and Duet+ScaleVLN [38], and compared their navigation performance when guided by the augmented instructions versus when guided by human-annotated instructions. To ensure a fair



(a) Top-down overview and visual observations.

Task	Speaker	Instruction
R2R	Human [6]	Head inside through the sliding glass doors on the other end of the deck. Stop in the living room and wait near the arm chair.
		Go straight until you pass the couches and the table then turn right and go inside. Wait near the grey couches.
		With the sink behind you, walk the length of the patio to the table with the high stools. Turn right and walk inside through the closest open door. Stop once stepping inside.
	BT-speaker [14]	turn around and walk towards the couch . turn right and walk past the couch . turn right and walk past the couch . turn right and wait near the white chair .
	EDrop-speaker [32]	walk past the couch and turn right . walk into the house and turn right . stop in front of the couch .
	CaneSpeaker (Ours)	Turn around and walk to the other side of the patio. Once you reach the other side, turn right and enter the house. Stop once you enter the house.
RxR	BT-speaker [14]	you begin facing a door , turn to your right , take a step forward and then turn to your right , you will see a dining table , take a step towards the left of the dining table , and then take a step towards the white couch in front of you , and then take a step towards the white couch in front of you , turn to your right , you will see a dining table , take a step towards the left of the dining table , you will see a glass railing to the left of the staircase , take a step to go down the staircase , you are done .
	EDrop-speaker [32]	you ' re standing in front of a glass door , turn around and you ' ll see a doorway to the left of the tv . walk through this doorway . and you ' ll see a living room , walk to the left of this white couch in front of you . and then turn around and walk towards the white couch that ' s on the right and you ' re done .
	CaneSpeaker (Ours)	You're standing near the entrance of the house, facing towards the glass door, turn around, you can see the two sofas, walk towards the back of the sofas , and take a right, you can see a door in front of you, enter into the room through the door, you can see your destination in front of you.
REVERIE	BT-speaker [14]	go to the living room and bring me the blue pillow on the table closest to the door
	EDrop-speaker [32]	go to the bedroom with the white walls and bring me the picture on the wall
	CaneSpeaker (Ours)	Go to the living room and bring me the first pillow on the sofa.
SOON	BT-speaker [14]	please help me find a cloth and white chair , which is in front of a painting . it ' s in a living room on the first floor . the balcony is near a bedroom , near a living room and a stair to a corridor .
	EDrop-speaker [32]	i want to find a white , rectangular and white lamp , which is set in the bright bedroom . the sofa is on the table , next to the door and next to the door . the living room is on the second floor and next to the living room .
	CaneSpeaker (Ours)	I want to find a sofa, which is white , small and rectangular. The sofa is next to the door, in front of the television and next to the window . It is in a bright living room on the first floor.

(b) The instructions for the above route generated by different speakers. The colors are used to highlight different types of errors: wrong orientation , wrong landmark , wrong/missing action , ambiguity , and hallucination .

Fig. 7. Qualitative example of instruction generation for a route from the val_unseen split of the R2R dataset in Matterport3D Simulator.

comparison, the English route-instruction pairs from RxR that had similar path lengths to those in R2R were selected and the corresponding pairs from Rx2R were sampled.

Table 7. Navigation guidance capability of Rx2R, compared to R2R and RxR. SR serves as the primary metric.

Methods	Trained on	Val on	val_seen				val_unseen			
			TL	NE ↓	**SR ↑**	SPL ↑	TL	NE ↓	**SR ↑**	SPL ↑
EDrop-follower [32]	R2R	R2R	11.00	3.99	62.10	59.00	10.70	5.22	52.20	48.00
		Rx2R	9.70	5.42	45.33	42.40	9.72	5.49	42.93	38.61
		RxR	9.78	6.04	42.67	40.11	9.71	5.73	40.60	36.61
Hamt [10]	R2R	R2R	11.16	2.52	75.51	72.06	11.46	3.62	66.28	61.55
		Rx2R	12.92	5.03	54.50	50.81	13.05	4.91	50.48	45.99
		RxR	11.75	5.36	47.83	45.12	12.39	5.24	44.21	40.56
Duet [11]	R2R	R2R	12.33	2.28	78.84	72.88	13.94	3.31	71.52	60.41
		Rx2R	16.36	4.91	54.17	45.72	16.66	5.09	50.80	40.55
		RxR	15.78	5.14	54.00	45.35	16.38	5.42	49.20	38.75
Duet+ScaleVLN [38]	R2R	R2R	12.88	2.13	80.80	74.58	13.23	2.38	79.40	69.97
		Rx2R	17.34	3.98	61.00	51.27	16.53	4.13	59.62	49.60
		RxR	16.98	4.60	58.50	50.41	16.37	4.46	57.07	47.48

