Visual Intelligence

RESEARCH Open Access

NavGemini: a multi-modal LLM agent for vision-and-language navigation

Ganlong Zhao[1], Guanbin Li[2*] and Yizhou Yu[1*]

Abstract

Large language models (LLMs) such as the GPT series exhibit impressive reasoning and in-context learning capabilities due to the substantial amount of data and computational resources involved in LLM training. Some previous studies have applied the LLMs to vision-and-language navigation (VLN) in order to create navigation agents that are entirely LLM-based, operating within a zero-shot setting, aiming to reveal and utilize LLMs' reasoning and planning capability for VLN tasks. However, these methods employ text-based LLMs for navigation agents, generating a text description of environmental observations during navigation. Other smaller LLMs are used for image-to-text translation, resulting in a gap between image-to-text translation and environmental navigation. Moreover, the high cost of advanced LLMs such as GPT-4 also hinders the application of LLM-based navigation agents. This is particularly the case given the increasing length of the context, which includes navigation history and the numerous visual images generated during navigation. In this paper, we propose NavGemini, a navigation system based entirely on the newly developed multi-modal LLM, Gemini-Pro-Vision. Our aim is to study and utilize the visual-spatial and multi-modal capabilities of LLMs in VLN tasks, while mitigating the challenges posed by token limits when LLMs process large amounts of image-based data and historical information, and when the available multi-modal LLMs perform relatively poorly. Our proposed NavGemini, with its elaborate prompts, successfully outperforms previous methods by 5.7% in terms of success rate, even when using inferior LLMs. This demonstrates the strong ability and potential of multi-modal LLM-based agents in VLN tasks.

Keywords: Vision-and-language navigation (VLN), Large language model (LLM), Multi-modality, Embodied AI

1 Introduction

Large language models (LLM) [1–4] have demonstrated impressive capabilities across various domains. Due to their strong reasoning capability when scaling up the model, LLMs are deemed to be one of the promising pathways to realize a universal embodied agent [5–8], in which the LLMs can provide cross-domain knowledge and analysis in decision-making or motion control.

Recently, LLMs have been applied to the area of vision-and-language navigation (VLN) [9, 10], where the embodied agents equipped with cameras need to perceive and navigate the indoor environment following the given human instructions. Previous VLN methods train agents with large-scale training data collected from similar indoor environments with intensive human labor, which is expensive and domain-specific. LLM-based VLN agents [11, 12] address the issue by incorporating LLMs into their construction and formulating the VLN as a zero- or few-shot learning process. Such LLM-based agents reduce training costs and reliance on annotated datasets, while improving their generalizability (e.g., from simulated to real environments, or from seen to unseen environments).

*Correspondence: liguanbin@mail.sysu.edu.cn; yizhouy@acm.org
[1]Department of Computer Science, The University of Hong Kong, Hong Kong, China
[2]School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, 510006, China

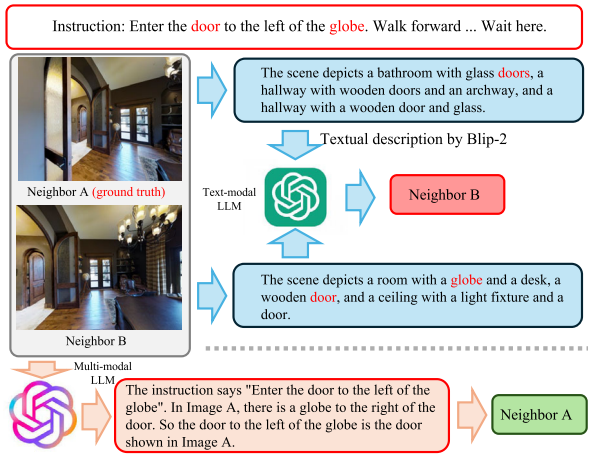

Figure 1 The comparison between text-modal LLM-based agent and multi-modal large language model (LLM)-based agent in VLN tasks. The image-to-text translation ignores the context of the instruction and tasks and thus generates inaccurate and misleading descriptions for the text-modal LLM. The multi-modal LLM successfully understands the difference between the two neighbors and selects the correct one based on the context of the instruction

The recently proposed LLM-based VLN agents [11, 12] use a zero-shot approach for VLN tasks. NavGPT [11] employs GPT-4 [2] to make navigation decisions and perform reasoning tasks in pure text form. More specifically, NavGPT first generates the textual description for the observations that the agents' cameras capture during navigation by a multi-modal image-to-text model Blip-2 [13]. Then, the prompt manager of NavGPT constructs the prompt by incorporating the given instruction, the information about the environment at the agent's current position, the reachable neighbors that can be selected for the next movement, and the navigation history. The constructed prompt is sent to GPT-4 for text auto-completion, and GPT-4 [2] will then analyze the situation and determine the next move. The prompt and the response are then concatenated as the navigation history for the next step.

Previous LLM-based VLN agents adopt single-modal LLMs for navigation decision-making in the pure text form and separate the image-to-text process from the decision-making. This could lead to the image-text gap in the performance of VLN tasks. As shown in Fig. 1, the image-to-text translation using a separated multi-modal model cannot generate more instruction-related textual descriptions of the observations at the agent's current position because reasoning and decision-making occur after the image-to-text translation. With the development of LLMs, there has been an increasing emergence of multi-modal LLMs, such as GPT-4-Vision [4] and Gemini [14]. Such multi-modal LLMs accept multi-modal queries that include both text and visual input. This makes it possible to construct the LLM-based VLN agents in an end-to-end, multi-modal manner.

However, there are some problems for the multi-modal LLM-based VLN agents. First, including images in the multi-modal query significantly increases the number of tokens in the prompt sent to the multi-modal LLMs, which reduces their response speed. Besides, due to the token limit of the prompt, it is impossible to include the visual-modal history, e.g., observations along the navigation path in the query. The multi-modal LLM-based VLN agent needs to develop a new mechanism to manage and embed the navigation history in the query with higher efficiency. Second, existing multi-modal LLMs often fail to understand the spatial relationship between the objects from different views and the VLN agent. For example, it is difficult for Gemini-Pro to understand the difference between going upstairs and downstairs. Therefore, an elaborate prompt management system needs to be constructed to manage different situations in indoor environment navigation. Third, the limited token quota for multi-modal LLM queries hinders long-range planning and backtracking during navigation, as the queries do not contain enough history information along the navigation path. Thus, the multi-modal LLM-based agents need to include a planning and backtracking mechanism. Finally, the computational burden and cost are also major barriers to the widespread adoption of LLM-based VLN agents.

In this paper, we propose a novel multi-modal LLM-based agent for VLN. We first disassemble the VLN task into different subtasks and design different prompts for those subtasks, and then we adopt a prompt manager to manage the prompts and select the proper prompt for the LLM query based on the situation of the VLN agents during navigation. The subtask prompts and the prompt manager aim to perform the VLN tasks in different scenarios with multi-modal LLM queries, given the limited token quota, and thus address the gap between the image-to-text translation and the navigation decision-making. More specifically, we design two important queries for VLN with LLMs: the main query and the instruction reduction query. The main query aims to select one neighbor as the next position the agent should move to at the first step, or stop the agent at the current position and terminate the navigation based on the given instruction. The main query focuses solely on the first step of the navigation, regardless of the length of the instruction, and thus does not require information about navigation history. The instruction reduction query generates the new instruction based on the instruction for the entire navigation task and the progress made in the last step of the agent's navigation. The new instruction specifies the rest of the navigation task that the agent should complete. Then, the newly generated instruction can be applied to the main query for the next step,

as the first step of the new instruction has not been completed. Moreover, we present a termination check and selection mechanism that identifies whether the navigation agent has reached its destination. This mechanism also selects the best-matched destination in history as the final stopping point. We design the backtracking and probe strategy to improve the success rate of the navigation.

