LangNav: Language as a Perceptual Representation for Navigation

Bowen Pan◇ **Rameswar Panda**† **SouYoung Jin**⋆ **Rogerio Feris**†
Aude Oliva◇† **Phillip Isola**◇ **Yoon Kim**◇
◇MIT CSAIL, †MIT-IBM Watson AI Lab, ⋆Dartmouth College
{bpan, oliva, phillipi, yoonkim}@mit.edu,
rpanda@ibm.com, rsferis@us.ibm.com, souyoung.jin@dartmouth.edu

Abstract

We explore the use of language as a perceptual representation for vision-and-language navigation (VLN), with a focus on low-data settings. Our approach uses off-the-shelf vision systems for image captioning and object detection to convert an agent's egocentric panoramic view at each time step into natural language descriptions. We then finetune a pretrained language model to select an action, based on the current view and the trajectory history, that would best fulfill the navigation instructions. In contrast to the standard setup which adapts a pretrained language model to work directly with continuous visual features from pretrained vision models, our approach instead uses (discrete) language as the perceptual representation. We explore several use cases of our language-based navigation (LangNav) approach on the R2R VLN benchmark: generating synthetic trajectories from a prompted language model (GPT-4) with which to finetune a smaller language model; domain transfer where we transfer a policy learned on one simulated environment (ALFRED) to another (more realistic) environment (R2R); and combining both vision- and language-based representations for VLN. Our approach is found to improve upon baselines that rely on visual features in settings where only a few expert trajectories (10-100) are available, demonstrating the potential of language as a perceptual representation for navigation.

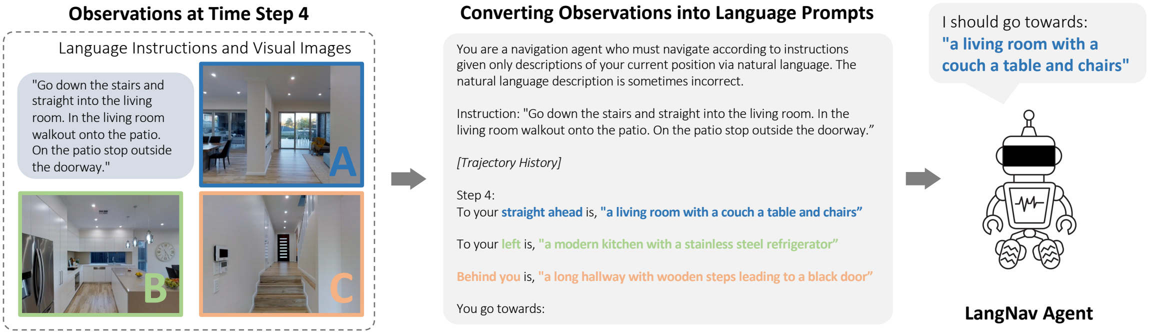
1 Introduction

Applications of large language models (LMs) to non-linguistic embodied tasks have generally focused on using the implicit world knowledge within LMs to predict sub-tasks and actions for planning (Ahn et al., 2022; Huang et al., 2022b,a; Singh et al., 2022). For instance, recent work has shown that LMs can be prompted to create a list of actions (e.g., `GoToBathroom`, `LocateToothbrush`) given a high-level goal given in natural language (e.g., "brush teeth") (Huang et al., 2022a). These approaches rely on the LM's priors on action sequences and inter-object correlations acquired through large-scale pretraining (Zhou et al., 2023b; Li et al., 2023; Zhao et al., 2023), and it has not been clear whether text-only models can be finetuned for tasks such as vision-and-language navigation which requires an egocentric agent follow instructions to navigate a 3D environment using visual input.

To be clear, there *is* a substantial body of work on using pretrained LMs for vision-and-language navigation tasks (Hong et al., 2021; Qi et al., 2021; Qiao et al., 2022, *inter alia*). The standard approach is to use a pretrained LM over the natural language instructions to extract text features that are combined with the agent's perceptual representations, which are given by continuous image features extracted from pretrained vision models (Wang et al., 2019; Hao et al., 2020). While effective in data-rich regimes, the direct use of vision features makes the approach difficult to apply in cases where only a few labeled trajectories exist (e.g., 10 trajectories), as these approaches need to learn a full joint vision-language module that combines a pretrained vision model with a pretrained text model. A popular strategy in such low data regimes is to generate synthetic data or transfer knowledge from other domains. However, generating realistic perception data is itself a difficult task, and domain transfer with models that rely purely on visual features can overfit to the non-transferable features (Anderson et al., 2021).

This paper explores an alternative approach for vision-and-language navigation by exploiting language itself as the perceptual representation space. Our approach uses off-the-shelf vision models to obtain textual descriptions of the agent's egocentric panoramic view. The text descriptions are then fed to an LM which must select the next action given the instruction and (text descriptions of) the previous actions or observations. See Figure 1 for an overview. The use of language to represent an

Findings of the Association for Computational Linguistics: NAACL 2024, pages 950–974
June 16-21, 2024 ©2024 Association for Computational Linguistics


Observations at Time Step 4
Language Instructions and Visual Images
"Go down the stairs and straight into the living room. In the living room walkout onto the patio. On the patio stop outside the doorway."
A
B
C
Converting Observations into Language Prompts
I should go towards: "a living room with a couch a table and chairs"
LangNav Agent

You are a navigation agent who must navigate according to instructions given only descriptions of your current position via natural language. The natural language description is sometimes incorrect.

Instruction: "Go down the stairs and straight into the living room. In the living room walkout onto the patio. On the patio stop outside the doorway."

[Trajectory History]

Step 4:
To your **straight ahead** is, **"a living room with a couch a table and chairs"**

To your left is, "a modern kitchen with a stainless steel refrigerator"

Behind you is, "a long hallway with wooden steps leading to a black door"

You go towards:

Figure 1: Overview of language-based navigation (LangNav). We describe the task instructions and visual observations (from off-the-shelf vision systems) through text. A language model is then finetuned to predict which direction to move towards based on the language descriptions. Here, views **A**, **B**, and C correspond to the front, left, and rear views of the agent.

agent's perceptual field makes it possible to readily utilize the myriad capabilities of language models, especially when the training data is limited. In our first case study, we show how we can use a small amount of seed training data (10-100 trajectories) to cheaply obtain synthetic "trajectories" from a powerful but closed-source LM (GPT-4; OpenAI, 2023). We find that finetuning a smaller language model (LLaMA/LLaMA2; Touvron et al., 2023a,b) on the generated trajectories mixed with the original seed data results in a langauge-based navigation agent that outperforms a vision-based agent that is finetuned on the same seed data. In our second study, we explore the use of language as a domain-invariant representation to perform domain transfer, where we transfer an agent trained on a computer-generated environment (ALFRED; Shridhar et al., 2020) to the real-world R2R (Anderson et al., 2018b) environment. Insofar as language is hypothesized to have co-evolved with the human brain to enable efficient communication (Deacon, 1997), it naturally abstracts away low-level perceptual details, and we indeed find that LangNav exhibits improved transfer compared to the vision-based agent. We further show that language can provide further benefits even in the presence of vision-based features. Our results collectively suggest that language as a perceptual representation can be helpful in the low-data navigation settings.

2 Background: Room-to-Room Vision-language Navigation

A popular testbed for vision-and-language navigation (VLN) is the room-to-room dataset (R2R; Anderson et al., 2018b), in which an agent must perceive and navigate a real-world 3D environment based on a language instruction U and an initial state S_0. At each time step t, the agent uses the current observation O_t, the original language instructions U, and the trajectory history H_t, to predict the panoramic action a_t. The current observation is given by a set of panoramic images that describe the agent's egocentric view, i.e., $O_t = \{I_{t,0}, ..., I_{t,V}\}$ where V corresponds to the number of discretized view angles.[1] The panoramic action a_t corresponds to which navigable view in O_t to go towards, i.e., $a_t \in O_t$. After selecting an action, the state transitions from S_t to S_{t+1}. The aim is to output the command `STOP` after reaching the goal G specified by U in state S_0.

The standard approach in R2R is to process the panoramic images $\{I_{t,0}, ..., I_{t,V}\}$ with a pretrained visual encoder E_v to extract continuous visual features $F_{t,v} = \{E_v(I_{t,0}), ..., E(I_{t,V})\}$. The language instruction is typically processed by a pretrained language encoder E_l (e.g., BERT (Devlin et al., 2019)) to extract the language features $F_l = E_l(U)$. These features, along with a hidden state representation of the trajectory history h_{t-1}, are fed to a joint vision-language module (e.g., another Transformer) that attends over $\{I_{t,0}, ..., I_{t,V}\}$ to select the action a_t.

3 Language as a Perceptual Representation for Navigation

We begin by describing the perception-to-text models employed for converting visual observations into text (§ 3.1). We then discuss the prompt templates for converting the text into natural language (§ 3.2), followed by a description of the offline imitation learning algorithm for learning (§ 3.3).

---

[1] In R2R this can be as many as 36 (12 headings and 3 elevations). However we follow previous works only consider the navigable views, which is often many fewer than 36.

3.1 Vision-to-text System

We use off-the-shelf vision models to convert visual observations into language descriptions. Specifically, we use an image captioning model (BLIP; Li et al., 2022a) and an object detection model (Deformable DETR; Zhu et al., 2020) over each view angle $I_{t,j}$ to obtain the text descriptions,

$$C_{t,j} = \text{IMAGECAPTIONER}(I_{t,j}),$$
$$x_{t,j,0}, \ldots, x_{t,j,M} = \text{OBJECTDETECTOR}(I_{t,j}),$$

where M is the number of detected objects.[2]

3.2 Prompt Templates

Figure 1 illustrates how the image caption and the detected objects are combined via templates to construct pieces of text on which to condition the language model. Based on the prompt template, the language model will be finetuned on the (language representations of) output actions $\{a_1, \ldots, a_T\}$. We briefly describe the prompt template (see appendix G for a full example).

Task description D. The task description is given by:

```
You are a navigation agent who must
navigate according to instructions given
only descriptions of your current [...].
```

Navigation instruction U. The navigation instruction, which provides instructions to the agent on how to reach the goal, can be from R2R (our main dataset), synthesized by GPT-4 (for data augmentation), or ALFRED (for domain transfer). An example instruction from R2R is:

```
Travel forward past the wall with all
the light switches and into the first
room on your right.
```

Current observation O_t. We use templates to convert the image caption $C_{t,j}$ and objects obtained $x_{t,j,0}, \cdots, x_{t,j,M}$ from $I_{t,j}$ (§ 3.1). For instance, if the agent is facing a heading of 90 degrees and an elevation of 0 degrees and there is a candidate navigable direction $I_{t,j}$ located at a heading of 120 degrees and an elevation of 0 degrees, the text description for this view angle would be:

```
To your 30 degree right is “{C_{t,j}}”.
Details: {x_{t,j,0}}, ..., {x_{t,j,M}}.
```

We create such templates for all the navigable view angles $\{I_{t,0}, \ldots, I_{t,V}\}$.

