End-to-End Navigation with Vision-Language Models: Transforming Spatial Reasoning into Question-Answering

Dylan Goetting DYLANGOETTING@BERKELEY.EDU

Himanshu Gaurav Singh HIMANSHU_SINGH@BERKELEY.EDU

Antonio Loquercio ALOQUE@SEAS.UPENN.EDU

Editors: G. Pappas, P. Ravikumar, S. A. Seshia

Abstract

We present VLMnav, an embodied framework to transform a Vision-Language Model (VLM) into an end-to-end navigation policy. In contrast to prior work, we do not rely on a separation between perception, planning, and control; instead, we use a VLM to directly select actions in one step. Surprisingly, we find that a VLM can be used as an end-to-end policy zero-shot, i.e., without any fine-tuning or exposure to navigation data. This makes our approach open-ended and generalizable to any downstream navigation task. We run an extensive study to evaluate the performance of our approach in comparison to baseline prompting methods. In addition, we perform a design analysis to understand the most impactful design decisions. Visual examples and code for our project can be found at jirl-upenn.github.io/VLMnav/.

Keywords: Navigation, VLM, Embodied AI, Exploration

1. Introduction

The ability to navigate effectively within an environment to achieve a goal is a hallmark of physical intelligence. Spatial memory, along with more advanced forms of spatial cognition, is believed to have begun evolving early in the history of land animals and advanced vertebrates, likely between 400 and 200 million years ago Muzio and Bingman (2022). Because this ability has evolved over such a long period, it feels almost instinctual and trivial to humans. However, navigation is, in reality, a highly complex problem. It requires the coordination of low-level planning to avoid obstacles alongside high-level reasoning to interpret the environment's semantics and explore the directions that are most likely to get the agent to achieve their goals.

A significant portion of the navigation problem appears to involve cognitive processes similar to those required for answering long-context image and video questions, an area where contemporary vision-language models (VLMs) excel OpenAI et al. (2024); Team et al. (2024). However, when naively applied to navigation tasks, these models face clear limitations. Specifically, when given a task description concatenated with an observation-action history, VLMs often struggle to produce fine-grained spatial outputs to avoid obstacles and fail to effectively utilize their long-context reasoning capabilities to support effective navigation Ramakrishnan et al. (2024); Nasiriany et al. (2024); Rahmanzadehgervi et al. (2024).

To address these challenges, previous work has included VLMs as a component of a modular system to perform high-level reasoning and recognition tasks. The systems generally contain an explicit 3D mapping module and a planner to deal with the more embodied part of the task, e.g., motion and exploration Kim et al. (2024); Majumdar et al. (2022); Gadre et al. (2023); Yu et al. (2023);

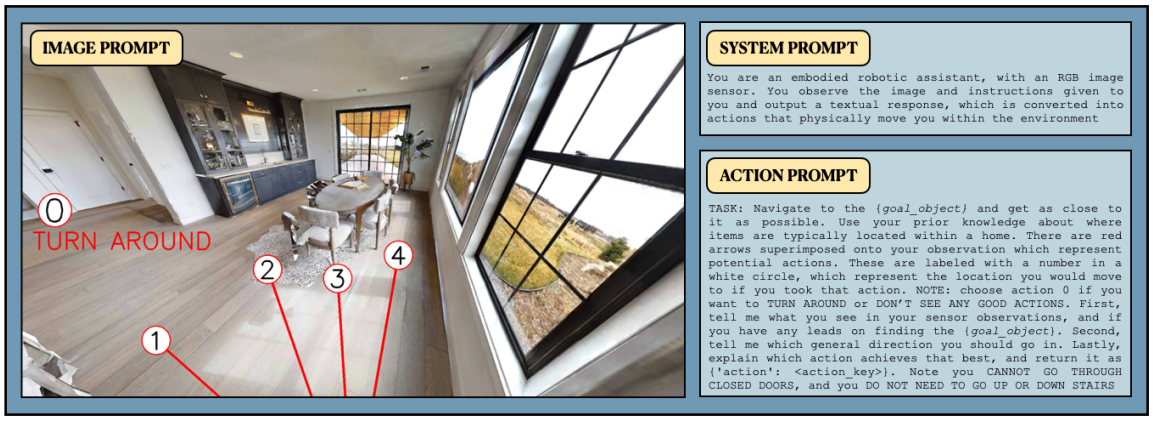

Figure 1: **Prompt:** The full action prompt for VLMnav consists of three parts: A system prompt to describe the embodiment, an action prompt to describe the task, the potential actions, and the output instruction, and an image prompt showing the current observation along with the annotated actions

Kuang et al. (2024). While modularity has the advantage of utilizing each component only for the sub-task it excels at, it comes at the disadvantage of system complexity and task specialization.

In this work, we show that an off-the-shelf VLM can be used as a zero-shot and end-to-end language-conditioned navigation policy. The key idea to achieve this goal is transforming the navigation problem into something VLMs excel at: *answering a question about an image*.