In summary, the contributions of our paper are as follows:

1) We are the first to incorporate the multi-modal LLMs into VLN agents to mitigate the gap between image-to-text translation and navigation decision-making.
2) We propose a novel multi-modal LLM-based agent that disassembles the VLN task into a series of subtasks and performs the subtasks with elaborate prompts and a prompt manager for process control.
3) Our method outperforms previous LLM-based VLN agents in both R2R [9] and REVERIE [15] datasets. Besides, our method is based on cost-free LLM APIs, which is important for the wide application of LLM-based VLN agents.

2 Related work

VLN [9, 10, 15] requires an agent to perform a language-driven navigation task in an indoor environment, which is both important and fundamental for widely applicable embodied navigation agents. Tailored to various practical scenarios, many different VLN benchmarks have been studied, including step-by-step instructions such as R2R [9] or RxR [16], navigation with dialogs such as CVDN [17], and navigation for remote object grounding such as REVERIE [15] and SOON [18]. Besides, there are also benchmarks in both discrete [9] and continuous environments [10] for evaluating agents' performance in coarse-grained and fine-grained control.

Existing VLN agents rely on elaborately designed modules and training strategies, or other data-related techniques including data augmentation [19, 20], memory mechanism [21–23], and pre-training [24, 25] to alleviate data scarcity. However, adapting the VLN agents trained in the seen environments to unseen environments is difficult, as shown in the large performance gap in previous VLN studies [21, 26]. In order to address more complex unseen scenarios and a more diverse range of instructions, some methods [11, 12, 27, 28] adopt the LLMs for their reasoning and knowledge storage and perform the VLN tasks in a zero-shot manner. However, previous LLM-based VLN agents perform the navigation tasks in a plain text format. For example, NavGPT [11] first uses Blip [13] to convert the visual-modal observations along the navigation path to the text-modal description, and then adopts GPT-4 [2] for navigation inference, resulting in the translation gap between observation and textual inference. Besides, the LLM queries lead to excessive model invocation costs, which hinders the LLM-based VLN agents from wide application in real scenarios. In this paper, we aim to mitigate the two problems by designing an elaborate prompt system and adopting inexpensive and inferior LLMs for VLN agents.

Based on Transformers [29], LLMs [1, 30–39] have achieved massive success in the field of language processing by scaling up the model size and the pre-training corpus. Some studies first pre-train the LLMs to obtain most of the capability, and then adopt instruction tuning [40, 41] to improve model performance and generalization to unseen tasks. Some research focuses on interaction tools and plugins [42, 43], Internet access API calls [44], or local databases to expand the knowledge of LLMs [45]. Some techniques include a hierarchical system to align the reasoning and corresponding actions [46, 47], and chain of thought (COT) [48].

Vision language models (VLMs) [49–52] are LLMs augmented with visual inputs and can process visual language tasks in a multi-modal manner. VLMs use cross-attention [49, 50] to fuse the visual information into intermediate embeddings or they use an auto-regressive approach to tokenize the visual input alongside the text tokens [13, 51, 53–58]. In this paper, we adopt Gemini-Pro-Vision [14] as the VLM for multi-modal queries in our method.

Some studies [5–7, 59–62] aim to leverage LLMs for embodied applications. These LLMs are utilized for planning, reasoning [7, 63–65], and physical environment perception with textual description [66], perception APIs [67], or multi-modal models [55]. Other research focuses on robot manipulation and low-level control through pre-defined primitives and sequential policy composition [5], as well as reward design and exploration in reinforcement learning [68]. For navigation tasks, LLMs are utilized for landmark or subgoal identification [69], code generation [70], or commonsense knowledge extraction [71].

3 Method

The framework of our method is depicted in Fig. 2. The instruction for each episode is sent to the prompt manager to prepare the meta information for further queries. Then, the agent prepares the observations of each neighboring viewpoint as the candidates for the next movement. After the neighboring viewpoint filtering, the prepared observations and instructions, along with their metadata, are sent to the prompt manager for the main query. There, the multi-modal LLM receives the input and selects one observation as the predicted viewpoint for the agent's next movement. Given the predicted viewpoint, the agent collects the wide-angle observation before and after moving toward the selected viewpoint. Then, the agent sends the two observations with the instruction to LLMs for instruction reduction. The new instruction generated by the in-

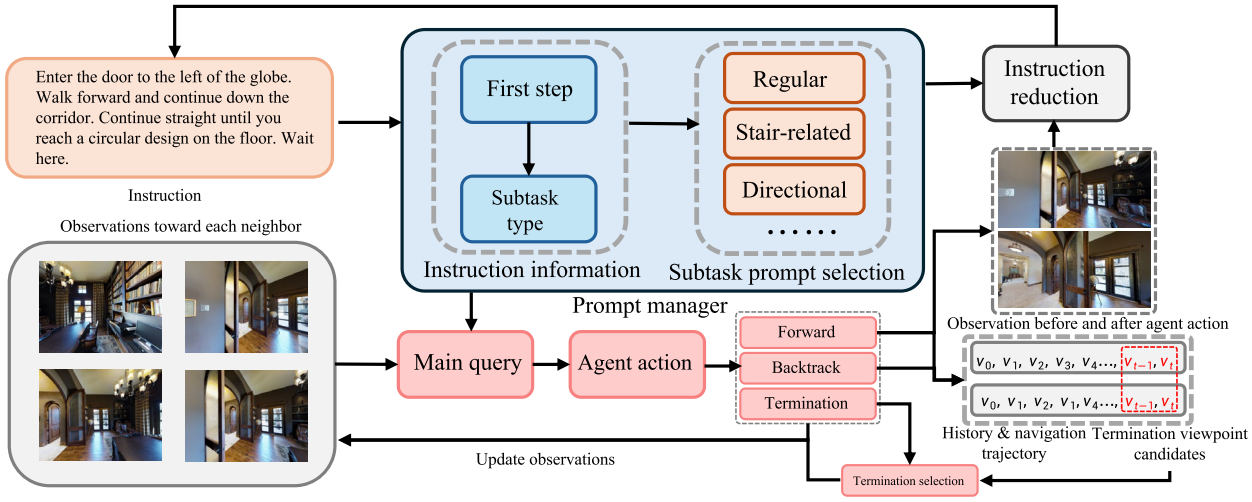

Figure 2 The main framework of NavGemini. The prompt manager analyzes the instruction and selects the corresponding subtask prompt for the main query. The main query decides the agent action by selecting one neighbor to forward based on the observations, termination or backtracking. The instruction reduction approach reduces the instruction to eliminate the completed subtasks. The trajectories are maintained for termination selection and backtracking. v_i is the i-th viewpoint in the navigation trajectory

struction reduction approach is used as the new instruction for the next movement step. The overall algorithm is shown in Algorithm 1.

3.1 Problem formulation

In this paper, we study the LLM agents under the VLN setting in discrete environments, where the agent navigates the environments through the pre-defined discrete viewpoints. Given an instruction $\mathcal{W} = (w_1, w_2, \ldots, w_N)$ consisting of N words in natural language, the agent starts from the given starting viewpoint and aims to follow the instruction $\mathcal{W}$ to reach the destination viewpoint. Suppose the agent is located at viewpoint v_t at step t, the agent can obtain the panoramic observation $\mathcal{O}$ at v_t and the directions and distances of the M_{v_t} neighbouring viewpoints $\{v'_1, v'_2, \ldots, v'_{M_{v_t}}\}$. Each neighboring viewpoint v'_n has a different direction (heading) h_n and distance d_n concerning the agent's current position and orientation. The agent must select a neighboring viewpoint to move toward as the next step or stop at the current viewpoint when the agent believes the destination is reached.

Similar to previous studies [11], the neighbor selection is based on neighbor observations. More specifically, each neighbor v'_n of the current viewpoint v_t is attached to a neighbor observation $\mathcal{O}_{v_t,v'_n}$ which is what the agent camera observes when looking to v'_n from v_t. The LLM agent selects a neighbor observation $\mathcal{O}_{v_t,v'_n}$ as the target, which indicates that the agent should move toward v'_n as its next action. If the agent decides to stop at the current position after observing all the neighbors and the surrounding environment, it selects no neighbor and terminates the navigation.