The way-finding performances of four R2R baseline agents [10, 11, 32, 38] on R2R, RxR, and Rx2R were evaluated, as presented in Table 7. The SR for these agents increases by 3.09% on average when using Rx2R instructions compared to using the original RxR instructions, suggesting that leveraging an LLM to extract and distill key information from RxR instructions before presenting them to R2R agents enhances the agents' performance, thus validating the navigation guidance capability of the Rx2R instructions. Still, there remains a gap between the SRs for R2R and Rx2R, which can be attributed to the more varied and complex landmarks referenced in the instructions that present additional challenge to the agents.

Effect of Rx2R and CANE on Agent-Training. To further validate how the proposed datasets affect agent training, experiments comparing the navigation performance of Duet [11] and NavGPT-2 [42] agents trained on Rx2R and CANE with ones trained on PREV [18] and E_PREV (generated using PREV routes and EDrop-speaker) were conducted. We denote the instructions for R2R and Rx2R in CANE by CANE_{R2R} and $\text{CANE}_{\text{Rx2R}}$. In Table 8, Method#1-#8 represent agents trained on different datasets. These agents were validated on: (a) paths from R2R, and (b) paths from Rx2R with R2R-level length, assessing their ability to generalize to instructions with different annotation styles.

As shown in Table 8, the agents trained on Rx2R and CANE evidently demonstrate advanced way-finding capabilities. Firstly, by comparing the agent trained with/without large-scale augmented dataset (Method#1 and Method#4), it can be seen that incorporating large-scale augmented datasets like CANE for agent training can significantly improve navigation performance. This can be attributed to the vast amount of route-instruction pairs that allows the agent to learn the alignment between navigation instructions and visual observations and obtain an initial exploration strategy. Additionally, compared to existing SOTAs, when only using the R2R-style instructions, the agent trained on our datasets (Method#4) outperforms the original Duet agent (Method#2) on the val_unseen split of R2R and Rx2R by 1.02% and 6.36%, respectively. These results demonstrate that the instructions with enhanced semantic consistency and formulation variation in CANE can improve the agent's way-finding and its ability to generalize to diverse instructions. Furthermore, incorporating Rx2R-style instructions from CANE and the Rx2R training set leads to even greater improvements, with Method#8 achieving an increase of 2.34% and 13.48% on SR on R2R and Rx2R compared to the original Duet agent. This highlights the value of the diverse landmarks along with structured languages in Rx2R-style instructions which enable agents to better ground instructions to routes. Finally, as shown in Table 9, the way-finding results of the LLM-based NavGPT-2 agents

Table 8. Way-finding results of the VLN agent Duet [11] trained on different datasets. SR serves as the primary metric. Best in bold.

Method #	Pre-training Dataset						Fine-tuning Dataset		R2R val_unseen				Rx2R val_unseen			
	R2R	Rx2R	PREV	E_PREV	$CANE_{R2R}$	$CANE_{Rx2R}$	R2R	Rx2R	TL	NE ↓	**SR** ↑	SPL ↑	TL	NE ↓	**SR** ↑	SPL ↑
1	✓						✓		12.48	3.73	62.49	53.80	13.75	3.17	41.40	35.16
2	✓		✓				✓		13.94	3.31	71.52	60.41	16.88	5.55	46.48	37.52
3	✓			✓			✓		15.15	3.13	72.07	59.68	18.77	5.36	48.83	39.22
4	✓				✓		✓		15.18	3.12	72.54	60.47	19.69	5.05	52.84	41.85
5	✓	✓			✓	✓	✓		14.13	3.09	72.63	**60.94**	19.31	4.49	56.36	44.86
6	✓	✓					✓	✓	13.49	3.79	66.71	56.84	15.83	5.28	48.33	39.59
7	✓				✓		✓	✓	15.70	2.99	73.01	60.05	17.98	4.35	58.64	47.00
8	✓	✓			✓	✓	✓	✓	16.03	**2.97**	**73.86**	60.69	18.37	**4.11**	**59.96**	**48.38**

Table 9. Way-finding results of the VLN agent NavGPT-2 [42] trained on different datasets. SR serves as the primary metric. Best in bold.