Action a_t. Selecting an action involves choosing a navigable view out of O_t to move towards, i.e., $a_t \in O_t$. For example, suppose $a_t = I_{t,j}$, i.e., the agent decided to go to the j-th view angle. Then this is recorded as:

```
You go towards: “{C_{t,j}}”
```

To actually have the agent generate a_t we simply decode from an LM's distribution, $p_{\text{LM}}(\cdot \mid D, U, H_t, O_t)$, via greedy decoding. Here $H_t = \{O_i, a_i\}_{i=0}^{t-1}$ encodes the observation and action trajectory.[3]

Updating trajectory history H_t. We update the observation and action trajectory history via appending the text representations of O_t and a_t to H_t:

```
Step {t}: To your {direction_1} is
{caption_1}; To your {direction_2}
is {caption_2}; [...]; You chose:
{caption_of_selected_direction}.
```

This history serves to inform the model about its current position within the high-level instruction, enabling it to make more informed decisions when selecting actions.

3.3 Imitation Learning on Demonstrations

We create an instruction-following dataset by transforming the expert trajectory from the original dataset into instruction-following demonstrations. Formally, let $\mathcal{D} = \{W^{(i)}\}_{i=1}^N$ be the set of training trajectories, where each $W^{(i)}$ can be represented as a natural language sequence from the above template, $W^{(i)} = (D^{(i)}, U^{(i)}, H_1^{(i)}, O_1^{(i)}, a_1^{(i)}, \ldots, H_{T^{(i)}}^{(i)}, O_{T^{(i)}}^{(i)}, a_{T^{(i)}}^{(i)})$. Here $T^{(i)}$ is the number of actions in the example $W^{(i)}$, which is typically between 5 to 7. Given the above, we optimize the log likelihood of the (language descriptions of) actions, i.e., the objective for trajectory $W^{(i)}$ is given by, $\sum_{t=1}^{T^{(i)}} \log p_{\text{LM}}(a_t^{(i)} \mid D^{(i)}, U^{(i)}, H_t^{(i)}, O_t^{(i)})$.

While behavior cloning on gold trajectories is simple, it is prone to error propagation. In particular, the history trajectory is obtained by a shortest-path algorithm (which has knowledge of the goal)

---

[2] We did not experiment much with different off-the-shelf vision systems and quickly converged on these two models which seemed to produce reasonable results. Since LangNav separates perception from navigation, we expect that advances made in perception (e.g., through better captioning systems) will automatically result in improvements to our system, which is a nontrivial advantage of our approach compared to systems that entangle perception and navigation into a single model.

[3] In general we found the finetuned LM to have no issue generating from the set of navigable directions (i.e., $\{C_{t,0}, \ldots, C_{t,V}\}$) without constrained decoding.

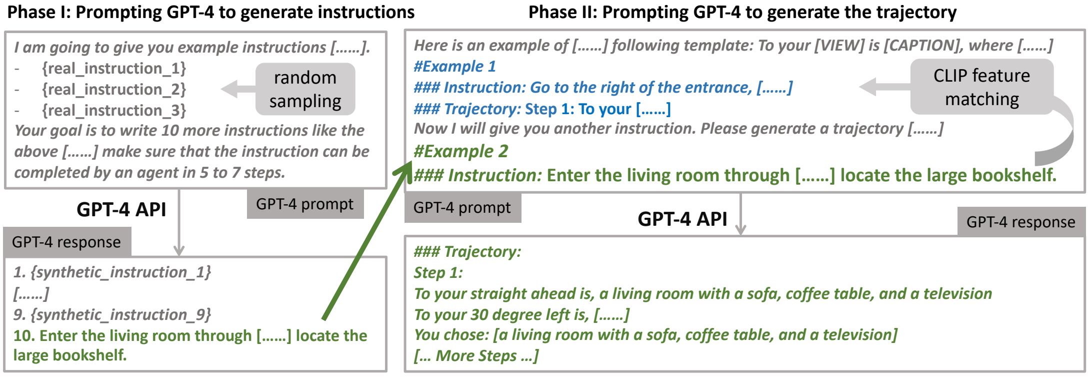

Phase I: Prompting GPT-4 to generate instructions
I am going to give you example instructions [……].
- {real_instruction_1}
- {real_instruction_2}
- {real_instruction_3}
Your goal is to write 10 more instructions like the above [……] make sure that the instruction can be completed by an agent in 5 to 7 steps.
random sampling
GPT-4 API
GPT-4 prompt
GPT-4 response
1. {synthetic_instruction_1}
[……]
9. {synthetic_instruction_9}
10. Enter the living room through [……] locate the large bookshelf.
Phase II: Prompting GPT-4 to generate the trajectory
Here is an example of [……] following template: To your [VIEW] is [CAPTION], where [……]
#Example 1
### Instruction: Go to the right of the entrance, [……]
### Trajectory: Step 1: To your [……]
CLIP feature matching
Now I will give you another instruction. Please generate a trajectory [……]
#Example 2
### Instruction: Enter the living room through [……] locate the large bookshelf.
GPT-4 prompt
GPT-4 API
GPT-4 response
### Trajectory:
Step 1:
To your straight ahead is, a living room with a sofa, coffee table, and a television
To your 30 degree left is, [……]
You chose: [a living room with a sofa, coffee table, and a television]
[… More Steps …]

Figure 2: Pipeline for generating synthetic navigation trajectories from GPT-4. We first prompt GPT-4 with 3 randomly sampled navigation instructions U to generate 10 more synthetic navigation instructions (Phase 1). Then for each generated navigation instruction, we prompt GPT-4 to generate the trajectory that fulfills the generated instruction (Phase 2). See appendix H for details.

and thus adheres closely to an optimal policy π^*. However, during prediction, trajectories can deviate significantly from the optimal policy, leading to a distribution shift that can adversely affect performance. To allow for the policy to recover from deviations from the optimal path, we adopt the following strategy to create our imitation learning dataset: (1) at each time step, we sample a random action with probability ρ; (2) once a random action is selected, we use the shortest-path algorithm to obtain the ground truth next action; (3) we repeat this process until the goal is reached; (4) once the goal is reached, this becomes part of the training demonstration data. (See appendix F for details.)

4 Empirical Study

Our primary experiments with LangNav target the low-data setting, motivated by the observation that obtaining annotated data for embodied tasks such as vision-language navigation can be very costly (often more so than is the case for text-only or vision-only tasks). Specifically, we are interested in learning the most performant system based on a small number (10 or 100) of in-domain seed navigation trajectories. We sample our seed trajectories from the Room-to-Room (R2R) dataset (Anderson et al., 2018b), a popular vision-and-language navigation dataset consisting of 21,567 navigation instructions in the Matterport3D environment. The dataset includes 90 scenes, with 61 scenes in the train and validation "seen" sets, and 11 scenes in the validation "unseen" set. Our 10-shot dataset is randomly sampled the train set within 1 scene, while our 100-shot dataset spans 2 scenes.

Evaluation. To contextualize our approach against prior work, we evaluate LangNav on both "seen" and "unseen" sets from R2R. The "seen" set contains scenes identical to the training set (but the instructions and trajectories differ). However, this distinction is less important for our low-data regime, since we only make use of 1 scene (for the 10-shot case) or 2 scenes (for the 100-shot case). I.e., the majority of scenes in the "seen" validation subset are actually never seen by the agent.

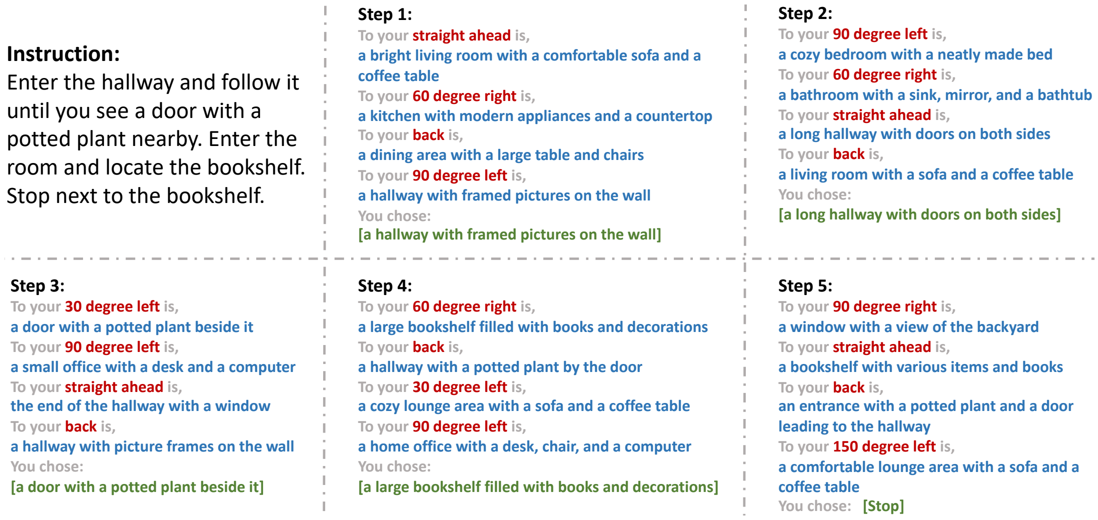
We use the standard R2R task performance metrics (Anderson et al., 2018a): *Navigation Error* (`NE`), the average distance between the agent's final position and the goal in meters; *Success Rate* (`SR`), the ratio of trajectories in which the agent stopped within 3 meters of the goal; *Oracle Success Rate* (`OSR`), the ratio of trajectories in which the agent stopped within 3 meters to the goal with a view of the goal; and *Success* weighted by the normalized inverse of the *Path Length* (`SPL`).

4.1 Case Study 1: Language Enables Efficient Synthetic Data Generation

In NLP, obtaining synthetic data from an appropriately prompted large LM with which to learn a smaller model has been shown to be an effective approach in data-scarce settings (Wang et al., 2021; Lang et al., 2022; Taori et al., 2023; Dai et al., 2023; Gunasekar et al., 2023, *inter alia*).[4] However, this approach is difficult to extend to non-linguistic perceptual tasks such as VLN since generating realistic perception data is itself difficult. In our first case study, we show that working in pure language space makes it possible to easily generate synthetic

---

[4] However see Gudibande et al. (2023) for a critical discussion of this approach.