3.2 Main query: neighbour selection with multi-modal prompt

In this section, we aim to mitigate the central problem of VLN with multi-modal LLMs, i.e., neighbor selection with multi-modal prompts. The difficulty of this problem lies in several aspects. First, the images in the multi-modal prompts that are sent to the multi-modal LLMs take a lot of token quota in LLM queries. Prompts with a larger number of tokens not only increase the computational burden but also decrease the performance of LLMs. Therefore, we have to design multi-modal prompts for neighbor selection that reduce the number of tokens for greater efficiency. Second, the token limit for LLM queries restricts the usage of multi-modal history. When an LLM agent needs to decide on its next move during a navigation task, both the textual and visual history information are important for the decision. However, the history of the agent could be so extensive that it would be impossible to fit all the visual observations along the navigation path into the multi-modal prompts. One natural solution to this problem is to truncate the history, which involves setting a limit on the length of the history observations and removing the oldest observations along the path. However, this will cause a misalignment between the instruction and the history, resulting in unsatisfactory performance. Third, prompt management is necessary for multi-modal LLMs. The instructions in VLN often include different types of sub-tasks in the same instructions, and different sub-tasks require different multi-modal prompts to provide more fine-grained queries and higher performance in navigation. For example, the instruction "Turn

Algorithm 1: NavGemini: A multi-modal LLM agent

Input: Instruction I_0; Starting viewpoint v_0
Params: t_c: Starting step for termination check; M_{step}: Max step number. M_{depth}: Max depth.

Initialize history trajectory $\mathcal{T}_{\text{h}}$ and navigation trajectory $\mathcal{T}_{\text{n}}$, termination candidate set $\mathbf{C}$.
for $t = 0, \ldots, M_{\text{step}} - 1$ **do**
  Collect neighbor observations $\mathcal{O}' = \{\mathcal{O}_{v_t,v'_1}, \ldots, \mathcal{O}_{v_t,v'_n}\}$
  if instruction I_t is "stop" or $|\mathcal{T}_n| > M_{\text{depth}}$ **then**
    if $|\mathbf{C}| > 0$ **then**
      Perform termination selection with $\mathbf{C}$
      Update trajectories $\mathcal{T}_{\text{h}}$ and $\mathcal{T}_{\text{n}}$
      break
    else
      Backtrack and update trajectories $\mathcal{T}_{\text{h}}$ and $\mathcal{T}_{\text{n}}$
      Restore instruction I_{t-1} as I_{t+1}
  else
    Select neighbour v_{t+1} by main query with I_t and $\mathcal{O}'$
    Update trajectories $\mathcal{T}_{\text{h}}$ and $\mathcal{T}_{\text{n}}$
    Generate the instruction I_{t+1} by instruction reduction
    Perform termination check for v_{t+1} and update $\mathbf{C}$ if $t + 1 \geq t_c$

Output: History trajectory $\mathcal{T}_{\text{h}}$

left, then go to the living room" involves two different subtasks, the directional subtask and the target-relative subtask. To achieve higher performance, we need to design two different prompts for each task, and using a simple overall prompt for both subtasks will reduce the accuracy.

Based on the above analysis, we propose solving the neighbor selection problem in the first step only. In other words, we disregard the history of neighbor selection and instruct the multi-modal LLMs to focus solely on the initial selection during navigation, regardless of the length of the navigation instruction. Hereinafter, this process is referred to as the main query for LLM-based VLN. The main query is performed by the prompt manager in our method. The prompt manager consists of the first step prompt, the subtask type prompt, and a series of neighbor selection prompts designed for each type of subtask. Given an instruction w and the observations $\{\mathcal{O}_{v_t,v'_1}, \ldots, \mathcal{O}_{v_t,v'_n}\}$ of the n neighboring viewpoints from the agent's current position v_t, the prompt manager first uses the first step prompt to extract the first movement of the given instruction, and then the subtask type prompt decides which subtask the first step of the instruction belongs to. Given the subtask type, the prompt manager selects the corresponding neighbor selection prompt to perform the main query.

We present three different subtasks for LLM-based VLN instructions: regular subtask, directional subtask, and stair-related subtask. Directional subtask refers to the subtask that involves direction instructions, e.g., turn left/right/around, and move forward/back. Stair-related subtask includes two different instructions: go upstairs and go downstairs. All the other instructions are assigned to the regular subtask. After selecting the corresponding neighbor selection prompt, we prepare the observations $\{\mathcal{O}_{v_t,v'_1}, \ldots, \mathcal{O}_{v_t,v'_n}\}$ of the n neighbors from the agent's current position v_t, and query the multi-modal LLM to determine which observation most likely indicates the direction in which the agent should move at the first step.

3.3 Instruction reduction

The multi-modal LLM-based agent can perform the first step of the main query when given a navigation instruction. However, the navigation path of the instruction always contains multiple steps. In this section, we generate a new instruction based on the last step taken by the navigation agent. This new instruction has the first step as the subtask that the agent should perform in the current step. For example, if the agent walked down one flight of stairs after being given the instruction "Walk down one flight of stairs and stop on the landing," the instruction reduction process should generate the new instruction "Stop on the landing". With the newly generated, reduced instructions, the LLM-based VLN agent can reuse the main query for the next step without including the history observations or the path in the query.

The key to the instruction reduction process is the identification of the subtasks that have been accomplished in the last step. Therefore, in the instruction reduction query, we present observations of the agent's movement before and after the last step $\{\mathcal{O}_{\text{before}}, \mathcal{O}_{\text{after}}\}$. Besides, we also provide the directional and stair-related information for the final step so that the LLM can identify the completeness of the directional and stair-related subtask.

More specifically, during the instruction reduction phase, we first instruct the LLM to extract the landmarks and directions mentioned in the instruction in order because the landmarks and directions can be strong indicators of whether the current navigation task is complete. We also extract the first step subtask of the current instruction for LLM's reference in instruction reduction. The two observations $\{\mathcal{O}_{\text{before}}, \mathcal{O}_{\text{after}}\}$, the landmark and direction list, and the first step of the instruction are provided for instruction reduction query. We ask the multi-modal LLM to summarize the key objects in the two observations and to compare the differences between the two images. Then, the LLM needs to generate a new instruction that considers the difference in observations, the first step of the

instruction, and the direction and stair information of the last step.

The process of reducing instructions ends when all the subtasks in the instruction are completed in the final step. In this case, the LLM generates "stop" as the new instruction, and the agent stops at the current position. Therefore, the instruction reduction naturally provides a termination check for VLN.

3.4 Termination check and selection

The LLM-based VLN agent can perform the navigation task with the main query and instruction reduction. However, the instruction reduction often fails to identify the completeness of the final step of navigation tasks, because termination identification requires an in-place understanding of the surrounding environment, whereas the instruction reduction only focuses on observations in front of the LLM-based agent. Therefore, we use a termination check and selection mechanism to identify termination viewpoints, which can provide a more concise stopping viewpoint from the visited viewpoints in the history path.

Suppose the multi-modal LLM-based agent arrives at the viewpoint v_t at step t, and the history trajectory till step t is $\mathcal{T} = \{v_1, v_2, \ldots, v_t\}$. Given a hyper-parameter t_c, we collect the panoramic observations $\mathcal{O}_{\text{pano}}$ of v_t if $t \geq t_c$. Then, we use a termination check prompt to query the multi-modal LLM if the viewpoint v_t is the destination of the corresponding instruction. The multi-modal LLM can answer the query with one of the three options: Yes, No, or Unknown. "Unknown" indicates that the LLM is not sure whether viewpoint v_t is the destination due to insufficient information. Please note that the termination check prompt does not terminate the agent regardless of the query answer. The navigation agent terminates when the instruction reduction process generates "stop" as the new instruction or the navigation path has reached the navigation depth limit, and all the viewpoints since step t_c will go through the termination check.