Method	R2R val_seen				R2R val_unseen			
	TL	NE ↓	**SR** ↑	SPL ↑	TL	NE ↓	**SR** ↑	SPL ↑
NavGPT-2	13.53	3.69	69.64	63.77	14.40	3.65	67.22	56.07
NavGPT-2 w/ PREV	13.30	3.11	73.07	67.19	14.58	3.62	68.58	56.95
NavGPT-2 w/ CANE	13.14	**2.97**	**74.73**	**68.36**	13.61	**3.43**	**69.86**	**58.53**

are consistent with the results of Duet, as the agent trained on CANE exhibits improved navigation performance. This further demonstrates that CANE can improve both traditional and LLM-based agents' way-finding ability.

Cross-task Generalization Ability To further validate the cross-task generalization capability of the proposed CANE dataset and CaneSpeaker, we further conducted additional experiments where the navigation abilities of VLN agents trained on different augmented datasets were tested on tasks with fine-grained instructions. Specifically, we selected REVERIE and SOON, two benchmark tasks featuring coarse-grained instructions, and adopted four different augmented datasets for pre-training: (a) PREV [18], (b) $CANE_{Rx2R}$, (c) Original, the original augmentation method used in Duet [11] (an augmented dataset with REVERIE instructions for REVERIE and no augmentation for SOON), and (d) $CANE_{REVERIE}$, the REVERIE-style instructions in CANE.

As shown in Table 10 and Table 11, agents pre-trained on $CANE_{Rx2R}$ outperform those trained on PREV when evaluated on REVERIE and SOON, indicating improved navigation ability. This result corroborates the cross-task generalization potential of $CANE_{Rx2R}$, which can be attributed to the more precise visual descriptions that enhance visual grounding.

However, as shown in Table 10, on the REVERIE task, the $CANE_{Rx2R}$-trained agent still underperforms compared to those trained with task-specific REVERIE instructions, highlighting that task-specific supervision remains optimal. This discrepancy may arise from the different focus of the instructions, which require the agent to learn various navigation strategies. This validates the necessity for a cross-task speaker as the best pre-training strategy is to use task-specific instructions. Furthermore, the comparison between agents trained on Original and $CANE_{REVERIE}$ shows that the latter achieves superior navigation performance, demonstrating that CaneSpeaker can generate not only accurate fine-grained instructions but also effective REVERIE-style instructions, further validating its cross-task instruction generation capability.

4.7 Ablation Studies

In this subsection, the results of ablation studies are reported to verify the effectiveness of each module in our approach. The ablation studies were conducted on RxR, due to its complexity,

Table 10. Way-finding results on REVERIE of the VLN agent Duet [11] trained on different datasets. "Aug. Data" stands for augmented data. SR serves as the primary metric. Best in bold.

Aug. Data	**REVERIE** val_seen				**REVERIE** val_unseen			
	TL	OSR ↑	**SR** ↑	SPL ↑	TL	OSR ↑	**SR** ↑	SPL ↑
PREV	13.40	64.16	61.56	55.67	20.35	44.62	40.16	29.27
$\text{CANE}_{\text{Rx2R}}$	12.30	65.64	62.97	58.07	19.32	48.03	43.17	32.34
Original	14.40	75.19	71.54	61.87	24.70	51.32	45.87	31.47
$\text{CANE}_{\text{REVERIE}}$	14.69	**75.69**	**72.24**	**64.63**	23.96	**53.39**	**47.34**	**33.64**

Table 11. Way-finding results on SOON of the VLN agent Duet [11] trained on different datasets. "Aug. Data" stands for augmented data. SR serves as the primary metric. Best in bold.