To do so, we develop a novel prompting strategy that enables VLMs to explicitly consider the problem of exploration and obstacle avoidance. This prompting is general, in the sense that it can be used for any vision-based navigation task.

Compared to prior approaches, we do not employ modality-specific experts Ren et al. (2024); Yu et al. (2023); Shah et al. (2023a), do not train any domain-specific models Zhang et al. (2024); Ehsani et al. (2023) and do not assume access to probabilities from the models Ren et al. (2024); Yu et al. (2023).

We evaluate our approach on established benchmarks for embodied navigation Yadav et al. (2022); Khanna* et al. (2024), where results confirm that our method significantly improves navigation performance compared to existing prompting methods. Finally, we draw design insights from ablation experiments over several components of our embodied VLM framework.

2. Related Work

The most common approach for learning an end-to-end navigation policy involves training a model from scratch using offline datasets Shah et al. (2021, 2023b); Chang et al. (2023); Shah et al. (2023d,c). However, collecting large-scale navigation data is challenging, and as a result, these models often struggle to generalize to novel tasks or out-of-distribution environments.

An alternative approach to enhance generalization is fine-tuning existing vision-language models (VLMs) with robot-specific data Brohan et al. (2022, 2023); Kim et al. (2024); Zhang et al. (2024). Although this method can lead to more robust end-to-end policies, fine-tuning may destroy features not present in the fine-tuning dataset, ultimately limiting the model's generalization ability.

An alternate line of work focuses on using these models zero-shot Kuang et al. (2024); Zhou et al. (2023); Yu et al. (2023); Shah et al. (2023a); Ren et al. (2024); Gadre et al. (2023); Nasiriany et al. (2024), by prompting them such that the responses align with task specifications. For instance, Gadre et al. (2023); Chang et al. (2023) use CLIP or DETIC features to align visual observations to language goals, build a semantic map of the environment, and use traditional methods for planning.

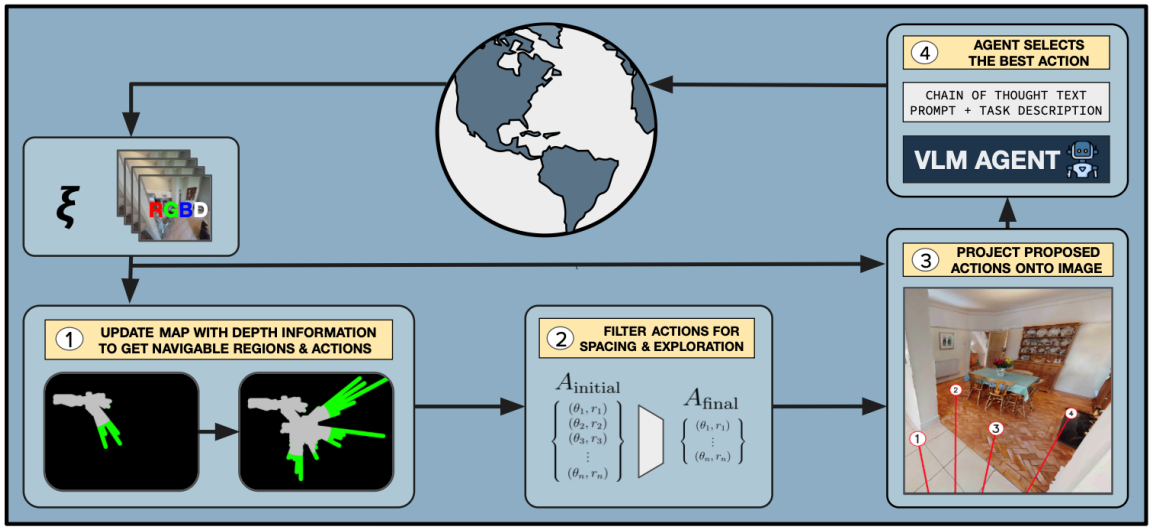

Figure 2: **Approach:** Our method is made up of four key components: (i) *Navigability*, which determines locations the agent can actually move to, and updates the voxel map accordingly. An example update step to the map shows the marking of new area as explored (gray) or unexplored (green). (ii) *Action Proposer*, which refines a set of final actions according to spacing and exploration. (iii) *Projection*, which visually annotates the image with actions. (iv) *Prompting*, which constructs a detailed chain-of-thought prompt to select an action

Other works design specific modules to handle the task of exploration Shah et al. (2023a); Ren et al. (2024); Kuang et al. (2024); Topiwala et al. (2018). These systems often require an estimation of confidence to know when to stop exploring, which is commonly done using token or object probabilities Ren et al. (2024); Yu et al. (2023). In addition, many of these approaches also use low-level navigation modules, which abstract away the action choices to a pre-trained point-to-point policy such as the Fast Marching Method Chang et al. (2023); Gadre et al. (2023); Shah et al. (2023a); Kuang et al. (2024); Yu et al. (2023).