After the navigation agent stops, the termination check generates a set of termination viewpoint candidates $\mathbf{C}$ that includes the viewpoints with "Yes" answers, and we need to select one viewpoint from $\mathbf{C}$ as the stopping point. To this end, we use a termination selection prompt to select the destination from $\mathbf{C}$. If $\mathbf{C}$ contains only one candidate v, then v will be selected as the destination. If $\mathbf{C} = \{v_1^T, v_2^T, \ldots, v_n^T\}$ where $n \geq 2$ and T is the timestamp when the navigation agent stops, we first collect the observations $\{\mathcal{O}_{\text{pre}(v_i^T), v_i^T} | v_i^T \in \mathbf{C}\}$ for each candidate $v_i^T \in \mathbf{C}$, where $\text{pre}(v_i^T)$ is the predecessor of candidate v_i^T in the history trajectory $\mathcal{T}$, and $\mathcal{O}_{\text{pre}(v_i^T), v_i^T}$ is the observation looking toward v_i^T at viewpoint $\text{pre}(v_i^T)$. Then, the observations are sent to the multi-modal LLM for termination selection query with the prompt, and the LLM needs to select one candidate as the final termination viewpoint for the agent. Note that all the viewpoints in $\mathbf{C}$ appear in the navigation trajectory $\mathcal{T}$, and the agent can always travel back to the selected candidate.

An important difference between termination check and termination selection is the observations sent to the LLMs. In termination check, we collect the panoramic observations of viewpoints to provide sufficient information about the surrounding environments, while in termination selection, we only collect the agent's observations of the candidate from the navigation history, which reduces the number of prompt tokens and provides visual information about the blind angle at v_i^T, such as above or below the agent.

3.5 Backtracking

The termination check process generates a set of termination viewpoint candidates $\mathbf{C}$. If the agent has reached the maximum number of navigation steps or has reduced the navigation instruction to the "stop" instruction, the agent needs to select one candidate from $\mathbf{C}$ as the final destination and stop at it. However, if $\mathbf{C}$ is empty, and no available viewpoint can be selected as the termination viewpoint, the agent may have navigated in the wrong direction and become farther from the ground truth destination. In this case, the agent needs to backtrack to the previous viewpoint and mark the current viewpoint as a dead end.

More specifically, our method sets two different hyperparameters for backtracking control: the max step number for navigation M_{step} and the max depth for navigation M_{depth}. The agent also maintains two different trajectories for backtracking: the history trajectory and the navigation trajectory. The history trajectory records all viewpoints that the agent has visited in the navigation history, while the navigation trajectory records the navigation path after eliminating the backtracking movement. For example, if the agent navigates from viewpoint a to b and then decides to backtrack to a, the history trajectory will be (a, b, a) while the navigation trajectory will be (a). When deciding the next movement, the agent checks whether there are no candidates in $\mathbf{C}$ and 1) the length of the navigation trajectory exceeds M_{depth}, or 2) all the next reachable neighbors of the current viewpoint are marked as dead ends. If yes, the agent backtracks to the last viewpoint recorded in the navigation trajectory and modifies the two trajectories accordingly. The agent also records the reduced instruction at each step and restores the corresponding instruction when backtracking. The agent terminates when 1) the history trajectory length exceeds M_{step}, or 2) the reduced instruction is "stop" and there are candidates in $\mathbf{C}$, or 3) the navigation trajectory length exceeds M_{depth} and there are candidates in $\mathbf{C}$.

3.6 Probe

The neighbor selection process selects the next viewpoint based on the observations of the agent's current position. However, such observations fail to provide LLMs with distance information about different neighbors. Two different neighbors with different distances can have similar directions and thus lead to similar observations. As the experiments showed, the multi-modal LLM cannot effectively distinguish the small angle changes in the neighbor selection, while the neighbors that are close in direction could lead to completely different areas due to their large distance gap. Therefore, we choose to use a probe prompt for a more fine-grained selection when the selected target viewpoint has a close-in-direction neighbor.

Given an angle threshold a_c as a hyper-parameter, if the agent arrives at viewpoint v_t, and the angle between the selected viewpoint v'_{t+1} in the main query and its nearest neighbor v''_{t+1} is smaller than a_c, the agent first navigates to v'_{t+1} and v''_{t+1} to collect the observations $\mathcal{O}_{v'_{t+1}}$ and $\mathcal{O}_{v''_{t+1}}$ in front of the agent. Then, the probe prompt is sent to the multi-modal LLM with the two observations. The LLM needs to return one observation as the next step v_{t+1}. For a fair comparison, we append the entire path of the probe process, i.e., $\{v_t, v'_{t+1}, v_t, v''_{t+1}, v_t, v_{t+1}\}$ to the history trajectory of the agent.

Our current design employs handcrafted prompt templates for different subtasks, which is a deliberate choice made to ensure stable performance despite the limitations of current multi-modal LLMs. Nevertheless, automated prompt optimization techniques [72–74] can potentially reduce the reliance on manual design, improve generalization to new tasks, and further enhance scalability. We consider this a promising direction for future work.

4 Experiment

4.1 Experiment settings

In this section, we select two widely-applied benchmarks for VLN agent evaluation, R2R [9] and REVERIE [15]. Both benchmarks are collected in discrete environments, i.e., the environments are discretized into viewpoints, and the navigation agent travels through the connections between viewpoints. R2R and REVERIE have different instruction styles. R2R provides step-by-step instructions to the navigation agents, which describe the detailed ground truth path from the starting point to the destination. REVERIE only offers high-level instructions that describe the destination and target object of the navigation task. Both R2R and REVERIE are based on the Matterport3D simulator [75]. The R2R dataset contains 7189 trajectories and each trajectory is described by three fine-grained instructions. There are four different dataset splits for R2R: training, validation seen, validation unseen, and test. The four dataset splits are collected from 61, 56, 11, and 18 indoor scenes, respectively. The "seen split" validation set comprises scenes that are present in the training set, whereas the "unseen split" validation set comprises scenes that are not present in the training set. Following previous zero-shot VLN studies, we apply the 783 trajectories in the 11 val unseen environments in all of our experiments. REVERIE shares the same dataset split with R2R, and there are 10,466 instructions from 60 scenes, 4944 instructions from 56 scenes, and 6292 instructions from 16 scenes from the training, validation and test set, respectively. Similar to R2R, we test our method on the validation unseen set, which contains 3521 instructions from 10 scenes for REVERIE. We neglect the object grounding task and focus on the navigation for REVERIE to unify the experiment settings.

We adopt the evaluation metrics for VLN, including navigation error (NE, the distance between the agent's final location and the target location), success rate (SR), oracle success rate (OSR, SR given Oracle stop policy), and SR penalized by path length (SPL).

Different from previous LLM-based VLN agents, which adopt GPT-4 for LLM queries, our method uses Gemini-Pro-Vision and Gemini-Pro for multi-modal queries and text-modal queries, respectively. Gemini-Pro-Vision and Gemini-Pro are currently free to use if the request per minute (RPM) stays within a certain limit. Therefore, compared to previous LLM-based VLN agents, we reduced the query cost by 100%. In our experiments, we set the t_c as 6, M_{step} as 20 and M_{depth} as 10. For LLM queries that involve images, we use Gemini-Pro-Vision, which is cost-free when the RPM is below 60. For purely textual LLM queries, we use Gemini-Pro, which is also free for use.

4.2 Experimental results on R2R dataset

The experimental results of our method on the R2R dataset are presented in Table 1. Following previous studies, we categorize previous VLN agents into three classes: Train Only, Pretrain + Fine-tune, and No Training. Train Only includes the VLN agents that are trained with the training split of the R2R dataset. Pretrain + Fine-tune methods first pre-train the agent with proxy tasks, and then fine-tune the agent with the VLN data. No Training indicates the zero-shot settings for the VLN task, which include DuET with initialized LXMERT, NavGPT, as well as two versions of our method: one with the probe and termination check strategies, and one without.