Instruction:
Enter the hallway and follow it until you see a door with a potted plant nearby. Enter the room and locate the bookshelf. Stop next to the bookshelf.

Step 1:
To your straight ahead is,
a bright living room with a comfortable sofa and a coffee table
To your 60 degree right is,
a kitchen with modern appliances and a countertop
To your back is,
a dining area with a large table and chairs
To your 90 degree left is,
a hallway with framed pictures on the wall
You chose:
[a hallway with framed pictures on the wall]

Step 2:
To your 90 degree left is,
a cozy bedroom with a neatly made bed
To your 60 degree right is,
a bathroom with a sink, mirror, and a bathtub
To your straight ahead is,
a long hallway with doors on both sides
To your back is,
a living room with a sofa and a coffee table
You chose:
[a long hallway with doors on both sides]

Step 3:
To your 30 degree left is,
a door with a potted plant beside it
To your 90 degree left is,
a small office with a desk and a computer
To your straight ahead is,
the end of the hallway with a window
To your back is,
a hallway with picture frames on the wall
You chose:
[a door with a potted plant beside it]

Step 4:
To your 60 degree right is,
a large bookshelf filled with books and decorations
To your back is,
a hallway with a potted plant by the door
To your 30 degree left is,
a cozy lounge area with a sofa and a coffee table
To your 90 degree left is,
a home office with a desk, chair, and a computer
You chose:
[a large bookshelf filled with books and decorations]

Step 5:
To your 90 degree right is,
a window with a view of the backyard
To your straight ahead is,
a bookshelf with various items and books
To your back is,
an entrance with a potted plant and a door leading to the hallway
To your 150 degree left is,
a comfortable lounge area with a sofa and a coffee table
You chose: [Stop]

Figure 3: An example of a generated trajectory from GPT-4. The example demonstrates a generated trajectory by following the pipeline in Figure 2. See more examples in appendix K.

data from a large LM based on a few seed trajectories. We further show that finetuning a smaller LM on a mixture of synthetic and R2R trajectories improves upon vision-based models.

Synthetic trajectory generation. We generate synthetic trajectories by using only the 10 R2R trajectories from a single scene. In R2R each trajectory has 3 navigation instructions given by 3 different annotators. Thus we have 30 navigation instructions $\{U^{(i)}\}_{i=1}^{30}$ in total. Our data generation pipeline can be divided into two phases. In phase 1, we randomly choose 3 R2R instructions as prompt examples and ask GPT-4 to create 10 more instructions similar to the examples, as shown in Figure 2. In phase 2, for each generated instruction, we prompt GPT-4 to generate a trajectory to fulfill the instruction, conditioned on a real demonstration instruction and trajectory. The real trajectory is obtained by selecting the trajectory whose instruction is closest to the synthetic instruction based on the CLIP (Radford et al., 2021) text features. See Figure 2 for an overview and appendix H for the prompts.[5]

We present an illustrative example in Figure 3 to demonstrate some qualitative characteristics of generated trajectories. We find that the generated trajectories have: *strong real-world priors*, i.e., they exhibit adherence to real-world room-object and object-object correlations, as evident from descriptions like "`a bathroom with a sink, mirror, [...]`"; *spatial consistency*, where the examples maintain spatial consistency within the generated trajectories—for instance, in `Step 4`, the generated position identifies the door with a potted plant, consistent with its position in `Step 3`; and *rich descriptions*—the generated trajectories have descriptive captions and objects that do not only relate to the given instruction, which makes it possible to successfully navigate through language only.

Experimental setup. We compare LangNav, which is a LLaMA2-7b model finetuned on a mixture of the 10,000 synthetic trajectories and 10/100 real trajectories, against the following baselines: *1. Random walk*, which selects a random action at each time step; *2. GPT-4 (Zero-shot / Few-shot)*, where we prompt GPT-4 to complete the trajectory by changing the task description of the template in § 3.2 (see appendix I for the full prompt). For the few-shot baseline, due to the context length we use one full navigation trajectory as a demonstration example; *3. NavGPT*, a recent work that also uses language as a perceptual representation (via image captioning and object detection) to perform navigation, but purely with GPT-4 (Zhou et al., 2023a); *4. RecBert*, a vision-based method that adopts a recurrent architecture proposed by Hong et al. (2021) to keep track of the trajectory history; *5. DuET*, another vision-based method which additionally builds representations of the global map during learning (Chen et al., 2022); and *6. LLaMA2-7B*, a

---

[5] We cannot entirely rule out the possibility that the GPT-4 training set included the text instructions seen in R2R. However, while the text instructions may have been encountered, the trajectories were unlikely to have been encountered during pretraining since we used vision systems to obtain the captions/objects. Out of the 10,000 generated instructions, we did not find any instructions that were in the actual R2R dataset.

Methods	# real	Val Seen				Val Unseen			
		NE↓	OSR↑	SR↑	SPL↑	NE↓	OSR↑	SR↑	SPL↑
Random Walk	0	10.2	5	3	1	9.5	6	3	2
LLaMA2-7B (Zero-shot)	0	10.2	0	0	0	9.5	0	0	0
GPT-4 (Zero-shot)	0	10.5	15	9	8	10.2	17	10	8
GPT-4 (Few-shot)	1	10.1	17	10	9	9.9	22	13	11
NavGPT (Zhou et al., 2023a)	0	-	-	-	-	6.5	42	34	29
RecBert (Hong et al., 2021)	10	10.8	9	7	6	10.1	13	9	9
DuET (Chen et al., 2022)	10	10.0	21	14	12	9.9	20	12	11
LLaMA2-7B	10	10.2	15	11	10	9.6	16	11	9
LangNav (with LLaMA2-7B)	10	**7.5**	**39**	**31**	**27**	**7.0**	**42**	**32**	**28**
RecBert (Hong et al., 2021)	100	9.3	27	20	19	9.4	26	19	17
DuET (Chen et al., 2022)	100	9.2	31	21	18	9.4	32	23	19
LLaMA2-7B	100	9.6	29	21	18	9.1	30	19	17
LangNav (with LLaMA2-7B)	100	**7.4**	**40**	**32**	**28**	**7.1**	**45**	**34**	**29**

Table 1: Results on the R2R dataset with 10 or 100 real world trajectories. LangNav finetunes LLaMA2-7B on the mixture of the real-world trajectories and 10,000 synthetic trajectories from GPT-4.

# synthetic data	Data-generating LM	# seed scenes	NE↓	OSR↑	SR↑	SPL↑
2,000	GPT-3.5	10	9.8	31.0	15.6	12.2
2,000	GPT-4-turbo	1000	8.1	42.9	24.9	19.6
500	GPT-4	10	8.0	38.2	24.5	20.6
2,000	GPT-4	10	7.0	42.2	31.1	26.6
10,000	GPT-4	10	7.0	41.9	31.6	27.5
2,000 + 2,000	GPT-4 + GPT-4-turbo	10 + 1000	7.1	43.2	32.6	28.3

Table 2: Performance on the R2R val unseen set as we vary the number of synthetically generated data, the underlying LM from which the synthetic data is generated, and number of seed scenes. Here the seed scenes refer to the scans from which trajectories are sampled, with multiple trajectories originating from each seed scene.

language-only baseline that does not make use of the synthetic data from GPT-4.

All finetuning methods use the same set of 10/100 trajectories. For these experiments, we did not find significant differences in performance when using the object detection module, and hence we only relied on the image captioning system to give the language description of each view angle in the prompt template. See appendix A for the training setup including hyperparameters.

Results. The results are shown in table 1. We find that our GPT-4 zero- and few-shot results underperform the NavGPT baseline despite using the same backbone model, potentially due to NavGPT's use of ground truth distance information and chain-of-thought prompting (Wei et al., 2022; Kojima et al., 2023). Just finetuning LLaMA2-7B on the 10/100 gold trajectories does not perform well, although it is comparable to the vision-based policies. Training on a mixture of synthetic and R2R trajectories improves performance by a nontrivial margin, and the LLaMA2-7B-based LangNav approaches the performance of NavGPT despite being many times smaller, indicating the effectiveness of our pipelined prompting strategy for distilling the rich navigation-relevant world knowledge within GPT-4 to a smaller (and more efficient) language model.[6]

Ablation study. In table 2 we vary both the number of synthetic trajectories and the data-generating LM. Switching the synthetic data source from GPT-4 to GPT-3.5/GPT-4-turbo results in noticeable declines, highlighting the importance of using a strong LM. Increasing the number of synthetic trajectories increases performance, although the gains are marginal when going from 2,000 to 10,000 trajectories. This is potentially due to the use of only

---

[6] While we still underperform NavGPT, the performance gap is relatively narrow—within 1% in terms of SPL. We observe that NavGPT employs object information filtered by a ground-truth depth map, limiting the data to objects within a 3-meter range. Such filtering is important to mitigate the redundancy and noise often associated with unfiltered object information (i.e., often too many irrelevant objects are detected). As highlighted in the NavGPT paper, this selective use of object information is important for achieving good performance.

Methods	Pretraining Data	R2R data	Val Seen				Val Unseen			
			NE↓	OSR↑	SR↑	SPL↑	NE↓	OSR↑	SR↑	SPL↑
RecBert	R2R	10	10.8	9	7	6	10.1	13	9	9
		100	9.3	27	20	19	9.4	26	19	17
	ALFRED	0	9.5	12	8	4	9.0	12	7	3
		10	10.8	11	7	6	10.7	13	9	7
		100	9.9	22	18	17	10.2	23	15	14
LangNav	None	10	10.3	17	10	8	9.8	20	11	8
		100	9.0	25	20	18	9.2	25	17	15
	ALFRED	0	9.2	20	17	15	8.9	24	18	16
		10	8.7	20	19	18	8.3	21	18	17
		100	8.1	29	25	24	8.0	29	24	22

Table 3: Domain transfer results where we pretrain a navigation agent on the simulated ALFRED environment (which uses rendered images) and finetune on the real-world R2R environment. We use LLaMA-7B (Touvron et al., 2023a) as our backbone model, and compare against the RecBert (Hong et al., 2021) baseline.