Aug. Data	**SOON** val_seen				**SOON** val_unseen			
	TL	OSR ↑	**SR** ↑	SPL ↑	TL	OSR ↑	**SR** ↑	SPL ↑
Original	38.41	44.78	37.96	28.60	36.20	50.91	36.28	22.58
PREV	34.08	51.77	42.48	32.25	29.87	51.33	37.46	27.91
$\text{CANE}_{\text{Rx2R}}$	32.45	**56.02**	**46.55**	**36.40**	30.57	**53.27**	**40.62**	**30.57**

Table 12. Ablation studies. SPICE serves as the primary metric. Best in bold.

Methods	**RxR** val_unseen				**R2R** val_unseen			
	SPICE ↑	CIDEr ↑	Meteor ↑	Bleu-4 ↑	**SPICE** ↑	CIDEr ↑	Meteor ↑	Bleu-4 ↑
CaneSpeaker (w/o CT)	0.179	0.262	0.189	0.159	0.196	0.338	0.219	0.260
CaneSpeaker (w/o IC)	0.212	0.369	0.227	0.242	0.248	0.438	0.236	0.289
CaneSpeaker (w/o LLM-IA)	0.191	0.324	0.188	0.171	0.245	0.401	0.240	0.266
CaneSpeaker (w/o Hindi)	0.201	0.412	0.211	0.214	0.257	0.416	0.235	0.276
CaneSpeaker (w/o Rx2R)	0.214	0.426	**0.266**	0.223	0.234	0.428	0.236	0.281
CaneSpeaker (FF)	0.208	0.404	0.205	0.228	0.258	0.437	0.239	**0.300**
CaneSpeaker	**0.220**	**0.437**	0.248	**0.267**	**0.266**	**0.469**	**0.245**	**0.300**

marked by free-form language and diverse landmarks, and R2R were also included to assess the generalization ability of the baselines. The baselines in the ablation studies include:

- CaneSpeaker (w/o CT): CaneSpeaker without cross-task training (trained only on RxR).
- CaneSpeaker (w/o IC): CaneSpeaker without incorporating the image captioning data.
- CaneSpeaker (w/o LLM-IA): CaneSpeaker without incorporating the augmented instructions generated by LLM-IA.
- CaneSpeaker (w/o Hindi): CaneSpeaker without using Hindi instructions to generate augmented instructions in Rx2R.
- CaneSpeaker (w/o Rx2R): CaneSpeaker incorporating the generated Rx2R instructions, but treating them as R2R instructions.
- CaneSpeaker (FF): CaneSpeaker fine-tuned using full fine-tuning.

Cross-Task Training. The impact of cross-task training on the performance of CaneSpeaker were evaluated. As shown in Table 12, the SPICE score on RxR improves by 0.012 when additional VLN/VL tasks are incorporated (CaneSpeaker (w/o LLM-IA)) compared to training solely on RxR (CaneSpeaker (w/o CT)). This improvement can be attributed to the shared comprehension and reasoning skills required across these tasks. Despite differences in task formulation, the inclusion of

diverse yet related training data enhances the speaker's ability to generate high-quality instructions. This result underscores the benefit of leveraging cross-task training to improve generalization in instruction generation.

Ablation of the Image Caption Task. We conducted ablation experiments to assess the impact of incorporating additional image captioning datasets. As shown in Table 12, while CaneSpeaker (w/o IC) performs reasonably well on n-gram-based metrics such as BLEU-4, it underperforms on SPICE, the primary semantic-based metric. This suggests that although the speaker can learn basic linguistic patterns, it struggles to identify the correct visual landmarks in unseen environments. These results highlight the value of incorporating external image captioning data, which helps the speaker avoid overfitting to seen environments and enhances its visual generalization capabilities.

Impact of LLM-IA. The effect of incorporating the Rx2R instructions generated by LLM-IA on CaneSpeaker were examined. Comparing CaneSpeaker (w/o LLM-IA) and CaneSpeaker in Table 12, an increase of 0.029 and 0.021 on the SPICE scores on RxR and R2R can be observed, respectively. This can be attributed to the fact that Rx2R plays a crucial role in speaker training by not only expanding the training dataset but also enhancing the model's ability to perform effective landmark selection and instruction grounding. Additionally, this reinforces our earlier conclusion that integrating well-designed auxiliary tasks can improve instruction generation performance, further validating the advantage of the proposed instruction augmentation method.