As shown in Table 1, both variants of our method significantly outperform the competitors in the No Training setting in SR. More specifically, our method using the probe and termination check strategies improves the SR by 5.7% while our method without using the strategies improves the SR by 3.3%. Our method using the probe and termination check strategies also achieves the best performance in NE and OSR, but significantly increases the total length of the navigation path. The reason is that both the probe and termination check strategies require the VLN

Table 1 Comparison with previous methods on R2R validation unseen split. TL: total length; NE: navigation error; OSR: oracle success rate; SR: success rate; SPL: success rate penalized by path length. The best score is highlighted in bold

Training schema	Method	TL	NE↓	OSR↑	SR↑	SPL↑
Train Only	Seq2Seq [9]	8.39	7.81	28	21	-
	Speaker Follower [76]	-	6.62	45	35	-
	EnvDrop [77]	10.70	5.22	-	52	48
Pretrain + Fine-tune	PREVALENT [25]	10.19	4.71	-	58	53
	VLN⟳BERT [78]	12.01	3.93	69	63	57
	HAMT [21]	11.46	2.29	73	66	61
	DuET [26]	13.94	3.31	81	72	60
No Training	DuET (Init. LXMERT [79])	22.03	9.74	7	1	0
	NavGPT	11.45	6.46	42	34	29
	Ours (w Probe & Termination)	40.33	**5.79**	**55.7**	**39.7**	13.3
	Ours (w/o Probe & Termination)	11.66	6.00	45.8	37.3	**32.1**

Table 2 Comparison with previous methods on REVERIE validation unseen split

Training schema	Method	OSR↑	SR↑	SPL↑
Train Only	Seq2Seq [9]	8.07	4.20	2.84
	RCM [76]	14.2	9.29	6.97
	SMNA [77]	11.3	8.15	6.44
	FAST-MATTN [77]	28.2	14.4	7.19
Pretrain + Fine-tune	HAMT [21]	35.4	31.6	29.6
	DuET [26]	50.0	45.8	35.3
No Training	NavGPT	28.3	19.2	14.6
	Ours (w Probe & Termination)	**28.6**	**22.0**	8.7
	Ours (w/o Probe & Termination)	25.6	20.6	**16.9**

agent to repeatedly move back and forth between several viewpoints, thus greatly increasing the TL and reducing the SPL. For example, a probe operation usually increases the navigation path by between 2 and 4 viewpoints, i.e., a → b → a → c → a → b/c, where a is the current position and b and c are the two candidates for probe operation. Our method removes the two components and successfully achieves the best performance in SPL, which is 3.1% higher than NavGPT, and the TL of this method is only slightly higher than NavGPT. As shown in Table 1, the experimental results clearly demonstrate the superiority of our method compared to other zero-shot VLN agents.

4.3 Experimental results on the REVERIE dataset

The experiment results of our method on the REVERIE dataset are presented in Table 2. Different from the R2R, REVERIE only provides high-level instructions that briefly describe the destination and target objects for the VLN agents. We compare our method with other competitors in three metrics, OSR, SR and SPL. Similar to Table 1, there are three different groups: Train Only, Pretrain + Fine-tune, and No Training, and we provide the performance of two different variants of our method. As shown in Table 2, our method using the probe and termination check strategies outperforms NavGPT in both OSR and SR by 0.3% and 2.8%, respectively, while our method without using the probe and termination check strategies outperforms NavGPT in both SR and SPL by 1.4% and 2.3%, respectively.

5 Analysis

5.1 The performance contribution for each component

To demonstrate the effectiveness of each component in our method, we ablate our method and provide the experimental result on the R2R dataset validation unseen split in Table 3. Main query and instruction reduction are necessary for the navigation tasks, and we ablate the three other components, backtrack, termination check and selection, and probe. We report three metrics: total length (TL), success rate (SR), and SPL. As shown in Table 3, backtrack, termination, and probe contribute to the performance improvement on SR by 0.5%, 0.7%, and 1.7%, respectively. Backtracking contributes little because, in most navigation, the agent can find at least one candidate termination viewpoint and thus does not satisfy the backtracking condition. Though the success rate is improved, both the probe and termination will significantly increase the total length of the navigation path and thus reduce the SPL.

Table 3 Performance contribution of the three components in our method

Components			Performance		
Backtrack	Termination	Probe	TL	SR↑	SPL↑
			11.30	36.8	31.7
✓			11.66	37.3	32.1
✓	✓		20.54	38.0	20.1
✓	✓	✓	40.33	39.7	13.3

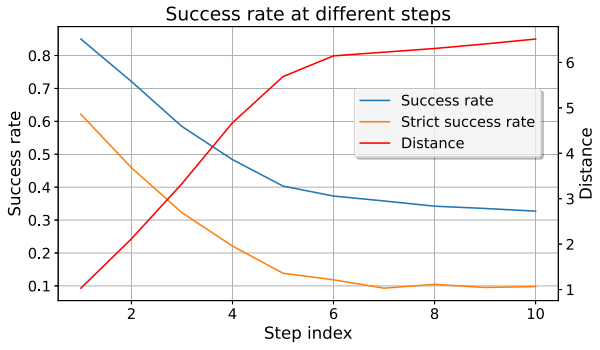

Figure 3 The step-wise success rate and distance to the corresponding ground truth viewpoints during navigation. As navigation continues, both the success rate and the strict success rate decrease, and the distance increases

5.2 Performance at each step

Figure 3 summarizes the step-wise statistics of our method during navigation on the R2R dataset. We provide three metrics: SR, strict SR, and distance at each step. Each metric is calculated concerning the corresponding viewpoint in the ground truth path. For example, the SR at the first step is the ratio of navigation paths in which the first step is within 3 meters of the first viewpoint in the ground truth path. Different from the SR, the strict SR is the ratio of navigation paths that strictly match the ground truth paths. The distance metric is the distance between the agent's current position and the corresponding viewpoint in the ground truth path. As shown in Fig. 3, as the length of navigation increases, both the SR and the strict SR decrease, and the distance continues to increase.

5.3 Analysis for termination check and selection

The termination check process generates answers given the panoramic observations for each viewpoint to check whether it is the final destination of a navigation instruction. The answer is limited to three categories: Yes, No, and Unknown. Table 4 lists the accuracy for each category of answer. Since the termination check process is followed by the termination selection for further classification, the most important property for the termination check should be a low false negative rate. False negative predictions will cause the agent to deviate from the ground truth destination. As shown in Table 4, only 13.8% of the viewpoints

Table 4 Accuracy of termination check for each type of query answer

Answer	Yes	No	Unknown
True	0.401	0.138	0.244
False	0.599	0.862	0.756

with the answer "No" are within 3 meters of the destination. Please note that the ratio could be even smaller if we only considered the viewpoints that strictly matched the ground truth destination to be true. For the Unknown prediction, the accuracy lies between Yes and No, which demonstrates that the LLM can effectively identify the difference between the three responses and answer the query accordingly.

5.4 The performance of main query

We verify the effectiveness of the main query in our method by analyzing the SR at the first step, because it does not involve instruction reduction. As shown in Fig. 3, at the first step, the strict SR is 62.1% and the SR is 85.0%. However, the SR does not reflect the accuracy at the first step because normally the agent is not far from the ground truth viewpoint at the first step. Therefore, we further compiled statistics on another indicator, the angular deviation SR, which checks whether the angular deviation between the selected viewpoint and the ground truth viewpoint is below a certain threshold. Given thresholds of 30 and 60 degrees, the angular deviation SR is 72.4% and 82.2%, respectively, i.e., 72.4% and 82.2% of the selected viewpoints at the first step forms an angle with the ground truth viewpoint that is less than or equal to 30 and 60 degrees. In other words, the multi-modal LLM can generally identify the direction correctly given the instruction in most cases. Navigation errors occur because the LLM cannot distinguish the small angular deviations between two nearby choices.

Moreover, in the first step of the navigation on the R2R val-unseen dataset, which contains 2349 instructions, the probe operation contributes 2.7% strict SR improvement. More specifically, for the first step of all the 2349 instructions, the probe operation makes 119 corrections and 55 mistakes. There are 119 cases in which the main query selects a neighbor to the ground truth viewpoint, and then the probe operation corrects it. Please note that the correctness of the first step is important for the navigation because in most cases the ground truth path rarely involves significant turns. This can be supported by the fact that the probe operation still improves the SR by 1.7% after the entire navigation (see Table 3).