Visual Prompting Methods: To enhance the task-specific performance of VLMs, recent work has involved physically modifying images before passing them to the VLM. Examples include Shtedritski et al. (2023), which annotates images to help recognize spatial concepts. Yang et al. (2023) introduces *set-of-mark*, which assigns unique labels to objects in an image and references these labels in the textual prompt to the VLM. This visual enhancement significantly improves performance on tasks requiring visual grounding. Building on this, Koh et al. (2024); Yan et al. (2023) apply similar visual prompting methods to the task of web navigation and show VLMs are able to complete such tasks zero shot.

Prompting VLMs for Embodied Navigation: CoNVOI Sathyamoorthy et al. (2024) overlays numerical markers on an image and prompts the VLM to output a sequence of these markers in alignment with contextual cues (e.g., *stay on the pavement*), which is used as a navigation path. Unlike our work, they (i) rely on a low-level planner for obstacle avoidance rather than using the VLM's outputs directly as navigational actions, and (ii) do not leverage the VLM to guide the agent toward a specific goal location. PIVOT Nasiriany et al. (2024), introduces a visual prompting method that is most similar to ours. They approach the navigation problem by representing one-step actions as arrows pointing to labeled circles on an image. At each step, actions are sampled from an isotropic Gaussian distribution, with the mean and variance iteratively updated based on feedback from the VLM. The final action is selected after refining the distribution. While PIVOT is capable

of handling various real-world navigation and manipulation tasks, it has two significant drawbacks: (i) it does not incorporate depth information to assess the feasibility of action proposals, leading to less efficient movement; and (ii) it requires many VLM calls to select a single action, resulting in higher computational costs and latency.

3. Overview

We present VLMnav, designed as a navigation system that takes as input goal $\mathcal{G}$, which can be specified in language or an image, RGB-D image I, pose ξ, and subsequently outputs action a. The action space consists of rotation about the yaw axis and displacement along the frontal axis in the robot frame, which allows all actions to be expressed in polar coordinates. As it is known that VLMs struggle to reason about continuous coordinates Rahmanzadehgervi et al. (2024), we instead transform the navigation problem into the selection of an action from a discrete set of options Yang et al. (2023). Our core idea is to choose these action options in a way that avoids obstacle collisions and promotes exploration.

Figure 2 summarizes our approach. We start by determining the navigability of the local region by estimating the distance to obstacles using a depth image (Sec. 3.1). Similar to Chang et al. (2023); Shah et al. (2023a); Ren et al. (2024); Sathyamoorthy et al. (2024); Gadre et al. (2023); Yu et al. (2023); Topiwala et al. (2018) we use the depth image and pose information to maintain a top-down voxel map of the scene, and notably mark voxels as *explored* or *unexplored*. Such a map is used by an *Action Proposer* (Sec. 3.2) to determine a set of actions that avoid obstacles and promote exploration. We then project this set of possible actions to the first-person-view RGB image with the *Projection* (Sec. 3.3) component. Finally, the VLM takes as input this image and a carefully crafted prompt, described in Sec. 3.4, to select an action, which the agent executes. To determine episode termination, we use a separate VLM call, detailed in Sec. 3.5.

3.1. Navigability

Using a depth image, we compute a *navigability mask* that contains the set of pixels that can be reached by the robot without crashing into any obstacles.

Next, for all directions $\theta \in fov$, we use the *navigability mask* to calculate the farthest straight-line distance r that the agent can travel without colliding. This creates a set of actions A_{initial} that are collision-free. Figure 3 illustrates an example calculation of the mask and navigable actions.'

At the same time, we use the depth image and the pose information to build a 2D voxel map of the environment. All observable areas within 2 meters of the agent are marked as *explored*, and the ones beyond as *unexplored*.

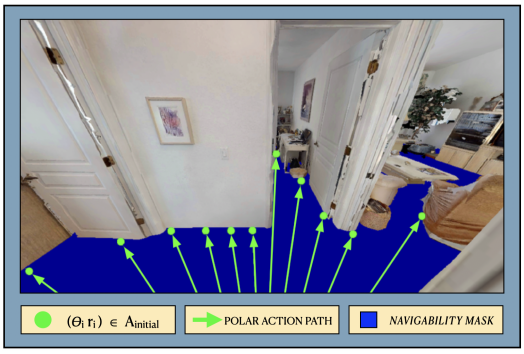

Figure 3: **Navigability:** An example step of the *Navigability* subroutine. The navigability mask is shown in blue and polar actions making up A_{initial} are in green