Does incorporating Hindi RxR instructions help? We perform an ablation study to assess whether incorporating multilingual instructions for instruction augmentation in LLM-IA contributes to speaker training. It can be observed in Table 12 that the inclusion of augmented Hindi instructions leads to consistent improvements across multiple evaluation metrics. This is because unlike tasks such as image captioning, navigation instruction generation requires the careful selection of appropriate landmarks and spatial reasoning between navigation steps. The inclusion of instructions in Hindi not only adds linguistic diversity but, more importantly, introduces different landmark selections and spatial descriptions. This diversity encourages the speaker to better learn how to generate contextually appropriate and spatially coherent instructions.

Should Rx2R Be Treated as an Additional Task? Experiments were conducted to further investigate whether Rx2R instructions should be explicitly treated as a distinct auxiliary task or simply merged with existing R2R instructions. From Table 12, it can be observed that while the SPICE score on RxR remains mostly similar, treating Rx2R in the same way as R2R leads to a significant drop of 0.032 on SPICE on R2R. Moreover, this performance degradation of CaneSpeaker (w/o Rx2R) on R2R surpasses even that of a speaker trained without augmented instructions, suggesting a negative effect of the direct merging strategy. This drop can be attributed to the inherent differences between Rx2R and R2R instructions: Rx2R retains varied, instance-oriented landmark selection patterns from RxR, whereas R2R instructions mostly include regional landmarks. Consequently, when Rx2R instructions are incorrectly treated as R2R instructions, the speaker generates instructions that blend regional and instance-oriented landmarks, leading to inconsistencies that negatively impact R2R instruction generation performance. These results highlight the necessity of our strategy that explicitly models Rx2R as a separate auxiliary task rather than merging it into R2R, ensuring optimal speaker performance and instruction generation quality.

LoRA vs. Full Fine-tuning. We also evaluated the impact of using LoRA, in comparison to full fine-tuning. As shown in Table 12, LoRA achieves better performance across key evaluation metrics. This can be attributed to the following fact. Although the base VLM used in our work is relatively small, with 0.45B parameters in the LLM backbone, it remains large relative to the scale of the training data, even with the proposed Rx2R dataset. Moreover, full fine-tuning may lead to the forgetting of the model's previously acquired reasoning capabilities. In such a regime,

Table 13. Impact of different LLMs on the performance of LLM-IA. SR serves as the primary metric. Best in bold.

LLMs	Size	val_seen				val_unseen			
		TL	NE ↓	**SR ↑**	SPL ↑	TL	NE ↓	**SR ↑**	SPL ↑
Phi3.5 [1]	**3.8B**	16.21	5.44	50.12	42.29	16.58	5.24	47.22	37.95
Qwen2.5 [33]	**0.5B**	20.10	7.41	33.91	26.62	20.65	7.04	31.54	23.10
	1.5B	18.41	5.79	47.75	38.00	18.56	5.82	43.23	33.39
	3B	16.04	5.70	47.22	39.62	16.76	5.67	44.17	34.80
	7B	15.96	4.77	53.94	45.79	16.11	4.95	50.02	40.78
LLaMA3 [16]	**1B**	17.05	7.27	33.50	26.75	17.06	6.92	31.48	24.20
	3B	15.65	4.95	52.54	44.69	16.65	5.20	48.17	39.00
	8B	15.52	**4.64**	56.56	49.27	15.41	4.71	53.63	45.38
GPT 4.1 [2]	-	15.35	4.68	**57.33**	**49.38**	15.38	4.69	53.59	45.24
Gemini 2.5 [15]	-	15.55	**4.64**	**57.33**	49.15	15.24	**4.65**	**54.07**	**45.55**

full fine-tuning risks overfitting and inefficiency, while LoRA provides a more parameter-efficient alternative that promotes better generalization.