10 real trajectories from a single scene to prompt LLMs which results in lack of instruction diversity (see examples in appendix E). To investigate the influence of the scene diversity, we use 1,000 navigation instructions sampled from various R2R scenes to prompt GPT-4-turbo[7] to generate 2,000 additional synthetic trajectories. We can see that although the 2,000 trajectories generated by GPT-4-turbo are not of the same quality as those generated by GPT-4, scaling up using these trajectories outperforms the results from the 10,000-trajectory set.

4.2 Case Study 2: Language as a Bridge for Domain Transfer

We next experiment with using language as a domain-invariant representation space to transfer a policy that has been trained on a different (rendered) environment (ALFRED; Shridhar et al., 2020), to the real-world R2R environment. There are significant differences between ALFRED and R2R which makes straightforward domain transfer challenging. ALFRED uses images rendered from the synthetic AI2THOR environment (Kolve et al., 2017), while R2R, based on the Matterport3D, incorporates images captured from real indoor environments. ALFRED's navigation trajectories and instructions are also simpler and shorter compared to R2R's instructions: R2R instructions involve guiding the agent between rooms, whereas ALFRED trajectories mainly keep the agent within a single room and provides instructions for household tasks. Finally in ALFRED, the agent is limited to rotating left/right by 90° and moving forward, while in R2R, the agent can move in any combination of 12 candidate heading directions and 3 elevation directions. See appendix B for detailed discussion of these differences, and see appendix A for the experimental setup.

Results. We pretrain both RecBert (Hong et al., 2021)[8] and LangNav on the simulated ALFRED environment and finetune on 0/10/100 R2R trajectories with object information. LangNav uses LLaMA1-7b (Touvron et al., 2023a) as the language model. The evaluation results for both methods are presented in table 3. Interestingly, for RecBert, pretraining on ALFRED actually *hurts* performance, potentially due to the model's overfitting to the idiosyncracies of the rendered environment. And without any R2R data, RecBert performs at near chance, whereas LangNav is able to exhibit some level of zero-shot transfer. Pretraining in ALFRED consistently leads to performance improvements for LangNav.

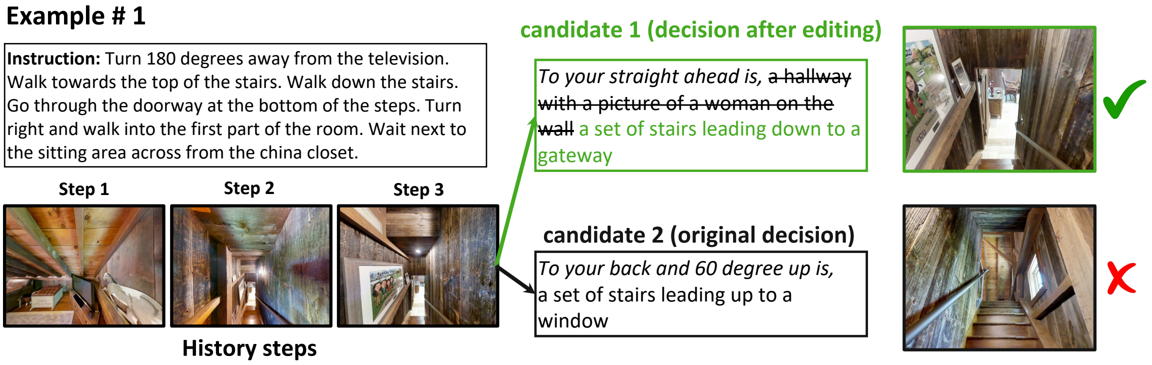
4.3 Case Study 3: Combining Language and Vision Representations

Our final case study explores whether language-based perceptual representations can improve performance *on top of* traditional continuous vision features. This is motivated by the observation that (1) in the full data setting, LangNav still underperforms the state-of-the-art approaches which rely on pure vision features (see table 5 of appendix C),

---

[7] We chose GPT-4-turbo for its lower cost.

[8] Given that RecBert (Hong et al., 2021) has similar performance to DuET (Chen et al., 2022) in the few-shot setting according to table 1, we choose RecBert to be the baseline because it is simpler and does not require a topological map.


Example # 1
Step 1
Step 2
Step 3
History steps
candidate 1 (decision after editing)
candidate 2 (original decision)

Instruction: Turn 180 degrees away from the television. Walk towards the top of the stairs. Walk down the stairs. Go through the doorway at the bottom of the steps. Turn right and walk into the first part of the room. Wait next to the sitting area across from the china closet.

To your straight ahead is, ~~a hallway with a picture of a woman on the wall~~ a set of stairs leading down to a gateway

To your back and 60 degree up is, a set of stairs leading up to a window

Figure 4: Interpreting and editing a model's predictions through language. At the beginning, the agent incorrectly selected "candidate 2" to ascend the stairs. The failure might stem from the ambiguous interpretation of mistaking the stairs for a hallway in "candidate 1". After editing the description (marked in green), the agent correctly alters its choice to walk down the stairs.

# Training	Perceptual features	SR↑	SPL↑
100	Vision only	19.0	17.4
100	Vision + language	19.3	18.0
Full train	Vision only	47.1	43.4
Full train	Vision + language	48.8	44.1

Table 4: Results when combining continuous visual features with language features with RecBert. Evaluations are conducted on R2R val unseen set.

and (2) realistic VLN scenarios would likely have access to continuous vision features as well.

We extend the RecBert (Hong et al., 2021) by concatenating language features to the visual features to represent the candidate image view. Concretely, the original RecBert uses ResNet-152 (He et al., 2016) to extract the visual feature to represent each view; our extension simply concatenates the caption representations (from BERT-base (Devlin et al., 2019)) to the image representation for each view. We train this new model on both the 100-shot and the full training set case.

Results. The results are listed in table 4. We find that language features improve the performance in both 100-shot and full training set cases, which indicates that language as a perceptual representation can provide additional benefits on top of continuous visual features, even in non-low-data settings. This is potentially due to language serving as useful prior for aspects of images that are salient for navigation.

5 Discussion

Interpretability and editability through language. Our use of language as a "bottleneck" perceptual representation makes it possible to (more easily) *interpret* and *edit* a model's predictions. As a qualitative case study, we inspect trajectories where the model made a mistake and manually inspect the captions. We find that model mistakes are generally due to incorrect or ambiguous captions. We manually edit the captions to be correct, and find that in many cases, this is able to change the model's predictions to be correct. See Figure 4 for a concrete example. We applied this procedure to 10 randomly selected trajectories which contained an error, and found that we were able to edit the model's decision to the correct one in 7 out of 10 trajectories. (For the other 3 trajectories, the failure was not due to incorrect captions).

Disentangling vision and language models. One the one hand, LangNav's use of a vision pipeline might seem like a step back from pure deep learning-based approaches which generally favor learning everything "end-to-end". On the other, the disentangling of the image module from the language module means our approach can readily make use of independent advances in vision and language models. This might become especially important given the recent trend in only providing API access to state-of-the-art language models.

Non-standard navigation environments. Our main experiments are on the R2R benchmark, which is realistic insofar as it makes use of real household environments. Another testbed for LangNav would be environments that lack existing datasets, such as offices or supermarkets. While the lack of existing benchmarks precludes our testing of LangNav on such non-standard environments, we performed a preliminary study where we tried generating synthetic trajectories from an office environment. We show an example in appendix J, where we find that GPT-4 is able to generate synthetic trajectories that contain common object-scene correlations in office environments and moreover exhibit great spatial consistency. Testing language as a perceptual representation in a variety of environments remains an interesting avenue for future work.

6 Related Work

Language Models for Task Planning. Several studies have explored language-based planning (Jansen, 2020; Sharma et al., 2021; Li et al., 2022b; Huang et al., 2022a; Ahn et al., 2022; Huang et al., 2022b). Huang et al. (2022a) use GPT-3 (Brown et al., 2020) and Codex (Chen et al., 2021a) for action plan generation with semantic translation using Sentence-RoBERTa (Huang et al., 2022a). SayCan (Ahn et al., 2022) grounds actions using FLAN (Wei et al., 2021) and action value functions (Shah et al., 2021). Huang et al. (2022b) explore incorporating grounded feedback into LLMs, while Xiang et al. (2023) propose enhancing LLMs' with embodied task instructions.

Instruction Tuning. There has been much recent work finetuning smaller language models such as LLaMA on synthetic instruction-following data generated by GPT-3.5/GPT-4 (Peng et al., 2023; Taori et al., 2023; Chiang et al., 2023; Wu et al., 2023). Existing works have generally focused on traditional language tasks. Our work instead finetunes LMs for embodied navigation tasks using language descriptions.

Vision-and-Language Navigation. There has been much work on vision and language navigation on the R2R dataset (Anderson et al., 2018a). Approaches such as the speaker-follower model (Fried et al., 2018) and environmental dropout method (Tan et al., 2019), reinforced cross-modal matching (Wang et al., 2019), and self-monitoring (Ma et al., 2019) have been proposed. Recent advancements include VLBERT-based methods (Hong et al., 2021) and object-informed sequential BERT (Qi et al., 2021). Qiao et al. (2022) incorporate additional pretext tasks into VLN pre-training based on Hong et al. (2021). ALFRED (Shridhar et al., 2020) involves interactive actions in a synthetic environment (Kolve et al., 2017), with methods utilizing dense single vector representations (Shridhar et al., 2020; Singh et al., 2021; Pashevich et al., 2021; Kim et al., 2021; Blukis et al., 2022) or a panoramic view space (Suglia et al., 2021). CLIP-Nav (Dorbala et al., 2022) explores the zero-shot VLN with CLIP while Kurita and Cho (2020) proposes a generative language model-based navigation approach. For instruction synthesis, Nguyen and Daumé III (2019) and Thomason et al. (2020) studies rule-based instruction synthesis in Matterport3D environment. Finally, our work is closely related to Zhou et al. (2023a) and Schumann et al. (2023), which also use language descriptions of an agent's perceptual representation to perform navigation with an LM.

7 Conclusion

We show that we can learn to navigate in a real-world environments by using language as a perceptual representation. Language naturally abstracts away low-level perceptual details, which we find to be beneficial for efficient data generation and sim-to-real transfer. However, this is also a serious limitation insofar as a picture really is worth a "thousand words" in some cases; we are certainly not suggesting the abandonment of traditional (continuous) vision features for vision-language navigation. But our case studies nonetheless demonstrate the promise of language as a perceptual representation for vision-language navigation.

Limitations

While we find that LangNav is promising in settings where only a handful of real trajectories are available, on the full dataset it still underperforms vision-based agents by a nontrivial margin, as shown in table 5 of appendix C. This is especially true when compared to state-of-the-art approaches such as ScaleVLN (Wang et al., 2023) which make use of large-scale pretraining data as well as more involved imitation/reinforcement learning algorithms that require access to an environment oracle. However, we note that while LangNav underperforms baselines in data-rich regimes, it overfits less to scenes seen during training, as demonstrated by the smaller drop in performance when applying the policy to unseen scenes during training.

