VLM-Guided Subgoal Planning for Indoor Navigation in Habitat-Lab

Aayush Fadia
Department of Computer Science
afadia@andrew.cmu.edu

Megan Lee
Department of Computer Science
meganlee@andrew.cmu.edu

Ashish Marisetty
Department of Computer Science
amariset@andrew.cmu.edu

Abstract

Indoor navigation in home environments requires agents to jointly reason about object semantics, spatial layout, and low-level control, often leading to poor sample efficiency in end-to-end reinforcement learning (RL) approaches. We propose augmenting RL policies with guidance from Vision-Language Models (VLMs), leveraging semantic and spatial knowledge from internet-scale pretraining to handle high-level reasoning while allowing the RL policy to focus on control. Our approach extends a standard PPO agent with a `Think` action that queries a lightweight VLM to generate natural language subgoals, which are embedded using CLIP and provided as additional observations to the policy. Evaluated on the Habitat-Matterport3D (HM3D) Object Navigation task, our method improves path efficiency by up to 35% SPL in single-environment settings while achieving comparable success rates with fewer environment steps. When scaled to multiple environments, the VLM-integrated policy maintains these trends, yielding approximately 11% higher success rates and 35% higher SPL relative to the baseline, albeit with increased training requirements. These results demonstrate that simple, on-demand integration of VLM-generated subgoals can improve navigation efficiency without requiring substantial architectural modifications.

1 Introduction

This project addresses the challenge of designing effective control policies for robotic systems operating on long-horizon object-goal navigation tasks in complex, realistic indoor environments. We experimented using the Habitat-Lab simulator [6] and the Habitat-Matterport3D (HM3D) dataset [5], which provides over 1,000 high-fidelity 3D scans of real indoor environments. The core task, Object Navigation (ObjectNav), requires an agent to navigate from a random starting position to a specified goal object (e.g., "chair," "sofa," "microwave") illustrated in Figure 1[1]. The agent uses a discrete action space: move forward (0.25m), turn left (10 degrees), turn right (10 degrees), and stop. Success is defined by a strict criterion: the agent must execute a stop action while located within 1.0m (geodesic distance) of the target object and facing it with an unobstructed line of sight. This rigorous standard tests both object localization and precise positioning.

The central premise of our work is that Vision-Language Models (VLMs), pretrained on massive internet-scale datasets, possess intrinsic world knowledge and common-sense spatial reasoning that can be leveraged as high-level planners for embodied navigation. VLMs such as CLIP [3] learn joint representations of visual and linguistic concepts through contrastive learning on hundreds of millions of image-text pairs, enabling them to understand semantic relationships (e.g., "beds are

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

typically found in bedrooms") without task-specific training. This learned alignment between vision and language provides a promising foundation for hierarchical navigation systems where VLMs handle high-level semantic reasoning while RL policies focus on low-level control.

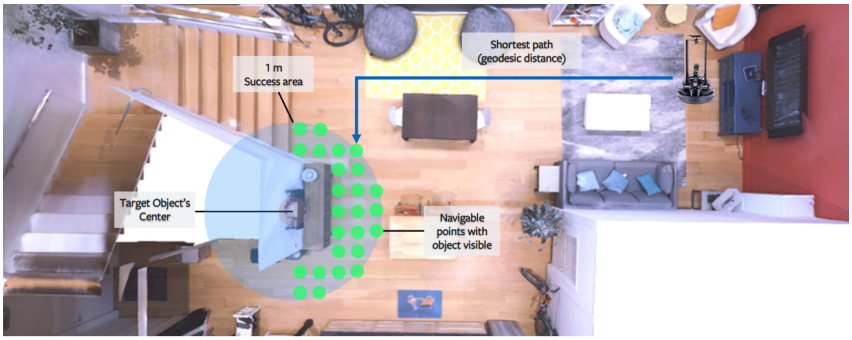

Figure 1: ObjectNav evaluation criteria illustration from [1]. The goal object (monitor) is highlighted. Success requires the agent to stop within a geodesic distance threshold from navigable locations where the object is visible with proper camera orientation (green region).

2 Relevant Work

Early approaches to Object Goal Navigation (ObjectNav) primarily used end-to-end Reinforcement Learning (RL), where an agent learns to map visual observations directly to actions. While conceptually simple, these methods suffered from poor sample efficiency and slow convergence, struggling to simultaneously learn object recognition, spatial reasoning, and precise control. For instance, an end-to-end DD-PPO baseline often required millions of environment frames but yielded low Success Rate and SPL[2].

To overcome these limitations, modular approaches emerged, decomposing ObjectNav into separate components like perception, planning, and control. SemExp [2] introduced Goal-Oriented Semantic Exploration, which builds semantic maps of the environment to guide exploration based on the target object's typical location (e.g., a refrigerator in a kitchen). This modularity significantly improved performance and efficiency. PONI [4] further refined this by treating object search as a perception problem, learning potential functions on semantic maps via supervised learning, which drastically reduced the training cost compared to RL-based methods.