6 Ablation on subtask prompts

We have three different subtasks in our method: regular, directional, and stair-related subtasks. In this section,

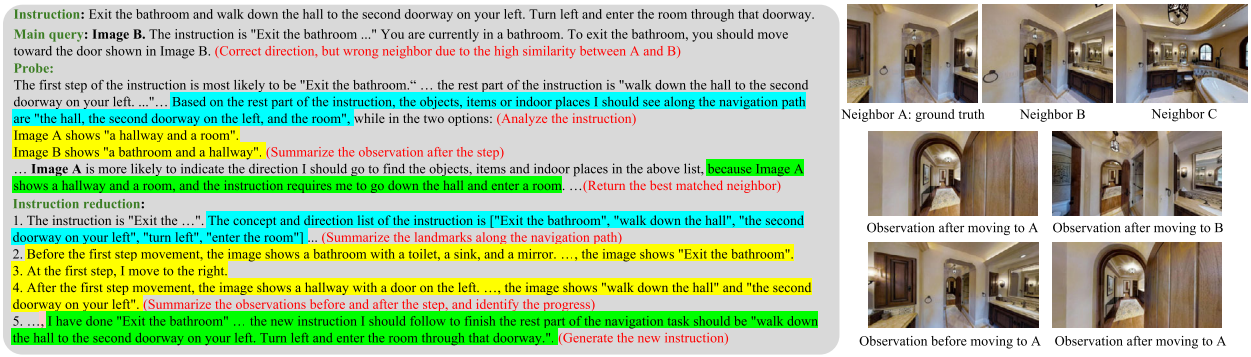

Figure 4 A successful case on the R2R dataset that shows the components of our method. The case demonstrates the multi-modal LLM's ability in multi-modal analysis based on both observations and instruction context. The instruction analysis (blue), the observation summary (yellow), and the conclusion (green) in the LLM's response are highlighted. The images on the right sides are the observations provided to the LLMs for queries

Table 5 The performance contributions of three different subtasks in our method. They are regular, directional, and stair-related subtasks

Components			Performance	
Regular	Directional	Stair	SR↑	OSR↑
✓			35.1	48.7
✓	✓		36.2	50.8
✓	✓	✓	39.7	55.7

we study the performance contribution of each subtask, and the experimental results are presented in Table 5. As shown in Table 5, both directional and stair-related subtasks contribute to performance improvement.

6.1 Case study

We present a successful example on the R2R dataset in Fig. 4 to demonstrate the multi-modal reasoning ability of multi-modal LLMs in VLN tasks. As shown in Fig. 4, given the instruction, the main query successfully obtains the correct direction, but the selected viewpoint is incorrect due to the high similarity between observations toward neighbors A and B. Then, the probe operation analyzes the observations after moving to A and B, respectively, and discover that A is more likely to indicate the direction to finish the rest of the VLN task. After the first step, the instruction reduction process analyzes the landmarks along the navigation path and summarizes the observations before and after the first step. Then, the agent moves to the doorway at the first step and is facing the hallway leading to another area. Finally, the instruction reduction query generates the new instruction for the next step correctly.

7 Conclusion

In this paper, we propose a novel multi-modal LLM-based agent, NavGemini for VLN in a zero-shot manner. NavGemini decomposes the VLN task into a series of subtasks and constructs an elaborate prompt management system to query the multi-modal LLM during navigation. To better utilize the multi-modal LLMs at a lower cost and reduce the prompt length for a higher-quality response, NavGemini uses an instruction reduction strategy to eliminate the completed part of the given instruction and perform the VLN task, focusing on the first step of the reduced instruction during navigation. Moreover, NavGemini introduces termination check and selection, backtracking, and probe operations to further improve the agent's performance. The higher performance in extensive experiments and the lower cost demonstrate the superiority and the potential of our method in zero-shot VLN.

Abbreviations

API, application programming interface; COT, chain of thought; GPT, generative pre-trained transformer; LLM, large language model; NE, navigation error; OSR, oracle success rate; RPM, request per minute; SPL, success rate penalized by path length; SR, success rate; TL, total length; VLN, vision-and-language navigation.

Acknowledgements

This work is supported by the National Key R&D Program of China (Nos. 2024YFB3908503 and 2024YFB3908500) and by the National Natural Science Foundation of China (No. 62322608).

Author contributions

GZ collected data and performed the experiment. GZ, GL, and YY contributed significantly to the analysis and manuscript preparation. GZ and GL performed the data analyses and wrote the manuscript. GL helped perform the analysis with constructive discussions. YY revised the manuscript. All authors read and approved the final manuscript.

Funding information

This work was supported by the National Natural Science Foundation of China (No. 62322608).

Data availability

The two VLN datasets analyzed in this study, R2R [9] and REVERIE [15], are available at https://bringmeaspoon.org/ and https://yuankaiqi.github.io/REVERIE_Challenge/dataset.html, respectively.

Declarations

Competing interests
The authors declare no competing interests.

Received: 25 June 2025 Revised: 14 December 2025
Accepted: 16 December 2025 Published online: 06 January 2026