3.2. Action Proposer

We design the *Action Proposer* routine to refine $A_{\text{initial}} \rightarrow A_{\text{final}}$, an action set that is interpretable for the VLM and promotes exploration. Taking advantage of the information accumulated

in our voxel map, we look at each action and define an exploration indicator variable e_i as

$$e_i = \begin{cases} 1 & \text{if region } (\theta_i, r_i) \text{ is unexplored} \\ 0 & \text{if region } (\theta_i, r_i) \text{ is explored} \end{cases}$$

To build A_{final}, we need to prioritize unexplored actions, and also ensure there is enough visual spacing between actions for the VLM to discern. We start by adding unexplored actions to A_{final} if an angular spacing of θ_δ is maintained.

$$A_{\text{final}} \leftarrow A_{\text{final}} \cup \{(\theta_i, r_i) \mid e_i = 1 \text{ and } |\theta_i - \theta_j| \geq \theta_\delta, \forall (\theta_j, r_j) \in A_{\text{final}}\}$$

To sufficiently cover all directions but still maintain an exploration bias, we supplement A_{final} by adding explored actions subject to a *larger* angular spacing of $\theta_\Delta > \theta_\delta$:

$$A_{\text{final}} \leftarrow A_{\text{final}} \cup \{(\theta_i, r_i) \mid e_i = 0 \text{ and } |\theta_i - \theta_j| \geq \theta_\Delta, \forall (\theta_j, r_j) \in A_{\text{final}}\}$$

Lastly, we want to ensure these actions don't move the agent too close to obstacles, so we clip

$$r_i \leftarrow \min(\frac{2}{3} \cdot r_i,\ r_{max}) \quad \forall (\theta_i, r_i) \in A_{\text{final}}$$

Occasionally, the agent can get stuck in a corner where there are *no* navigable actions ($A_{\text{initial}} = \emptyset$). To address this, we add a special action $(\pi, 0)$, which rotates the agent by 180°. This also allows efficient entry/exit of rooms where the agent quickly identifies that the goal is not in that room.

The proposed set A_{final} now has three important properties: (i) actions correspond to navigable paths, (ii) there is sufficient visual spacing between actions, and (iii) there is an engineered bias towards exploration. We call this approach to exploration *explore bias*.

3.3. Projection

Visually grounding these actions in a space the VLM can understand and reason about is the next step. The *Projection* component takes in A_{final} from 3.2 and RGB image I, and outputs annotated image $\hat{I}$. Similarly to Nasiriany et al. (2024), each action is assigned a number and overlayed onto the image. We assign the special rotation action with 0 and annotate it onto the side of the image along with a label *Turn Around*. We find that visually annotating it, instead of just describing it in the textual prompt, helps ground its probability of being chosen to that of the other actions.

3.4. Prompting

To elicit a final action, we craft a detailed textual prompt T, which is fed into the VLM along with $\hat{I}$. This prompt primarily describes the details of the task, the navigation goal, and how to interpret the visual annotations. Additionally, we ask the model to describe the spatial layout of the image and to make a high-level plan *before* choosing the action, which serves to improve reasoning quality as found by Wei et al. (2023); Kojima et al. (2022). For image-based navigation goals, the goal image is simply passed into the VLM in addition to T and $\hat{I}$. The full prompt can be found in Figure 1.

The action chosen by the VLM, $P_{\text{vlm}}(a^*|\hat{I}, T) \in A_{\text{final}}$ is then directly executed in the environment. Notably, this does not involve any low-level obstacle avoidance policy as in other works Chang et al. (2023); Shah et al. (2023a); Gadre et al. (2023); Yu et al. (2023); Kuang et al. (2024).

TERMINATION PROMPT

```
The agent has the following navigation task: \n{task}\n. The agent
has sent you an image taken from its current location. Your job is to
determine whether the agent is close to the specified {goal_object}.
First, tell me what you see in the image, and tell me if there is a
{goal_object} that matches the description. Then, return 1 if the
agent is close to the {goal_object}, and 0 if it isn't. Format your
answer in the json {'done': <1 or 0>}
```

Figure 4: **Termination:** The separate prompt for determining episode termination

3.5. Termination

To complete a navigation task, the agent must terminate the episode by calling special action *stop* within a threshold distance of the goal object. Compared to other approaches that leverage a low-level navigation policy Chang et al. (2023); Shah et al. (2023a); Gadre et al. (2023); Yu et al. (2023); Kuang et al. (2024), our method does not explicitly choose a target coordinate location to navigate to, and therefore we face an additional challenge of determining when to stop. Our solution is to use a separate VLM prompt that explicitly asks whether or not to stop, which is shown in Figure 4. We do this for two reasons:

1. Annotations: The arrows and circles from Sec. 3.3 introduce noise and clutter to the image, making it more difficult to understand.

2. Separation of tasks. To avoid any task interference, the action call is only concerned with navigating and the stopping call is only concerned with stopping.

To avoid terminating the episode too far away from the object, we terminate the episode when the VLM calls *stop* two times in a row. After the VLM calls *stop* the first time, we turn off the navigability and explore bias components to ensure the agent doesn't move away from the goal object.