Acknowledgements

This work is supported by MIT-IBM Watson AI Lab.

References

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

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

Peter Anderson, Ayush Shrivastava, Joanne Truong, Arjun Majumdar, Devi Parikh, Dhruv Batra, and Stefan Lee. 2021. Sim-to-real transfer for vision-and-language navigation. In *Conference on Robot Learning*, pages 671–681. PMLR.

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

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. 2022. A persistent spatial semantic representation for high-level natural language instruction execution. In *Conference on Robot Learning*, pages 706–717. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021a. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

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

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

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Zihao Wu, Lin Zhao, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, et al. 2023. Chataug: Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv:2302.13007*.

Terrence William Deacon. 1997. *The symbolic species: The co-evolution of language and the brain*. 202. WW Norton & Company.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics.

Vishnu Sashank Dorbala, Gunnar Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S Sukhatme. 2022. Clip-nav: Using clip for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2211.16649*.

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. *Advances in Neural Information Processing Systems*, 31.

Arnav Gudibande, Eric Wallace, Charlie Snell, Xinyang Geng, Hao Liu, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. The false promise of imitating proprietary llms. *arXiv preprint arXiv:2305.15717*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

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. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

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

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2022b. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Peter A Jansen. 2020. Visually-grounded planning without vision: Language models infer detailed plans from high-level instructions. *arXiv preprint arXiv:2009.14259*.

Byeonghwi Kim, Suvaansh Bhambri, Kunal Pratap Singh, Roozbeh Mottaghi, and Jonghyun Choi. 2021. Agent with the big picture: Perceiving surroundings for interactive instruction following. In *Embodied AI Workshop CVPR*, volume 2, page 7.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. 2017. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv*.

Shuhei Kurita and Kyunghyun Cho. 2020. Generative language-grounded policy in vision-and-language navigation with bayes' rule. *arXiv preprint arXiv:2009.07783*.

Hunter Lang, Monica N Agrawal, Yoon Kim, and David Sontag. 2022. Co-training improves prompt-based learning for large language models. In *International Conference on Machine Learning*, pages 11985–12003. PMLR.

Belinda Z Li, William Chen, Pratyusha Sharma, and Jacob Andreas. 2023. Lampp: Language models as probabilistic priors for perception and action. *arXiv e-prints*, pages arXiv–2302.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022a. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*.

Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. 2022b. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*.

Khanh Nguyen and Hal Daumé III. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. *arXiv preprint arXiv:1909.01871*.

OpenAI. 2023. Gpt-4 technical report.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Yuankai Qi, Zizheng Pan, Yicong Hong, Ming-Hsuan Yang, Anton Van Den Hengel, and Qi Wu. 2021. The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1655–1664.

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

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

Raphael Schumann, Wanrong Zhu, Weixi Feng, Tsu-Jui Fu, Stefan Riezler, and William Yang Wang. 2023. Velma: Verbalization embodiment of llm agents for vision and language navigation in street view. *arXiv preprint arXiv:2307.06082*.

Dhruv Shah, Peng Xu, Yao Lu, Ted Xiao, Alexander Toshev, Sergey Levine, and Brian Ichter. 2021. Value function spaces: Skill-centric state abstractions for long-horizon reasoning. *arXiv preprint arXiv:2111.03189*.

Pratyusha Sharma, Antonio Torralba, and Jacob Andreas. 2021. Skill induction and planning with latent language. *arXiv preprint arXiv:2110.01517*.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2022. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*.

Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. 2021. Factorizing perception and policy for interactive instruction following. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1888–1897.

Alessandro Suglia, Qiaozi Gao, Jesse Thomason, Govind Thattai, and Gaurav Sukhatme. 2021. Embodied bert: A transformer model for embodied, language-guided visual task completion. *arXiv preprint arXiv:2108.04927*.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR.

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. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? gpt-3 can help. *arXiv preprint arXiv:2108.13487*.

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

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. *arXiv preprint arXiv:2307.15644*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Proceedings of NeurIPS*.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2023. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *CoRR*, abs/2304.14402.

Jiannan Xiang, Tianhua Tao, Yi Gu, Tianmin Shu, Zirui Wang, Zichao Yang, and Zhiting Hu. 2023. Language models meet world models: Embodied experiences enhance language models.

Zirui Zhao, Wee Sun Lee, and David Hsu. 2023. Large language models as commonsense knowledge for large-scale task planning. *arXiv preprint arXiv:2305.14078*.

Gengze Zhou, Yicong Hong, and Qi Wu. 2023a. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. *arXiv preprint arXiv:2305.16986*.

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. 2023b. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. *arXiv preprint arXiv:2301.13166*.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.

A Implementations Details

We used the LLaMA-7B model (Touvron et al., 2023a) and the LLaMA2-7B model (Touvron et al., 2023b) for our method, fine-tuning it on 72 V100-32GB GPUs with a batch size of 144. The training tokens had a maximum length of 1024, while during inference, the maximum length was set to 2048. The AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 2×10^{-5} and weight decay of 0 was employed for optimization. The WarmupDecayLR learning rate scheduler was used for learning rate scheduling. For image captioning in both the R2R and ALFRED tasks, BLIP (Li et al., 2022a) was utilized. Deformable DETR (Zhu et al., 2020) was used for object detection in the R2R dataset, with suppression of outdoor object categories. We used the ground-truth object detection results provided in ALFRED when we generated the instruction-following pairs in § 4.2. When prompting GPT-4 / GPT-4-turbo / GPT-3.5 API, we set the temperature as 1 and top_p as 1. The cost of collecting the generated 10,000 trajectories by prompting GPT-4 API (OpenAI, 2023) was around $500. In the few-shot learning experiments in § 4.1 and § 4.2, we set $\rho = 0$. While when fine-tuning with the full train set in appendix D, we set $\rho = 0.2$. We pretrain on 128K ALFRED instruction-following pairs whose format is given in § 3.2. We augment the observations in ALFRED to 12 views and randomly mask a variable number of views to mimic the irregular number of candidates in R2R. The RecBERT baselines in table 1, table 3, and table 4 are pre-trained on 10/100 trajectories from R2R with masked language modeling (MLM) and single action prediction (SAP) tasks (Hao et al., 2020). The DUET baselines in table 1 are pre-trained on 10/100 trajectories with MLM, SAP, and masked region classification (MRC) tasks (Chen et al., 2022).

B Differences between ALFRED and R2R.

The primary cause of the vast difference between ALFRED and R2R lies in their environmental rendering: ALFRED utilizes images from the synthetic AI2THOR environment (Kolve et al., 2017), whereas R2R, drawing from the Matterport3D database, features images from actual indoor environments. We summarize the differences in the following aspects:

Visual appearance. ALFRED uses images rendered from the synthetic AI2THOR environment, while R2R, based on the Matterport3D, incorporates images captured from real indoor environments. These image sources differ in texture, occlusion, illumination, and other visual aspects.

Step size. There is a difference in step sizes between the two tasks (see the right part of fig. 5). ALFRED uses a step size of 0.25 meters, while R2R has larger and more variable step sizes. To bridge this gap, we consolidate four consecutive `MoveAhead` steps into a single step along the ALFRED trajectory.

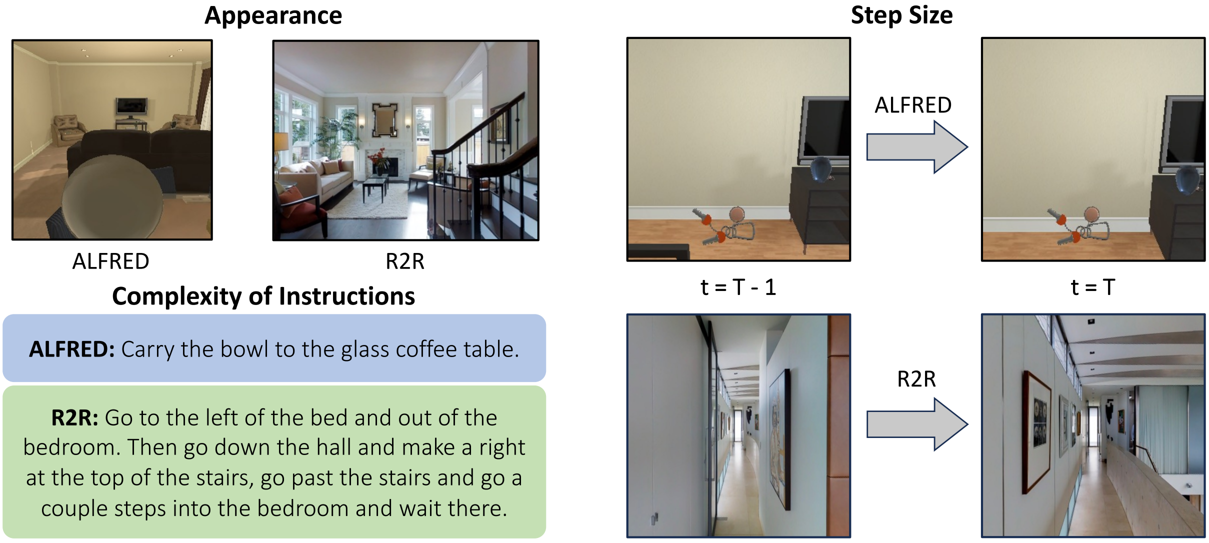
Action type. A complete ALFRED trajectory includes not only navigation actions but also interaction actions, where the interaction actions are combined with a target object to change the state of the surrounding environment. In order to filter the interaction actions in ALFRED, we divide each ALFRED trajectory into multiple sub-trajectories and keep the sub-trajectories that are labeled with the `GotoLocation` tag.

Instruction complexity. Due to trajectory splitting, ALFRED's navigation trajectories and instructions appear simpler and shorter compared to R2R's instructions. R2R instructions involve guiding the agent between rooms, whereas ALFRED trajectories mainly keep the agent within a single room.

Action space. In ALFRED, the agent is limited to rotating left/right by 90° and moving forward, while in R2R, the agent can move in any combination of 12 candidate heading directions and 3 elevation directions. The number of available movement directions is irregular. This difference in action space makes R2R trajectories more human-like. To address this, we introduce randomness by adding or reducing a heading offset of $\pm 30°$ to the agent's direction at each step in ALFRED, allowing rotations of 30° or 60° in addition to 90°.