References

1. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). LLaMA: open and efficient foundation language models. arXiv preprint. arXiv:2302.13971.
2. OpenAI (2023). GPT-4 technical report. arXiv preprint. arXiv:2303.08774.
3. Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., et al. (2023). PaLM 2 technical report. arXiv preprint. arXiv:2305.10403.
4. OpenAI (2023). GPT-4V(ision) system card. OpenAI. Retrieved November 1, 2025, from https://cdn.openai.com/papers/GPTV_System_Card.pdf.
5. Huang, W., Wang, C., Zhang, R., Li, Y., Wu, J., & Li, F.-F. (2023). VoxPoser: composable 3D value maps for robotic manipulation with language models. In J. Tan, M. Toussaint, & K. Darvish (Eds.), *Proceedings of the 7th conference on robot learning* (pp. 540–562). Retrieved November 1, 2025, from https://proceedings.mlr.press/v229/huang23c.html.
6. Mu, Y., Zhang, Q., Hu, M., Wang, W., Ding, M., Jin, J., Wang, B., Dai, J., Qiao, Y., & Luo, P. (2023). EmbodiedGPT: vision-language pre-training via embodied chain of thought. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Proceedings of the 37th international conference on neural information processing systems* (pp. 25081–25094). Red Hook: Curran Associates.
7. Ichter, B., Brohan, A., Chebotar, Y., Finn, C., Hausman, K., Herzog, A., Ho, D., Ibarz, J., Irpan, A., Jang, E., et al. (2022). Do as I can, not as I say: grounding language in robotic affordances. In K. Liu, D. Kulic, & J. Ichnowski (Eds.), *Proceedings of the 6th conference on robot learning* (pp. 287–318). Retrieved November 1, 2025, from https://proceedings.mlr.press/v205/ichter23a.html.
8. Xie, J., Chen, Z., Zhang, R., & Li, G. (2025). Large multimodal agents: a survey. *Visual Intelligence*, *3*, 24.
9. Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I. D., Gould, S., & van den Hengel, A. (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* (pp. 3674–3683). Piscataway: IEEE.
10. Krantz, J., Wijmans, E., Majumdar, A., Batra, D., & Lee, S. (2020). Beyond the nav-graph: vision-and-language navigation in continuous environments. In A. Vedaldi, H. Bischof, T. Brox, & J. Frahm (Eds.), *Proceedings of the 16th European conference on computer vision* (pp. 104–120). Cham: Springer.
11. Zhou, G., Hong, Y., & Wu, Q. (2024). NavGPT: explicit reasoning in vision-and-language navigation with large language models. In M. J. Wooldridge, J. G. Dy, & S. Natarajan (Eds.), *Proceedings of the 38th AAAI conference on artificial intelligence* (pp. 7641–7649). Palo Alto: AAAI Press.
12. Chen, J., Lin, B., Xu, R., Chai, Z., Liang, X., & Wong, K. K. (2024). MapGPT: map-guided prompting with adaptive path planning for vision-and-language navigation. In L. Ku, A. Martins, & V. Srikumar (Eds.), *Proceedings of the 62nd annual meeting of the Association for Computational Linguistics* (pp. 9796–9810). Stroudsburg: ACL.
13. Li, J., Li, D., Savarese, S., & Hoi, S. C. H. (2023). BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett (Eds.), *Proceedings of the international conference on machine learning* (pp. 19730–19742). Retrieved December 1, 2025, from https://proceedings.mlr.press/v202/li23q.html.
14. Gemini Team (2023). Gemini: a family of highly capable multimodal models. arXiv preprint. arXiv:2312.11805.
15. Qi, Y., Wu, Q., Anderson, P., Wang, X., Wang, W. Y., Shen, C., & van den Hengel, A. (2020). REVERIE: remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9979–9988). Piscataway: IEEE.
16. Ku, A., Anderson, P., Patel, R., Ie, E., & Baldridge, J. (2020). Room-across-room: multilingual vision-and-language navigation with dense spatiotemporal grounding. In B. Webber, T. Cohn, Y. He, & Y. Liu (Eds.), *Proceedings of the 2020 conference on empirical methods in natural language processing* (pp. 4392–4412). Stroudsburg: ACL.
17. Thomason, J., Murray, M., Cakmak, M., & Zettlemoyer, L. (2019). Vision-and-dialog navigation. In L. P. Kaelbling, D. Kragic, & K. Sugiura (Eds.), *Proceedings of the 3rd conference on robot learning* (pp. 394–406). Retrieved November 1, 2025, from http://proceedings.mlr.press/v100/thomason20a.html.
18. Zhu, F., Liang, X., Zhu, Y., Yu, Q., Chang, X., & Liang, X. (2021). SOON: scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12689–12699). Piscataway: IEEE.
19. Liu, C., Zhu, F., Chang, X., Liang, X., Ge, Z., & Shen, Y. (2021). Vision-language navigation with random environmental mixup. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 1624–1634). Piscataway: IEEE.
20. Wang, S., Montgomery, C., Orbay, J., Birodkar, V., Faust, A., Gur, I., Jaques, N., Waters, A., Baldridge, J., & Anderson, P. (2022). Less is more: generating grounded navigation instructions from landmarks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15407–15417). Piscataway: IEEE.
21. Chen, S., Guhur, P., Schmid, C., & Laptev, I. (2021). History aware multimodal transformer for vision-and-language navigation. In M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, & J. W. Vaughan (Eds.), *Proceedings of the 35th international conference on neural information processing systems* (pp. 5834–5847). Red Hook: Curran Associates.
22. Wang, H., Wang, W., Liang, W., Xiong, C., & Shen, J. (2021). Structured scene memory for vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 8455–8464). Piscataway: IEEE.
23. Pashevich, A., Schmid, C., & Sun, C. (2021). Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 15922–15932). Piscataway: IEEE.
24. Majumdar, A., Shrivastava, A., Lee, S., Anderson, P., Parikh, D., & Batra, D. (2020). Improving vision-and-language navigation with image-text pairs from the web. In A. Vedaldi, H. Bischof, T. Brox, & J. Frahm (Eds.), *Proceedings of the 16th European conference on computer vision* (pp. 259–274). Cham: Springer.
25. Hao, W., Li, C., Li, X., Carin, L., & Gao, J. (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* (pp. 13134–13143). Piscataway: IEEE.
26. Chen, S., Guhur, P., Tapaswi, M., Schmid, C., & Laptev, I. (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* (pp. 16516–16526). Piscataway: IEEE.
27. Qiao, Y., Lyu, W., Wang, H., Wang, Z., Li, Z., Zhang, Y., Tan, M., & Wu, Q. (2025). Open-Nav: exploring zero-shot vision-and-language navigation in continuous environment with open-source LLMs. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 6710–6717). Piscataway: IEEE.
28. Long, Y., Li, X., Cai, W., & Dong, H. (2024). Discuss before moving: visual language navigation via multi-expert discussions. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 17380–17387). Piscataway: IEEE.
29. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, & R. Garnett (Eds.), *Proceedings of the 31st international conference on neural information processing systems* (pp. 5998–6008). Red Hook: Curran Associates.
30. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, & H. Lin (Eds.), *Proceedings of the 34th international conference on neural information processing systems* (pp. 1877–1901). Red Hook: Curran Associates.
31. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. (2023). PaLM: scaling language modeling with pathways. *Journal of Machine Learning Research*, *24*, 240.
32. Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M. T., Li, X., Lin, X. V., et al. (2022). OPT: open pre-trained transformer language models. arXiv preprint. arXiv:2205.01068.

33. Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., & Le, Q. V. (2022). Finetuned language models are zero-shot learners. In *Proceedings of the 10th international conference on learning representations* (pp. 1–46). Retrieved December 1, 2025, from https://openreview.net/forum?id=gEZrGCozdqR.
34. Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., et al. (2023). Vicuna: an open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. Retrieved March 30, 2025, from https://lmsys.org/blog/2023-03-30-vicuna.
35. Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-XL: attentive language models beyond a fixed-length context. In A. Korhonen, D. R. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th conference of the Association for Computational Linguistics* (pp. 2978–2988). Stroudsburg: ACL.
36. Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., et al. (2022). GLaM: efficient scaling of language models with mixture-of-experts. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, & S. Sabato (Eds.), *Proceedings of the international conference on machine learning* (pp. 5547–5569). Retrieved December 1, 2025, from https://proceedings.mlr.press/v162/du22c.html.
37. Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch transformers: scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, *23*, 120.
38. Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Proceedings of the 36th international conference on neural information processing systems* (pp. 30016–30030). Red Hook: Curran Associates.
39. Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, H. F., Aslanides, J., Henderson, S., Ring, R., Young, S., et al. (2021). Scaling language models: methods, analysis & insights from training gopher. arXiv preprint. arXiv:2112.11446.
40. Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., et al. (2024). Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, *25*, 70.
41. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Proceedings of the 36th international conference on neural information processing systems* (pp. 27730–27744). Red Hook: Curran Associates.
42. Wu, C., Yin, S., Qi, W., Wang, X., Tang, Z., & Duan, N. (2023). Visual ChatGPT: talking, drawing and editing with visual foundation models. arXiv preprint. arXiv:2303.04671.
43. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., & Zhuang, Y. (2023). HuggingGPT: solving AI tasks with ChatGPT and its friends in hugging face. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Proceedings of the 37th international conference on neural information processing systems* (pp. 38154–38180). Red Hook: Curran Associates.
44. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Hambro, E., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: language models can teach themselves to use tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Proceedings of the 37th international conference on neural information processing systems* (pp. 68539–68551). Red Hook: Curran Associates.
45. Peng, B., Galley, M., He, P., Cheng, H., Xie, Y., Hu, Y., Huang, Q., Liden, L., Yu, Z., Chen, W., et al. (2023). Check your facts and try again: improving large language models with external knowledge and automated feedback. arXiv preprint. arXiv:2302.12813.
46. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., & Cao, Y. (2023). React: synergizing reasoning and acting in language models. In *Proceedings of the 11th international conference on learning representations* (pp. 1–33). Retrieved December 1, 2025, from https://openreview.net/forum?id=WE_vluYUL-X.
47. Karpas, E., Abend, O., Belinkov, Y., Lenz, B., Lieber, O., Ratner, N., Shoham, Y., Bata, H., Levine, Y., Leyton-Brown, K., et al. (2022). MRKL systems: a modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning. arXiv preprint. arXiv:2205.00445.
48. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Proceedings of the 36th international conference on neural information processing systems* (pp. 24824–24837). Red Hook: Curran Associates.
49. Alayrac, J., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. (2022). Flamingo: a visual language model for few-shot learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Proceedings of the 36th international conference on neural information processing systems* (pp. 23716–23736). Red Hook: Curran Associates.
50. Awadalla, A., Gao, I., Gardner, J., Hessel, J., Hanafy, Y., Zhu, W., Marathe, K., Bitton, Y., Gadre, S. Y., Sagawa, S., et al. (2023). OpenFlamingo: an open-source framework for training large autoregressive vision-language models. arXiv preprint. arXiv:2308.01390.
51. Bavishi, R., Elsen, E., Hawthorne, C., Nye, M., Odena, A., Somani, A., & Taşırlar, S. (2024). Fuyu-8B: a multimodal architecture for AI agents. Retrieved December 1, 2025, from https://www.adept.ai/blog/fuyu-8b.
52. Lin, J., Yin, H., Ping, W., Molchanov, P., Shoeybi, M., & Han, S. (2024). VILA: on pre-training for visual language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 26679–26689). Piscataway: IEEE.
53. Bai, J., Bai, S., Chu, Y., Cui, Z., Dang, K., Deng, X., Fan, Y., Ge, W., Han, Y., Huang, F., et al. (2023). Qwen technical report. arXiv preprint. arXiv:2309.16609.
54. Chen, X., Djolonga, J., Padlewski, P., Mustafa, B., Changpinyo, S., Wu, J., Ruiz, C. R., Goodman, S., Wang, X., Tay, Y., et al. (2023). PaLI-X: on scaling up a multilingual vision and language model. arXiv preprint. arXiv:2305.18565.
55. Driess, D., Xia, F., Sajjadi, M. S. M., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al. (2023). PaLM-E: an embodied multimodal language model. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett (Eds.), *Proceedings of the international conference on machine learning* (pp. 8469–8488). Retrieved December 1, 2025, from https://proceedings.mlr.press/v202/driess23a.html.
56. Liu, H., Li, C., Wu, Q., & Lee, Y. J. (2023). Visual instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Proceedings of the 37th international conference on neural information processing systems* (pp. 34892–34916). Red Hook: Curran Associates.
57. Ye, Q., Xu, H., Xu, G., Ye, J., Yan, M., Zhou, Y., Wang, J., Hu, A., Shi, P., Shi, Y., et al. (2023). mPLUG-Owl: modularization empowers large language models with multimodality. arXiv preprint. arXiv:2304.14178.
58. Zhu, D., Chen, J., Shen, X., Li, X., & Elhoseiny, M. (2024). MiniGPT-4: enhancing vision-language understanding with advanced large language models. In *Proceedings of the 12th international conference on learning representations* (pp. 1–17). Retrieved December 1, 2025, from https://openreview.net/forum?id=1tZbq88f27.
59. Pan, B., Panda, R., Jin, S., Feris, R., Oliva, A., Isola, P., & Kim, Y. (2024). LangNav: language as a perceptual representation for navigation. In K. Duh, H. Gómez-Adorno, & S. Bethard (Eds.), *Findings of the proceedings of the 2024 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies* (pp. 950–974). Stroudsburg: ACL.
60. Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., et al. (2023). RT-2: vision-language-action models transfer web knowledge to robotic control. In J. Tan, M. Toussaint, & K. Darvish (Eds.), *Proceedings of the 7th conference on robot learning* (pp. 2165–2183). Retrieved November 1, 2025, from https://proceedings.mlr.press/v229/zitkovich23a.html.
61. Schumann, R., Zhu, W., Feng, W., Fu, T., Riezler, S., & Wang, W. Y. (2024). VELMA: verbalization embodiment of LLM agents for vision and language navigation in street view. In M. J. Wooldridge, J. G. Dy, & S. Natarajan (Eds.), *Proceedings of the 38th AAAI conference on artificial intelligence* (pp. 18924–18933). Palo Alto: AAAI Press.
62. Hu, Y., Lin, F., Zhang, T., Yi, L., & Gao, Y. (2023). Look before you leap: unveiling the power of GPT-4V in robotic vision-language planning. arXiv preprint. arXiv:2311.17842.
63. Huang, W., Abbeel, P., Pathak, D., & Mordatch, I. (2022). Language models as zero-shot planners: extracting actionable knowledge for embodied agents. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, & S. Sabato (Eds.), *Proceedings of the international conference on machine learning* (pp. 9118–9147). Retrieved December 1, 2025, from https://proceedings.mlr.press/v162/huang22a.html.
64. Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., & Garg, A. (2023). ProgPrompt: generating situated robot task