4. Experiments

We evaluate our approach on two popular embodied navigation benchmarks, ObjectNav Batra et al. (2020) and GoatBench Khanna* et al. (2024), which use scenes from the Habitat-Matterport 3D dataset Yadav et al. (2023); Savva et al. (2019). Further, we analyze how the performance of an end-to-end VLM agent changes with variations in design parameters such as field-of-view, length of the contextual history used to prompt the model, and quality of depth perception.

Setup: Similar to Yadav et al. (2022), the agent adopts a cylindrical body of radius 0.17m and height 1.5m. We equip the agent with an egocentric RGB-D sensor with resolution (1080, 1920) and a horizontal field-of-view (FOV) of 131°. The camera is tilted down with a pitch of 25° similar to Ren et al. (2024), which helps determine navigability. We use Gemini Flash as the VLM for all our experiments, given its low cost and high effectiveness.

Metrics: As in prior work Khanna* et al. (2024); Yadav et al. (2022); Anderson et al. (2018), we use the following metrics: (i) Success Rate (SR): fraction episodes that are successfully completed (ii) Success Rate Weighted by Inverse Path Length (SPL): a measure of path efficiency.

Baselines: We use PIVOT Nasiriany et al. (2024) as a baseline as it is most similar to ours. To investigate the impact of our action selection method, we ablate it by evaluating *Ours w/o nav*: the same as ours but without the *Navigability* and *Action Proposer* components. The action choices for this baseline are a static set of evenly-spaced action choices, including the *turn around* action. Notably, these actions do not consider navigability or exploration. To further evaluate the impact of visual annotation, we also evaluate a baseline *Prompt Only*, which sees actions described in text ("turn around", "turn right", "move forward", ...) but not annotated visually. These different prompting baselines can be visualized in Fig 5.

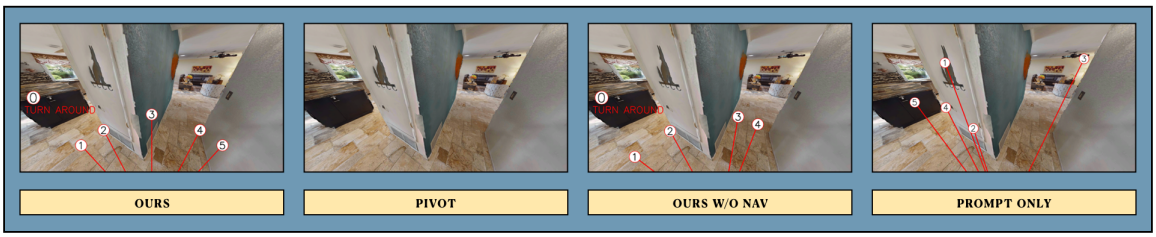

Figure 5: **Baselines:** Comparing the four different methods on a sample image. *Ours* contains arrows that point to navigable locations, *PIVOT* has arrows sampled from a random 2-D Gaussian, *Ours w/o nav* sees uniformly spaced arrows (note arrows 3 and 5 point into a wall), and *Prompt Only* sees just the raw RGB image

We note that in our experiments and baselines, we turn the *allow_slide* parameter on, which allows the agent to slide against obstacles in the simulator. Our experiments show that removing this assumption leads to large drops in performance.

4.1. ObjectNav

The Habitat ObjectNav benchmark requires navigation to an object instance from one of six categories *[Sofa, Toilet, TV, Plant, Chair, Bed]*. As in Yadav et al. (2022), to get the optimal path length, we take the minimum of the shortest paths to all instances of the object. These experiments are evaluated with a success threshold of 1.2 meters Shah et al. (2023a).

Run	SR	SPL
Ours	**50.4%**	**0.210**
Ours w/o nav	33.2%	0.136
Prompt Only	29.8%	0.107
PIVOT Nasiriany et al. (2024)	24.6%	0.106
Ours w/o sliding	12.9%	0.063

Table 1: **ObjectNav Results.** We evaluate four different prompting strategies on the ObjectNav benchmark, and see our method achieves highest performance in both accuracy (SR) and efficiency (SPL). Ablating the *allow_slide* parameter shows our method is dependent on sliding past obstacles

Table 1 summarizes our results. Our method outperforms PIVOT by over 25%, and nearly doubles its navigation efficiency in terms of SPL. We see that our action selection method is highly effective as shows a 17% improvement over *Ours w/o nav*. Removing visual annotations leads to a slight decrease in success rate but a significant reduction in SPL, indicating that visual grounding is important for navigation efficiency. Interestingly, we find that PIVOT performs worse than both of our ablations. We attribute this to limited expressivity in its action space, which prevents it from executing large rotations or turning around fully. This often leads to the agent getting stuck in corners, hindering its ability to recover and navigate effectively.