C Performance on full data

In Table 5 we show the performance of LangNav on the full dataset, as well as comparisons against the state-of-the-art. While we find that LangNav is promising in settings where only a handful of real trajectories are available, on the full dataset it still underperforms vision-based agents by a nontrivial margin. This is especially true when compared to state-of-the-art approaches such as ScaleVLN (Wang et al., 2023) which make use of large-scale pretraining data as well as more involved imitation/reinforcement learning algorithms that require



ALFRED: Carry the bowl to the glass coffee table.

R2R: Go to the left of the bed and out of the bedroom. Then go down the hall and make a right at the top of the stairs, go past the stairs and go a couple steps into the bedroom and wait there.

Figure 5: Task gap between ALFRED and R2R. We highlight notable distinctions between the navigation tasks in ALFRED and R2R, encompassing variations in appearance, step size, and instruction complexity. See appendix B for more details.

access to an environment oracle during training. However, we note that while LangNav underperforms baselines in data-rich regimes, it overfits less to scenes seen during training, as demonstrated by the smaller drop in performance when applying the policy to unseen scenes during training.

D Multi-Task Performance

One of the advantages of our approach is its inherent suitability for multitasking. Similar to LLMs use instruction to handle multiple language tasks concurrently, we consolidate task information and inputs into instructions. To validate the multitasking capability of our method, we extend its application to the ALFRED task.

Metrics on ALFRED. We evaluate our model on ALFRED using two metrics: *Task Success* (`Task`) and *Goal-Condition Success* (`GC`). `Task Success` measures the ratio of trajectories where object positions and state changes accurately match all task goal conditions at the end. `GC` assesses the ratio of completed goal conditions in each action sequence. `Task Success` is only considered successful when `GC` is also 1. On average, each ALFRED task has 2.55 goal conditions. We also calculate the *Path Length Weighted Metrics* (`PLW`) for both `Task` and `GC`, which normalize the metrics based on the actual action sequence length.

Results of the Multi-Task Model. In ALFRED task, we set $\rho = 0$ as the expert policy in ALFRED is suboptimal. To save training time and balance the data amount between R2R and ALFRED, we utilize only 50% of the training dataset, resulting in a dataset for ALFRED with 386K data pairs. For R2R task training, we maintain $\rho = 0.2$ and run each demonstration trajectory twice, resulting in a training set size of 235K for R2R. Consequently, the merged dataset for the multitask model contains a total of 621K instruction-following data pairs. We select VLN Bert (Hong et al., 2021) as the baseline for the R2R task and Seq2seq model (Shridhar et al., 2020) for the ALFRED task. Given the substantial differences between the R2R task and the ALFRED task (§ 4.2), our method is, to the best of our knowledge, the first model that simultaneously addresses these two tasks. In table 6 and table 7, we find that the multitask model exhibits superior performance compared to the single-task models. These results underscore the capability of our method to effectively handle multiple highly diverse tasks.

E Bias of generated navigation instructions

We list four generated instructions from one output of GPT-4 with 10 seed trajectories as we mentioned in § 4.1 as bellow:

Examples of generated instructions

```
1. Start from the main entrance door, pass
the living room, and enter the kitchen on
your right. Locate the refrigerator, then
turn left and stop just before the dining
```

Method	**Training data**	**Needs Oracle**	**Val Seen**	**Val Unseen**	**Drop**
Seq2Seq (SF) (Anderson et al., 2018b)	R2R	No	38.6	21.8	16.8
RCM (Wang et al., 2019)	R2R	Yes	67.4	42.5	24.9
Speaker-Follower (Fried et al., 2018)	R2R+SpeakerAug.	Yes	70.1	54.6	15.5
RecBert† (Hong et al., 2021)	R2R+PREV	Yes	71.8	54.5	17.3
HAMT (Chen et al., 2021b)	R2R+PREV	Yes	75.0	65.7	9.3
ScaleVLN (Wang et al., 2023)	R2R+PREV	No	67.2	47.4	19.8
ScaleVLN (Wang et al., 2023)	R2R+PREV	Yes	76.9	72.9	4.0
ScaleVLN (Wang et al., 2023)	R2R+PREV+ScaleVLN	No	71.1	57.0	14.1
ScaleVLN (Wang et al., 2023)	R2R+PREV+ScaleVLN	Yes	80.5	78.1	2.4
LangNav	R2R	No	55.0	43.2	11.8
LangNav (M)	R2R+ALFRED	No	55.9	45.6	10.3

Table 5: Comparison with state-of-the-art vision-based methods on the R2R dataset when trained on the full dataset. We use success rate (SR) as the performance metric. “Needs oracle” indicates that the model needs to rely on an oracle during training that can give the ground-truth next action based on a sampled path from the model.(M): Multi-Task model.

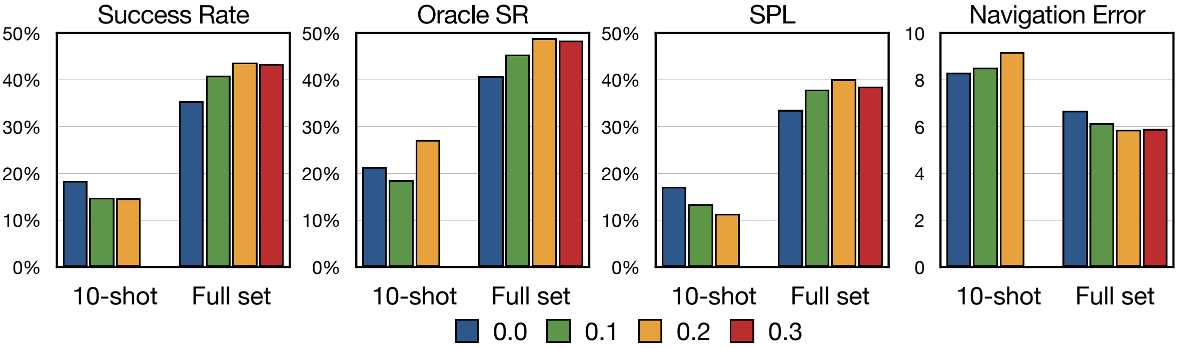


Figure 6: Investigating the Impact of the Randomness Factor ρ on Model Performance. This image caption depicts an ablation study exploring the influence of the randomness factor ρ on our model’s performance in both few-shot learning and full-set training scenarios. We test ρ with values of 0.0, 0.1, 0.2, and 0.3.

Table 6: Performance of the Multi-task Model on R2R. We demonstrate the multi-task capability of the LM agent. For single-task models, each model is trained within the task data. We trained the multi-task model with data from both R2R and ALFRED tasks.

Models	**R2R Seen**		**R2R Unseen**	
	SR↑	SPL↑	SR↑	SPL↑
Single-Task	55.0	51.0	43.2	37.9
Multi-Task	**55.9**	**51.7**	**45.6**	**40.0**

Table 7: Performance of the Multi-task Model on ALFRED. ST: Single-Task. MT: Multi-Task.

	ALFRED Seen		**ALFRED Unseen**	
	Task↑	GC↑	Task↑	GC↑
ST	0.0 (0.0)	6.0 (4.7)	0.5 (0.1)	9.5(7.8)
MT	0.0 (0.0)	**6.4 (5.0)**	**0.6 (0.2)**	**9.8 (7.8)**

```
table.
```

```
2. Navigate from the couch in the living
room, move towards the mantel, and then
stop next to the fireplace. Avoid any
furniture and obstacles on your path.

3. Begin at the foot of the bed in the
master bedroom. Walk forward and enter the
attached bathroom. Once you’re inside,
stop next to the bathtub.

4. Start in the family room, walk towards
the TV, then turn right and pass the
bookshelf. Stop when you reach the large
bay window overlooking the garden.
```

We can see from the above synthetic instructions that (a) patterns of the synthetic instructions are similar, which are like "Start from place A, go pass place B, stop at place C", (b) scenes are limited to the living area and a single floor, however, the R2R tasks always require the agent navigating across floors and in some non-living area.

F Impact of the randomness factor

We conduct the ablation study to investigate the impact of the randomness factor ρ on the model's performance in both few-shot learning and full-set fine-tuning scenarios. Interestingly, we observe different behaviors of the model with varying ρ values in these scenarios. Fig. 6 illustrates our findings. In the 10-shot scenario, increasing ρ negatively affected the model's performance. However, in the full-set scenario, the model achieved optimal performance at $\rho = 0.2$.

G Example of the Full Trajectory

We plot an example of a full text-based trajectory in R2R as we mentioned in § 3.2 as bellow:

Full R2R trajectory

Task description:

You are a navigation agent who must navigate according to instructions given only descriptions of your current position via natural language. The natural language description is sometimes incorrect.

Navigation instruction:

Go across the kitchen passed the pizza oven into the next room. Stop just inside the bedroom.

Navigation trajectory:

Step 1:

(🍕 **environment:**)
```
To your 60 degree left is,
a kitchen with a stove, sink, and
refrigerator
Details: oven, bowl, dining table,
microwave, refrigerator
To your 30 degree left is,
a kitchen with white cabinets and a
white stove top oven
Details: oven, bowl, microwave,
potted plant, chair
To your 60 degree right is,
a living room filled with furniture
and a fire place
Details: couch, chair
To your back is,
a room with a bed and a painting on
the wall
```

You chose:

(🤖 **navigation agent:**)
```
a living room filled with furniture
and a fire place
```

Step 2:

(🍕 **environment:**)
```
To your 150 degree left is,
a kitchen with a stove top oven
sitting next to a counter
Details: oven, dining table,
microwave, refrigerator, chair
To your back is,
a kitchen with a wooden table and
white appliances
Details: oven, bowl, dining table,
microwave, refrigerator, apple,
couch, chair
To your 30 degree right is,
a living room with a couch a chair
and a window
Details: couch
To your 120 degree right is,
a dining room table with a bowl of
fruit on it
Details: chair, bowl, dining table
To your 150 degree right is,
a bowl of fruit sits on a wooden
table
Details: couch, chair, bowl, dining
table
```

You chose:

(🤖 **navigation agent:**)
```
a living room with a couch a chair
and a window
```

Step 3:

(🍕 **environment:**)
```
To your back is,
a kitchen with a table, chairs, and
stairs
Details: oven, dining table,
refrigerator, potted plant, chair
To your 150 degree right is,
a room with a table, chairs, and
stairs
Details: oven, chair, refrigerator,
dining table
To your straight ahead and 30 degree
down is,
a view of a hallway from the top of
stairs
Details: refrigerator
To your 90 degree right and 30 degree
up is,
a room with a staircase and a mirror
on the wall
Details: toilet
```

You chose:

(🤖 **navigation agent:**)
```
a view of a hallway from the top of
stairs
```

Step 4:

(🍕 **environment:**)
```
To your back and 60 degree up is,
a living room filled with furniture
and a ceiling fan
Details: oven, potted plant,
refrigerator
```

```
To your 90 degree left and 30 degree
up is,
a living room with a couch and a
table
To your straight ahead and 30 degree
up is,
a bedroom with a bed and a mirror on
the wall
Details: bed
```

You chose:

(🤖 **navigation agent:**)

```
a bedroom with a bed and a mirror on
the wall
```

Step 5:

(🗺 **environment:**)

```
To your back is,
a hallway leading to a kitchen and
living room
Details: refrigerator, potted plant
To your 30 degree left is,
a room with a wooden door and a
mirror
To your straight ahead is,
a bedroom with a bed, dresser, mirror
and a ceiling fan
Details: potted plant, bed
To your 30 degree right is,
a bedroom with a bed and a ceiling
fan
Details: potted plant, bed
To your 60 degree right is,
a bedroom with a bed, dresser and
mirror
Details: potted plant, bed
```

You chose:

(🤖 **navigation agent:**)

```
stop
```

H Complete Prompt Template of Generating Trajectories for GPT-4

We list our complete templates for prompting GPT-4 to generate synthetic instructions (Phase I) and synthetic trajectories to fulfill the instruction (Phase II).