Impact of Different LLMs for LLM-IA. An ablation study were also conducted to assess the impact of different LLMs on LLM-IA. Specifically, we examined how variations in parameter size and model type influence the quality of the generated instructions. To this end, three VLN agents (Hamt [10], Duet [11], and Duet+ScaleVLN [38]) were employed to evaluate their navigation accuracy when guided by instructions generated using different LLMs.

Our comparison included the following models: LLaMA3 1B/3B/8B [16], Phi3.5 [1], and Qwen2.5 0.5B/1.5B/3B/7B [33]. Additionally, we conducted experiments comparing with stronger proprietary LLMs. Specifically, we evaluated two powerful non-open models: GPT-4.1 [2] and Gemini 2.5 [15]. The results, summarized in Table 13, indicate that among these models, LLaMA3 8B achieves the highest performance, with an average success rate of 56.56% and 53.63% on the val_seen and val_unseen splits of the generated Rx2R instruction set, respectively. A key observation from our results is that the parameter size of the LLM plays a crucial role in instruction quality. Across both LLaMA3 and Qwen2.5, larger models consistently yield better navigation success rates. This result aligns with findings from prior work on scaling laws for Language Models [20], which suggest that larger models exhibit improved language understanding, reasoning, and text generation capabilities. However, the performance gains diminish as the model size increases beyond a certain threshold. As shown in Table 13, models with fewer than 3B parameters perform poorly due to their limited model size, and for models with at least 3B parameters, performance improvements become more gradual. This suggests that even smaller models like LLaMA3 3B and Qwen2.5 3B contains the advanced reasoning abilities required to generate high-quality augmented instructions. Furthermore, while non-open models achieve slightly better performance, the improvements are marginal. This suggests that for LLM-IA, a relatively small open model with 8B parameters is sufficient to generate high-quality instructions. In contrast, using larger proprietary LLMs offers limited benefits while incurring significantly higher API costs and computational overhead.

Beyond model size, the differences between model families were also analyzed. Our results indicate that in models with at least 3B parameters, LLaMA outperform both Qwen2.5 and Phi3.5 at comparable parameter scales, likely due to training methodology and architectural differences. Consequently, LLaMA3 8B were selected as the LLM for generating the Rx2R dataset, as it provides

the highest instruction quality. However, for a better trade-off between instruction quality and computational feasibility, smaller models like LLaMA3 3B and Phi3.5 3.8B can also be considered.

5 Conclusion

In this paper, CaneSpeaker, a general speaker for generating VLN instructions was proposed. Supported by an LLM-based instruction augmentation module and a VLM, CaneSpeaker improves the quality of the generated instructions while reducing input constraints. Besides, two augmented datasets, Rx2R and CANE were also established. Experiments demonstrate that when trained on these datasets, VLN agents exhibit improved way-finding capabilities. Our work not only produces two high-quality datasets that can seamlessly improve VLN agents' way-finding abilities, but also provides valuable insights for future research in enhancing the navigation performance of VLN agents through the synthesis of high-quality instructions.

References

[1] Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 Technical Report: A highly capable language model locally on your phone. arXiv:2404.14219 [cs.CL]

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2024. GPT-4 technical report. arXiv:2303.08774 [cs.CL]

[3] Sanyam Agarwal, Devi Parikh, Dhruv Batra, Peter Anderson, and Stefan Lee. 2019. Visual landmark selection for generating grounded and interpretable navigation instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 7.

[4] Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. 2023. BEVBert: Multimodal map pre-training for language-guided navigation. arXiv:2212.04385 [cs.CV]

[5] Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. 2024. ETPNav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024), early access.

[6] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3674–3683.

[7] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3D: Learning from RGB-D data in indoor environments. In *Proceedings of the International Conference on 3D Vision*. 667–676.

[8] Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee Wong. 2024. MapGPT: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Bangkok, Thailand, 9796–9810.

[9] Jingwen Chen, Jianjie Luo, Yingwei Pan, Yehao Li, Ting Yao, Hongyang Chao, and Tao Mei. 2023. Boosting vision-and-language navigation with direction guiding and backtracing. *ACM Transactions on Multimedia Computing, Communications, and Applications* 19, 1, Article 9 (Jan. 2023), 16 pages.