plans using large language models. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 11523–11530). Piscataway: IEEE.
65. Wu, J., Antonova, R., Kan, A., Lepert, M., Zeng, A., Song, S., Bohg, J., Rusinkiewicz, S., & Funkhouser, T. A. (2023). TidyBot: personalized robot assistance with large language models. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (pp. 3546–3553). Piscataway: IEEE.
66. Sharma, P., Torralba, A., & Andreas, J. (2022). Skill induction and planning with latent language. In S. Muresan, P. Nakov, & A. Villavicencio (Eds.), *Proceedings of the 60th annual meeting of the Association for Computational Linguistics* (pp. 1713–1726). Stroudsburg: ACL.
67. Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., & Zeng, A. (2023). Code as policies: language model programs for embodied control. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 9493–9500). Piscataway: IEEE.
68. Tam, A. C., Rabinowitz, N. C., Lampinen, A. K., Roy, N. A., Chan, S. C. Y., Strouse, D., Wang, J., Banino, A., & Hill, F. (2022). Semantic exploration from language abstractions and pretrained representations. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Proceedings of the 36th international conference on neural information processing systems* (pp. 25377–25389). Red Hook: Curran Associates.
69. Shah, D., Osinski, B., Ichter, B., & Levine, S. (2022). LM-Nav: robotic navigation with large pre-trained models of language, vision, and action. In K. Liu, D. Kulic, & J. Ichnowski (Eds.), *Proceedings of the 6th conference on robot learning* (pp. 492–504). Retrieved November 1, 2025, from https://proceedings.mlr.press/v205/shah23b.html.
70. Huang, C., Mees, O., Zeng, A., & Burgard, W. (2023). Visual language maps for robot navigation. In *Proceedings of the IEEE international conference on robotics and automation* (pp. 10608–10615). Piscataway: IEEE.
71. Zhou, K., Zheng, K., Pryor, C., Shen, Y., Jin, H., Getoor, L., & Wang, X. E. (2023). ESC: exploration with soft commonsense constraints for zero-shot object navigation. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett (Eds.), *Proceedings of the international conference on machine learning* (pp. 42829–42842). Retrieved December 1, 2025, from https://proceedings.mlr.press/v202/zhou23r.html.
72. Pryzant, R., Iter, D., Li, J., Lee, Y. T., Zhu, C., & Zeng, M. (2023). Automatic prompt optimization with "gradient descent" and beam search. In H. Bouamor, J. Pino, & K. Bali (Eds.), *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 7957–7968). Stroudsburg: ACL.
73. Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: techniques and applications. arXiv preprint. arXiv:2402.07927.
74. Ramnath, K., Zhou, K., Guan, S., Mishra, S. S., Qi, X., Shen, Z., Wang, S., Woo, S., Jeoung, S., Wang, Y., et al. (2025). A systematic survey of automatic prompt optimization techniques. arXiv preprint. arXiv:2502.16923.
75. Chang, A. X., Dai, A., Funkhouser, T. A., Halber, M., Nießner, M., Savva, M., Song, S., Zeng, A., & Zhang, Y. (2017). Matterport3D: learning from RGB-D data in indoor environments. In *Proceedings of the international conference on 3D vision* (pp. 667–676). Piscataway: IEEE.
76. Fried, D., Hu, R., Cirik, V., Rohrbach, A., Andreas, J., Morency, L., Berg-Kirkpatrick, T., Saenko, K., Klein, D., & Darrell, T. (2018). Speaker-follower models for vision-and-language navigation. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Proceedings of the 32nd international conference on neural information processing systems* (pp. 3318–3329). Red Hook: Curran Associates.
77. Tan, H., Yu, L., & Bansal, M. (2019). Learning to navigate unseen environments: back translation with environmental dropout. In J. Burstein, C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 conference of the North American Chapter of the Association for Computational Linguistics: human language technologies* (pp. 2610–2621). Stroudsburg: ACL.
78. Hong, Y., Wu, Q., Qi, Y., Opazo, C. R., & Gould, S. (2021). VLN BERT: a recurrent vision-and-language BERT for navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1643–1653). Piscataway: IEEE.
79. Tan, H., & Bansal, M. (2019). LXMERT: learning cross-modality encoder representations from transformers. In K. Inui, J. Jiang, V. Ng, & X. Wan (Eds.), *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing* (pp. 5099–5110). Stroudsburg: ACL.

Publisher's note