Phase I: Prompt template for generating synthetic instructions

System prompt:
I am going to give you example instructions written by humans to train a deep learning-based navigation agent acting inside a home. These example instructions are intended to be completed by the navigation agent in 5-7 steps.

Few shot examples:

```
- {real_instruction_1}
- {real_instruction_2}
- {real_instruction_3}
```

🧑 **User:**

Your goal is to write 10 more instructions like the above that can be used to train a navigation agent. Since the navigation agent will be navigating in different home environments, your instructions should also be diverse and cover a wide range of home environments and rooms. You should make sure that the instruction can be completed by an agent in 5 to 7 steps.

Phase II: Prompt template for generating synthetic trajectories

System prompt:
Here is an example of a large language model acting as a blind navigation agent in an indoor environment through text descriptions. The agent is given an instruction at the start and must follow the instruction. At each time step, the agent is given descriptions of its field of view via the following template:

```
To your [VIEW] is [CAPTION]
- [VIEW] consists of the agent’s visible
field of view (e.g., 30 degrees right, 120
degrees left, etc.)
- [CAPTION] is the text description of
that view obtained from an image
captioning model
```

Few shot examples:

# Example 1
`### Instruction:`
`{real_instruction_example}`
`### Trajectory:`
`{real_trajectory_example}`

🧑 **User:**

Now I will give you another instruction. Please generate a trajectory of 5-7 steps that would complete the instruction.
# Example 2
`### Instruction:`
`{synthetic_instruction}`

I Prompts of Zero-shot and Few-shot Navigation for GPT-4

Here we attach the the task description D in the prompt template for prompting GPT-4 to navigate in the R2R evaluation dataset.

Zero-shot

System prompt:
You are a navigation agent who must navigate according to instructions given only descriptions of

your current position via natural language. The natural language description is sometimes incorrect.

🧑 **User:**

At each step, you will be given several directions and captions for each direction. You must choose one direction by printing only the [caption_of_the_direction] or choose "Stop" if you think the goal is reached.
For example:
Input:

```
To your [direction_1] is, [caption of the direction_1].
......
To your [direction_N] is, [caption of the direction_N].
You choose:
Output: [caption of the direction_3]
```

Hint: You should use the information inside the instructions, history steps, and current observations to make the decision.

Few-shot

System prompt:
You are a navigation agent who must navigate according to instructions given only descriptions of your current position via natural language. The natural language description is sometimes incorrect.

🧑 **User:**

At each step, you will be given several directions and captions for each direction. You must choose one direction by printing only the [caption_of_the_direction] or choose "Stop" if you think the goal is reached.
For example:
Input:

```
To your [direction_1] is, [caption of the direction_1].
......
To your [direction_N] is, [caption of the direction_N].
You choose:
Output: [caption of the direction_3]
```

Few shot examples:

And here is an example trajectory:

```
### Instruction:
Go down the stairs. Turn right and go down the hallway. Turn right and stand near the fireplace.
### Trajectory:
Step 1:
To your straight ahead is,
an ornate doorway leading to another room
To your 60 degree right is,
a red carpeted staircase leading to a chandelier
To your 120 degree right is,
a room with a red carpet and a large mirror
To your back and 30 degree down is,
a room with a red carpet and two windows
To your 120 degree left is,
a room with a red carpet and gold trim
You chose:
a room with a red carpet and gold trim
Step 2:
To your 150 degree right is,
a very ornate staircase in a house with red and white striped chairs
To your back is,
a red carpeted hallway leading to a staircase
To your 150 degree left is,
a hallway with a red carpet and a chandelier
To your 120 degree left is,
a room with a red carpet and a chandelier
To your 90 degree left is,
a room with a chandelier and two windows
To your 60 degree left is,
a room with a red carpet and a large mirror
To your 30 degree right is,
a hallway with a red carpet and wooden doors
You chose:
a hallway with a red carpet and wooden doors
Step 3:
To your back is,
a hallway with a red carpet and a chandelier
To your straight ahead is,
a hallway with a red carpet and a gold ceiling
a hallway with a red carpet and a gold ceiling
You chose:
a hallway with a red carpet and a gold ceiling
Step 4:
To your 90 degree right is,
a living room with a chandelier and a fireplace
To your 120 degree right is,
a room with a fireplace and a chandelier in it
To your back is,
a hallway with a red carpet and gold trim
To your 90 degree left is,
a room with a chandelier and a table in it
To your 30 degree right is,
a living room with a chandelier and a couch
You chose:
a living room with a chandelier and a fireplace
Step 5:
To your straight ahead is,
a living room filled with furniture and a fire place
To your 30 degree left is,
```

a living room filled with furniture and a chandelier
To your 150 degree right is,
a room with a chandelier and a table in it
To your back is,
a hallway with a chandelier and a mirror on the wall
To your 90 degree left is,
a living room filled with furniture and a chandelier
You chose:
Stop

🧑 **User:**

Hint: You should use the information inside the instructions, history steps, and current observations to make the decision.
Now let's start!

J Examples of Generated Trajectories in Exotic Environments

To validate that LLMs can synthesize useful data in more exotic environments, we conduct an experiment where we handcraft a trajectory in a real office environment and then prompt GPT-4 to generate synthetic trajectories within the scope of the office environment. Here we first plot the language trajectory we have sampled from a real office environment. And then, we attach the generated trajectories from GPT-4 to demonstrate the ability of a strong LLM to generate trajectories in more exotic environments.

Real-world trajectory

Navigation instruction:

Start from the pantry on your left, pass the board room to enter the atrium. Go through the door to get into the office area. Turn right at the first corner, and then turn right to stop at the printer.

Navigation trajectory:

Step 1:

(🗺 **environment:)**
To your straight ahead is,
a long hallway with light on the ceiling and a fridge on the left
To your 120 degree left is,
a lounge with a screen on the wall, floor-to-ceiling winders, desks, and many high chairs
To your 90 degree right is,
an auditorium with three big screens and tiered seating facing the stage

You chose:

(🤖 **navigation agent:)**
a long hallway with light on the ceiling and a fridge on the left

---

Step 2:

(🗺 **environment:)**
To your straight ahead is,
a long hallway with four posters on the left leading to the atrium
To your back is,
a lounge with a sofa and a shelf stocked with snacks

You chose:

(🤖 **navigation agent:)**
a long hallway with four posters on the left leading to the atrium

---

Step 3:

(🗺 **environment:)**
To your straight ahead is,
an atrium with lounge chairs on the right and a door on the straight ahead
To your 90 degree left is,
a front desk in the corner with a monitor on the desk.
To your 90 degree right is,
an exit gate towards a set of commercial elevators.
To your back is,
a long hallway with posters on the right wall

You chose:

(🤖 **navigation agent:)**
an atrium with lounge chairs on the right and a door on the straight ahead

---

Step 4:

(🗺 **environment:)**
To your straight ahead is,
An office area where there is a black board and a work pod room.
To your 90 degree left is,
An array of office desks with monitors and office chairs.
To your 90 degree right is,
A large office area with multiple arrays of desks and chairs.
To your back is,
An exit door with a phone room on the right.

You chose:

(🤖 **navigation agent:)**
A large office area with multiple arrays of desks and chairs

Step 5:

(🗺 **environment:)**

To you straight ahead is,
A large office area with multiple
arrays of desks and chairs, as well
as trash bins.
To your 60 degree left is,
A black board in the middle of an
office area.
To your 90 degree right is,
A copy/print room with recycle bins.
To your back is,
An array of office desks and chairs
with a black board.

You chose:

(**navigation agent:)**
A copy/print room with recycle bins

Step 6:

(**environment:)**
To your 90 degree right is,
A printer on the ground and several
instruction sheet on the wall
To your back is,
A black board and an array of office
desks

You chose:

(**navigation agent:)**
Stop

Synthetic trajectory #1

Navigation instruction:

Begin at the elevator lobby. Exit the lobby and turn left, heading towards the open office space. Walk past two rows of desks, then take a right. Continue straight until you see the coffee station. Turn left and stop at the second conference room.

Navigation trajectory:

Step 1:

(**environment:)**
To your straight ahead is,
an elevator lobby with multiple
elevators and a directory stand
To your 90 degree left is,
a closed door with a sign "Stairs"
To your 90 degree right is,
a wide corridor leading to a
reception area

You chose:

(**navigation agent:)**
a wide corridor leading to a
reception

Step 2:

(**environment:)**
To your straight ahead is,
a reception area with a white desk
and three chairs
To your 120 degree left is,
a corridor leading to an open office
space with several workstations
To your 90 degree right is,
a closed door marked "Emergency Exit"

You chose:

(**navigation agent:)**
a corridor leading to an open office
space with several workstations

Step 3:

(**environment:)**
To your straight ahead is,
an open office space filled with rows
of desks and office chairs
To your 90 degree left is,
an array of cubicles with computers.
To your 90 degree right is,
an exit door with a sign "Stairs"

You chose:

(**navigation agent:)**
an open office space filled with rows
of desks and office chairs

Step 4:

(**environment:)**
To your straight ahead is,
an open office space with rows of
desks and office chairs.
To your back is,
an office space with several
workstations.
To your 90 degree left is,
a hallway leading to the break room.
To your 90 degree right is,
a corner office space with a large
round table.