We note that disabling sliding results in a large drop in performance, signaling that while effective in simulation, our method would likely lead to collisions with obstacles in the real world. While our *Navigability* module can identify navigable locations, it does not consider the specific size and

shape of the robot in this calculation, leading to occasional collisions where the agent gets stuck since we lack an explicit action to backtrack previous motions.

4.2. Go To Anything Benchmark (GOAT)

GOAT Bench Khanna* et al. (2024) is a recent benchmark that establishes a higher level of navigation difficulty. Each episode contains 5-10 sub-tasks across three different goal modalities: (i) Object names, such as *refrigerator*, (ii) Object images, and (iii) Detailed text descriptions such as *Grey couch located on the left side of the room, next to the picture and the pillow*. Table 2 shows our results, evaluated on the val unseen split.

Run	SR	SPL	Image SR	Object SR	Description SR
Ours	**16.3%**	**0.066**	**14.3%**	**20.5%**	**13.4%**
Ours w/o nav	11.8%	0.054	7.8%	16.5%	10.2%
Prompt Only	11.3%	0.037	7.7%	15.6%	10.1%
PIVOT Nasiriany et al. (2024)	8.3%	0.038	7.0%	11.3%	5.9%

Table 2: **GOAT Results.** Comparison of prompting strategies on GOAT Bench, a more challenging navigation task. Across three different goal modalities, our method strongly outperforms baseline methods

Across all goal modalities, our model achieves significant improvements over baselines. These improvements are especially evident in image goals, where our model achieves nearly twice the success rate of all baseline methods. This highlights the robustness and general nature of our system. As with the ObjectNav results, *Ours w/o nav* and *Prompt only* perform comparable, and both outperform PIVOT. For all prompting methods, the image and description modalities prove more challenging than the object modality, similarly to what was found by Khanna* et al. (2024).

Comparison to state-of-the-art: We turn the *allow_slide* parameter off and compare to two state-of-the-art specialized approaches: (i) SenseAct-NN Khanna* et al. (2024) is a policy trained with reinforcement learning, using learned submodules for different skills; and (ii) Modular GOAT Chang et al. (2023) is a compound system that builds a semantic memory map of the environment and uses a low-level policy to navigate to objects within this map. Unlike SenseAct-NN, our work is zero-shot, and unlike Modular GOAT, we do not rely on a low-level policy or a separate object-detection module.

Run	SR	SPL
SenseAct-NN Skill Chain	**29.5%**	0.113
Modular GOAT	24.9%	**0.172**
Ours w/ sliding	16.3%	0.066
Ours	6.9%	0.049

Table 3: **Comparison to other works:** We see that specialized systems still produce superior performance. We also note these other works use a narrower FOV, lower image resolution, and a different action space, which could explain some of the differences

We compare the results of our approach to these baselines in Table 3. Interestingly, these methods have different strengths: a reinforcement learning approach leads to the highest success rate. Conversely, the modular navigation system achieves the highest navigation efficiency.

Our method shows lower performance compared to these specialized baselines across both metrics, even when permitted to slide over obstacles. Notably, we observe that in 13.9% of the runs, the VLM prematurely calls *stop* when it is between 1 to 1.5 meters from the target object. These instances are classified as failures, as the benchmark defines a run as successful only if the agent is within 1 meter of the object. This finding suggests that our VLM lacks the fine-grained spatial awareness necessary to accurately assess distances to objects. However, it also indicates that in

over 30% of the runs, our VLM agent is able to approach the goal object closely, highlighting its capability to reach near-target positions.

As shown in previous experiments, when not allowed to slide over objects, our approach's performance drastically decreases, as it gets frequently blocked between obstacles and does not have a way to backtrack its actions.

4.3. Exploring the design space of VLM agents for navigation

In this section, we look at major design choices that impact the navigation ability of VLM-based agents in our setup, all evaluated on the ObjectNav dataset.

4.3.1. How important is camera FOV for navigation?

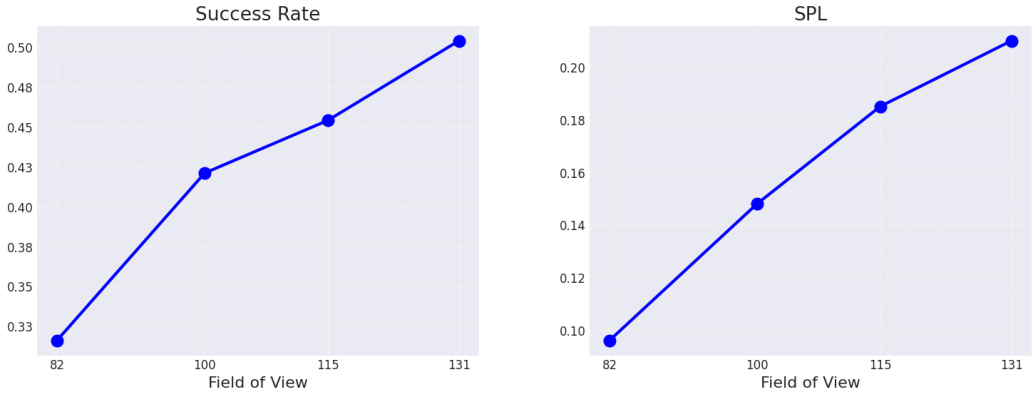

Figure 6: **Impact of sensor FOVs.** We evaluate the performance of four different sensor FOVs, and find that a wider FOV invariably leads to higher performance