[10] Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021. History aware multimodal transformer for vision-and-language navigation. In *Advances in Neural Information Processing Systems*. 5834–5847.

[11] 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*. 16537–16547.

[12] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. arXiv:1504.00325 [cs.CV]

[13] Sheng Fan, Rui Liu, Wenguan Wang, and Yi Yang. 2024. Navigation instruction generation with BEV perception and large language models. In *Proceedings of the European Conference on Computer Vision*. 368–387.

[14] 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 *Advances in Neural Information Processing Systems*. 3318–3329.

[15] Google Gemini Team. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf.

[16] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, et al. 2024. The Llama 3 herd of models. arXiv:2407.21783 [cs.AI]

[17] 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*. 1634–1643.

[18] 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*. 13137–13146.

[19] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. arXiv:2106.09685 [cs.CL]

[20] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. arXiv:2001.08361 [cs.LG]

[21] 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 Conference on Empirical Methods in Natural Language Processing*. 4392–4412.

[22] Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. 2025. NavCoT: Boosting LLM-based vision-and-language navigation via learning disentangled reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025), early access.

[23] Kunyang Lin, Peihao Chen, Diwei Huang, Thomas H. Li, Mingkui Tan, and Chuang Gan. 2023. Learning vision-and-language navigation from YouTube videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8317–8326.

[24] Rui Liu, Wenguan Wang, and Yi Yang. 2024. Volumetric environment representation for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16317–16328.

[25] Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2024. Discuss before moving: Visual language navigation via multi-expert discussions. In *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 17380–17387.

[26] Bowen Pan, Rameswar Panda, SouYoung Jin, Rogerio Feris, Aude Oliva, Phillip Isola, and Yoon Kim. 2024. LangNav: Language as a Perceptual Representation for Navigation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 950–974.

[27] 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*. 9982–9991.

[28] Yanyuan Qiao, Qianyi Liu, Jiajun Liu, Jing Liu, and Qi Wu. 2024. LLM as copilot for coarse-grained vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision*. 459–476.

[29] 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*. 15418–15427.

[30] Niyati Rawal, Roberto Bigazzi, Lorenzo Baraldi, and Rita Cucchiara. 2024. AIGeN: An adversarial approach for instruction generation in VLN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2070–2080.

[31] Qiang Sun, Yifeng Zhuang, Zhengqing Chen, Yanwei Fu, and Xiangyang Xue. 2021. Depth-guided AdaIN and shift attention network for vision-and-language navigation. In *Proceedings of the IEEE International Conference on Multimedia and Expo*. 1–6.

[32] Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2610–2621.

[33] Qwen Team. 2024. Qwen2.5: A party of foundation models. https://qwenlm.github.io/blog/qwen2.5/

[34] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and efficient foundation language models. arXiv:2302.13971 [cs.CL]

[35] Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. 2022. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15471–15481.

[36] Su Wang, Ceslee Montgomery, Jordi Orbay, Vighnesh Birodkar, Aleksandra Faust, Izzeddin Gur, Natasha Jaques, Austin Waters, Jason Baldridge, and Peter Anderson. 2022. Less is more: Generating grounded navigation instructions from landmarks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15428–15438.

[37] Xiaohan Wang, Wenguan Wang, Jiayi Shao, and Yi Yang. 2023. Lana: A language-capable navigator for instruction following and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19048–19058.

[38] Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. 2023. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12009–12020.

[39] Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge, and Eugene Ie. 2021. On the evaluation of vision-and-language navigation instructions. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics*. 1302–1316.

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

[41] Baichuan Zhou, Ying Hu, Xi Weng, Junlong Jia, Jie Luo, Xien Liu, Ji Wu, and Lei Huang. 2024. TinyLLaVA: A framework of small-scale large multimodal models. arXiv:2402.14289 [cs.LG]

[42] Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. 2024. NavGPT-2: Unleashing navigational reasoning capability for large vision-language models. In *Proceedings of the European Conference on Computer Vision*. Springer, 260–278.

[43] Gengze Zhou, Yicong Hong, and Qi Wu. 2024. NavGPT: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the Conference on Artificial Intelligence*, Vol. 38. 7641–7649.

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