You chose:

(**navigation agent:)**
an office space with rows of desks
and office chairs.

Step 5:

(**environment:)**
To your straight ahead is,
an open office space with rows of
desks and office chairs.
To your 30 degree left is,
a meeting room with a large table and
multiple chairs.
To your 120 degree right is,
a corridor leading to a door.

You chose:

(**navigation agent:)**
a corridor leading to a door.

Step 6:

(**environment:)**
To your straight ahead is,
a door at the end of the hallway
To your 90 degree right is,
a coffee station with a variety of coffee machines

You chose:

(**navigation agent:)**
a coffee station with a variety of coffee machines

Step 7:

(**environment:)**
To your straight ahead is,
a small kitchenette with a fridge and microwave
To your immediate left is,
a conference room featuring a large glass table
To your second left is,
a second conference room with a project screen and chairs

You chose:

(**navigation agent:)**
a second conference room with a project screen and chairs

Step 8:

(**environment:)**
To your straight ahead is,
a conference room with a projection screen and chairs.
To your back is,
a coffee station with various coffee machines.

You chose:

(**navigation agent:)**
Stop

Synthetic trajectory #2

Navigation instruction:

Start outside the IT department’s office. Walk straight and take the first left towards the stairwell. Go up one flight of stairs. Upon reaching the top, turn right and proceed to the end of the corridor. The library is on the right.

Navigation trajectory:

Step 1:

(**environment:)**
To your straight ahead is,
the hallway leading to an intersection with doors for the office spaces on the sides
To your 90 degree left is,
a door with a sign “IT Department” followed by some tech carts
To your 90 degree right is,
a hallway towards the elevators with seating areas on the sides

You chose:

(**navigation agent:)**
the hallway leading to an intersection with doors for the office spaces on the sides

---

Step 2:

(**environment:)**
To your straight ahead is,
a hallway with the elevator and murals on its sides
To your 90 degree left is,
a hallway that moves towards a stairwell
To your back is,
a hallway leading towards the IT department’s office

You chose:

(**navigation agent:)**
a hallway that moves towards a stairwell

Step 3:

(**environment:)**
To your straight ahead is,
a stairwell leading upwards with railing and stair lights
To your 90 degree left is,
a hallway towards a coffee machine and cafeteria entrance
To your 90 degree right is,
a door to a rest room with signage
To your back is,
a hallway with murals leading back to the IT department’s office.

You chose:

(**navigation agent:)**
a stairwell leading upwards with railing and stair lights

Step 4:

(**environment:)**
To your straight-ahead is,
another flight of stairs leading to a large window
To your 90 degree right is,
a hallway lined with posters and a bench
To your back is,
a flight of stairs that leads down with railing and stair lights

You chose:

(**navigation agent:**)
a hallway lined with posters and a bench

Step 5:

(**environment:**)
To your straight-ahead is,
a corridor leading to an intersection with a water fountain to the right
To your 90 degree left is,
a large window that illuminates the corridor and gives a view of the street
To your 90 degree right is,
a library entrance with a signage
To your back is,
the hallway lined with posters and a bench

You chose:

(**navigation agent:**)
a library entrance with a signage

Step 6:

(**environment:**)
To your straight ahead is,
a library filled with wooden bookshelves and desks with desk lamps
To your back is,
a corridor leading back to the stairwell

You chose:

(**navigation agent:**)
Stop

K More Examples of Generated Trajectories

We plot more examples of the generated trajectories from GPT-4 below:

Example #1

Navigation instruction:

Begin in the basement, walk towards the pool table. Pass the pool table and ascend the staircase. At the top of the stairs, turn right and walk through the hallway. Turn left into the den, walk through and stop by the window.

Navigation trajectory:

Step 1:

(**environment:**)
To your straight ahead is,
a basement with a pool table, a bar, and a seating area
To your 90 degree left is,
a wall with various framed photos and a door
To your 150 degree right is,
a storage area filled with boxes and shelves

You chose:

(**navigation agent:**)
a basement with a pool table, a bar, and a seating area

Step 2:

(**environment:**)
To your back is,
a basement entrance with a staircase leading down
To your 120 degree left is,
a basement with a bar, television, and several chairs
To your straight ahead is,
a basement with a pool table, a dartboard, and a bar
To your 90 degree right is,
a staircase leading upwards with a wooden railing

You chose:

(**navigation agent:**)
a basement with a pool table, a dartboard, and a bar

Step 3:

(**environment:**)
To your back is,
a basement with a seating area and framed photos on the wall
To your straight ahead is,
a pool table surrounded by chairs and a dartboard on the wall
To your 150 degree left is,
a basement with a bar, television, and several chairs
To your 90 degree right is,
a staircase leading upwards with a wooden railing

You chose:

(**navigation agent:**)
a staircase leading upwards with a wooden railing

Step 4:

(**environment:**)
To your back is,
a basement with a pool table and a dartboard on the wall
To your 90 degree left is,
a hallway with various framed photos on the walls
To your straight ahead is,
a den with a large bookshelf, couch, and coffee table

You chose:

(**navigation agent:)**
a hallway with various framed photos on the walls

Step 5:

(**environment:)**
To your back is,
a staircase leading downwards into the basement
To your straight ahead is,
a long hallway with several doors leading to other rooms
To your 120 degree right is,
a den with a comfortable seating area and a bookshelf

You chose:

(**navigation agent:)**
a long hallway with several doors leading to other rooms

Step 6:

(**environment:)**
To your back is,
a hallway with various framed photos on the walls
To your 90 degree left is,
a den filled with cozy furniture, bookshelves, and a television
To your 90 degree right is,
a bedroom with a large bed, dresser, and a window

You chose:

(**navigation agent:)**
a den filled with cozy furniture, bookshelves, and a television

Step 7:

(**environment:)**
To your back is,
a hallway leading to other parts of the house
To your 90 degree left is,
a den with several bookshelves, a fireplace, and a large rug
To your straight ahead is,
a window overlooking the garden outside

You chose:

(**navigation agent:)**
Stop

Example #2

Navigation instruction:

From the garage entrance, walk into the kitchen. Turn left and walk past the stove. Continue straight into the family room, then walk towards the sliding glass doors. Step outside and stop near the barbecue grill.

Navigation trajectory:

Step 1:

(**environment:)**
To your straight ahead is,
a clean garage with a parked car and a workbench
To your 90 degree left is,
a laundry room with a washer and a dryer
To your back is,
a driveway leading up to a garage door
To your 120 degree left is,
a garage entrance with an open door leading inside the house

You chose:

(**navigation agent:)**
a garage entrance with an open door leading inside the house

Step 2:

(**environment:)**
To your 30 degree left is,
a kitchen with a stove, a refrigerator, and a microwave oven
To your 90 degree right is,
a small mudroom with a shoe rack and a coat hook
To your back is,
a view of a garage with a car and a workbench

You chose:

(**navigation agent:)**
a kitchen with a stove, a refrigerator, and a microwave oven

Step 3:

(**environment:)**
To your straight ahead is,
a large kitchen with a stove and a range hood
To your 30 degree right is,
a kitchen with a center island and a sink on the countertop
To your 150 degree left is,
a kitchen with a refrigerator and a microwave oven
To your 120 degree right is,
a tiled floor leading to a hallway with a picture on the wall

You chose:

(**navigation agent:)**
a large kitchen with a stove and a range hood

Step 4:

(**environment:)**
To your straight ahead is,
a cozy family room with a TV and a fireplace
To your 90 degree right is,
a kitchen with a sink and a dishwasher
To your back is,
a kitchen with a countertop, a cooktop, and an oven

You chose:

(**navigation agent:)**
a cozy family room with a TV and a fireplace

Step 5:

(**environment:)**
To your 30 degree left is,
a sliding glass door leading to the backyard
To your 90 degree right is,
a comfortable sofa with a coffee table in front of a TV
To your back is,
a large kitchen with a stove and a range hood

You chose:

(**navigation agent:)**
a sliding glass door leading to the backyard

Step 6:

(**environment:)**
To your straight ahead is,
a well-maintained backyard with a barbecue grill and a dining table
To your 90 degree left is,
a sliding glass door leading inside the family room
To your back is,
a cozy family room with a TV and a fireplace

You chose:

(**navigation agent:)**
a well-maintained backyard with a barbecue grill and a dining table

Step 7:

(**environment:)**
To your 90 degree left is,
a barbecue grill on a patio in the backyard
To your straight ahead is,
a backyard with a dining table and chairs
To your 120 degree right is,
a backyard with a garden and a swing set
To your back is,
a sliding glass door leading inside the family room

You chose:

(**navigation agent:)**
Stop

Example #3

Navigation instruction:

Stand in front of the fireplace in the living room. Walk diagonally across the room, passing the sofa. Proceed through the door into the hallway. Turn left and continue toward the guest room. Stop just before entering and wait.

Navigation trajectory:

Step 1:

(**environment:)**
To your straight ahead is,
a living room with a fireplace and two chairs
To your 30 degree left is,
a living room with a large window and a dark brown couch
To your 90 degree right is,
a living room with a flat screen tv and a view of the backyard

You chose:

(**navigation agent:)**
a living room with a fireplace and two chairs

Step 2:

(**environment:)**
To your 60 degree left is,
a living room with a fireplace and a floor lamp
To your 30 degree right is,
a living room with a dark brown couch and a coffee table
To your straight ahead is,
a living room with a white rug in the middle

You chose:

(**navigation agent:)**
a living room with a dark brown couch and a coffee table

Step 3:

(**environment:)**
To your back is,
a living room with a fireplace and a christmas tree

To your 150 degree left is,
a living room with two chairs and a painting on the wall
To your straight ahead is,
a wooden door leading to a hallway

You chose:

(🤖 **navigation agent:**)
a wooden door leading to a hallway

Step 4:

(🗺️ **environment:**)
To your 90 degree left is,
a hallway with a view of a staircase and a table
To your straight ahead is,
a hallway with a painting on the wall and an open door

You chose:

(🤖 **navigation agent:**)
a hallway with a painting on the wall and an open door

Step 5:

(🗺️ **environment:**)
To your back is,
a hallway with a wooden floor and a closed door
To your 120 degree left is,
a guest bedroom with a neatly made bed and a dresser
To your 30 degree right is,
a hallway with white walls and floor-to-ceiling mirrors

You chose:

(🤖 **navigation agent:**)
Stop just before entering the guest bedroom