An agent's navigation abilities largely depend on how fine-grained its vision is. In this section, we study whether our VLM agent can benefit from high-resolution images. Specifically, we run our method using four different FOVs: 82° Yadav et al. (2022), 100°, 115° and 131° (iPhone 0.5 camera). The results of this experiment, shown in Fig. 6, indicate positive scaling behaviors on both navigation accuracy and efficiency.

4.3.2. Do longer observation-action histories help?

In this section, we study whether a VLM navigation agent can effectively use a history of observations. We create a prompt containing the observation history in a naive way, i.e., we concatenate observations and actions from the K most recent environment steps and feed this into the VLM as context. For all these experiments, we remove our exploration bias (see Sec. 3.2) to specifically isolate the contribution of a longer history.

History Length	SR	SPL
No history	46.8%	0.193
5	42.7%	0.180
10	45.4%	0.196
15	40.4%	0.170

Table 4: **Impact of adding context history.** We compare our method to alternatives of keeping the past 0, 5, 10, and 15 observations and actions. We see that adding context history does not improve the performance of our method

The results of these experiments are shown in Table 4. We find that when naively concatenating past observations and actions, our prompt strategy is unable to use a longer context. Indeed, the performance remains the same or decreases when increasing the history length.

4.3.3. HOW IMPORTANT IS PERFECT DEPTH PERCEPTION?

Within the simulator, the depth sensor provides accurate pixel-wise depth information, which is important for determining the navigability mask. To investigate the importance of quasi-perfect depth perception, we evaluate two alternate approaches that only use RGB: (i) **Segformer**, which uses Xie et al. (2021) to semantically segment pixels belonging to the *floor* region. We use this region as the *navigability mask* and bypass the need for any depth information. We estimate the distances to obstacles by multiplying the number of pixels with a constant factor. (ii) **ZoeDepth**, which uses Bhat et al. (2023) to estimate metric depth values. We use such predicted values instead of the ground-truth distances from the simulator and compute navigability in the original way.

Run	SR	SPL
Depth sensor	50.4%	0.210
Segformer Xie et al. (2021)	47.2%	0.183
ZoeDepth Bhat et al. (2023)	39.1%	0.161

Table 5: **Depth Ablation.** We evaluate two alternate approaches that only require RGB. We find that semantic segmentation performs close to using ground truth depth, whereas estimating depth values leads to a significant performance drop

The results of this study are presented in Table 5. We find that depth estimation from Bhat et al. (2023) is not accurate enough to identify navigable areas. Indeed, depth noise leads to a 10% drop in SR. However, using a segmentation mask instead of relying on depth information surprisingly proves to be quite effective, with only a decrease of 3% with respect to using perfect depth perception. Overall, our experiments show that a VLM navigation agent can perform well with only RGB information.

5. Conclusion

In this work, we present VLMnav, a novel visual prompt-engineering approach that enables an off-the-shelf VLM to act as an end-to-end navigation policy. The main idea behind this approach is to carefully select action proposals and project them on an image, effectively transforming the problem of navigation into one of question-answering. Through evaluations on the ObjectNav and GOAT benchmarks, we see significant performance gains over the iterative baseline PIVOT, which was the previous state-of-the-art in prompt engineering for visual navigation. Our design study further highlights the importance of a wide field of view and the possibility of deploying our approach with minimal sensing, i.e., only an RGB image.

Our method has a few limitations. The drastic decrease in performance from disabling the *allow_slide* parameter indicates that there are several collisions with obstacles, which could be problematic in a real-world deployment. In addition, we find that specialized systems such as Khanna* et al. (2024) outperform our work. However, as the capabilities of VLMs continue to improve, we hypothesize that our approach could help future VLMs reach or surpass the performance of specialized systems for embodied tasks.

References

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R.

Zamir. On evaluation of embodied navigation agents, 2018. URL https://arxiv.org/abs/1807.06757.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects, 2020. URL https://arxiv.org/abs/2006.13171.

Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth, 2023. URL https://arxiv.org/abs/2302.12288.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Matthew Chang, Theophile Gervet, Mukul Khanna, Sriram Yenamandra, Dhruv Shah, So Yeon Min, Kavit Shah, Chris Paxton, Saurabh Gupta, Dhruv Batra, et al. Goat: Go to any thing. *arXiv preprint arXiv:2311.06430*, 2023.

Kiana Ehsani, Tanmay Gupta, Rose Hendrix, Jordi Salvador, Kuo-Hao Zeng Luca Weihs, Yejin Kim Kunal Pratap Singh, Winson Han, Alvaro Herrasti, Ranjay Krishna, Dustin Schwenk, Eli VanderBilt, and Aniruddha Kembhavi. Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. *arXiv*, 2023.

Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

Mukul Khanna*, Ram Ramrakhya*, Gunjan Chhablani, Sriram Yenamandra, Theophile Gervet, Matthew Chang, Zsolt Kira, Devendra Singh Chaplot, Dhruv Batra, and Roozbeh Mottaghi. Goat-bench: A benchmark for multi-modal lifelong navigation, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks, 2024. URL https://arxiv.org/abs/2401.13649.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Yuxuan Kuang, Hai Lin, and Meng Jiang. Openfmnav: Towards open-set zero-shot object navigation via vision-language foundation models, 2024. URL https://arxiv.org/abs/2402.10670.

Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35:32340–32352, 2022.

Rubén N. Muzio and Verner P. Bingman. *Brain and Spatial Cognition in Amphibians: Stem Adaptations in the Evolution of Tetrapod Cognition*, page 105–124. Cambridge University Press, 2022.

Soroush Nasiriany, Fei Xia, Wenhao Yu, Ted Xiao, Jacky Liang, Ishita Dasgupta, Annie Xie, Danny Driess, Ayzaan Wahid, Zhuo Xu, Quan Vuong, Tingnan Zhang, Tsang-Wei Edward Lee, Kuang-Huei Lee, Peng Xu, Sean Kirmani, Yuke Zhu, Andy Zeng, Karol Hausman, Nicolas Heess, Chelsea Finn, Sergey Levine, and Brian Ichter. Pivot: Iterative visual prompting elicits actionable knowledge for vlms, 2024.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, and Ilge Akkaya et al. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind, 2024. URL https://arxiv.org/abs/2407.06581.

Santhosh Kumar Ramakrishnan, Erik Wijmans, Philipp Kraehenbuehl, and Vladlen Koltun. Does spatial cognition emerge in frontier models?, 2024. URL https://arxiv.org/abs/2410.06468.

Allen Z. Ren, Jaden Clark, Anushri Dixit, Masha Itkina, Anirudha Majumdar, and Dorsa Sadigh. Explore until confident: Efficient exploration for embodied question answering. In *arXiv preprint arXiv:2403.15941*, 2024.

Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, Mohamed Elnoor, Anuj Zore, Brian Ichter, Fei Xia, Jie Tan, Wenhao Yu, and Dinesh Manocha. Convoi: Context-aware navigation using vision language models in outdoor and indoor environments, 2024. URL https://arxiv.org/abs/2403.15637.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research, 2019. URL https://arxiv.org/abs/1904.01201.

Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Ving: Learning open-world navigation with visual goals. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13215–13222, 2021. doi: 10.1109/ICRA48506.2021.9561936.

Dhruv Shah, Michael Equi, Blazej Osinski, Fei Xia, Brian Ichter, and Sergey Levine. Navigation with large language models: Semantic guesswork as a heuristic for planning. In *7th Annual Conference on Robot Learning*, 2023a. URL https://openreview.net/forum?id=PsV65r0itpo.

Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models, 2023b. URL https://arxiv.org/abs/2104.05859.

Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. Gnm: A general navigation model to drive any robot, 2023c. URL https://arxiv.org/abs/2210.03370.

Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation, 2023d. URL https://arxiv.org/abs/2306.14846.

Aleksandar Shtedritski, Christian Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms, 2023. URL https://arxiv.org/abs/2304.06712.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, and Anmol Gulati et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024. URL https://arxiv.org/abs/2403.05530.

Anirudh Topiwala, Pranav Inani, and Abhishek Kathpal. Frontier based exploration for autonomous robot, 2018. URL https://arxiv.org/abs/1806.03581.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021. URL https://arxiv.org/abs/2105.15203.

Karmesh Yadav, Santhosh Kumar Ramakrishnan, John Turner, Aaron Gokaslan, Oleksandr Maksymets, Rishabh Jain, Ram Ramrakhya, Angel X Chang, Alexander Clegg, Manolis Savva, Eric Undersander, Devendra Singh Chaplot, and Dhruv Batra. Habitat challenge 2022. https://aihabitat.org/challenge/2022/, 2022.

Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, Alexander William Clegg, and Devendra Singh Chaplot. Habitat-matterport 3d semantics dataset, 2023. URL https://arxiv.org/abs/2210.05633.

An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, Zicheng Liu, and Lijuan Wang. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation, 2023. URL https://arxiv.org/abs/2311.07562.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*, 2023.

Bangguo Yu, Hamidreza Kasaei, and Ming Cao. L3mvn: Leveraging large language models for visual target navigation. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.

Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models, 2023. URL https://arxiv.org/abs/2305.16986.