More recently, the power of Vision-Language Models (VLMs), pre-trained on massive internet-scale datasets, has been leveraged for embodied navigation. Models like CLIP [3] demonstrate a rich, inherent understanding of object semantics and spatial relationships. Works such as LM-Nav [7] successfully integrated large language models with VLMs to extract landmarks and ground them through CLIP, enabling complex, long-horizon navigation from natural language instructions without task-specific fine-tuning. Other methods like NaVid [9] and NaviLLM [10] have continued to show that VLMs can act as powerful, generalist high-level planners by utilizing their learned world knowledge to guide the navigation process. This work aims to investigate the effective integration of this VLM high-level guidance into the low-level control loop of an RL-trained policy.

3 Methodology

3.1 PPO Baseline

We adapted a DD-PPO (Decentralized Distributed Proximal Policy Optimization) agent using the standard Habitat-Lab training pipeline. The agent architecture consists of a ResNet50 visual encoder backbone pretrained on the Gibson dataset, combined with a recurrent LSTM policy network featuring 2 layers and 512 hidden units. The agent processes three types of observations at each timestep: RGB images at 256×256 resolution, depth maps at 256×256 resolution, and the agent's own pose information (position and heading). The action space is discrete, consisting of four actions: move forward (0.25m), turn left (10 degrees), turn right (10 degrees), and stop. We used a learning rate of

Figure 2: RGB input for agent. Metrics are shown in top left with VLM outputs.

2.5×10^{-4} with a PPO clip parameter of 0.2, following standard hyperparameters from published Habitat baselines. The PPO algorithm used 4 epochs per update with 2 mini-batches, a discount factor $\gamma = 0.99$, and GAE parameter $\tau = 0.95$.

3.2 VLM Integration

In this section, we describe how we integrate a Visual Language Model (VLM) into our policy to leverage large-scale, internet-derived knowledge for navigating complex environments.

3.2.1 VLM Choice and Inference Setup

We use Qwen3-VL-2B [8] to generate navigation subgoals. This model demonstrates strong visual and spatial reasoning capabilities, enabling it to understand both the semantic and geometric structure of household environments. These properties make it well-suited for producing meaningful and actionable subgoals. Additionally, the model is lightweight and open source, allowing us to run inference efficiently on the same machine used for training the RL agent.

3.2.2 Interaction with the Policy

We explored multiple strategies for integrating a Visual Language Model (VLM) with the policy. In particular, we considered:

- invoking the VLM once at the beginning of an episode to generate a fixed sequence of textual instructions
- querying the VLM at a fixed frequency (every N steps) to provide periodic subgoals

While these approaches provide high-level guidance, they either lack adaptability to unforeseen states or incur unnecessary computational overhead. Instead, we adopt an agent-driven approach in which the policy autonomously decides when to request high-level guidance. Concretely, we extend the action space of the standard PPO agent with an additional discrete action, `Think`. When selected, this action triggers a VLM inference call using the agent's current egocentric RGB observation and a prompt designed to elicit a concise navigation subgoal relevant to the current scene and task.

At the beginning of each episode, we initialize the `Thought` sensor with the CLIP embedding of the task description (e.g., the target object category), providing the agent with global semantic context. When a `Think` action is executed, the generated subgoal text is embedded using CLIP and stored as the current `Thought`. To condition the policy on this information, we augment the observation space with an additional sensor, referred to as `Thought`, as seen in Figure 2. At each timestep, this sensor provides a fixed 512-dimensional embedding corresponding to the most recently generated subgoal.

At timesteps where no `Think` action is executed, the `Thought` sensor is set to a zero vector. This design ensures a consistent observation dimensionality and prevents stale subgoals from persisting across timesteps, allowing high-level guidance to be used only when explicitly requested by the policy.

Our policy architecture otherwise follows the default `habitat-baselines` setup, including a visual encoder for RGB and depth observations, and a LSTM model for temporal aggregation. The key modification lies in replacing the standard visual feature representation with CLIP embeddings for both the RGB observation and the `Thought` sensor. By embedding both modalities into a shared semantic space, the policy can directly reason about the alignment between its current visual context and the language-derived guidance, enabling more effective coordination between high-level planning and low-level control.

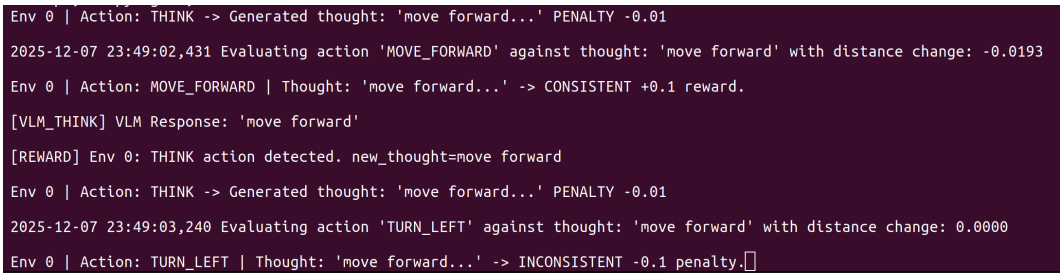

Figure 3: VLM response and rewards

3.2.3 Thinking Penalties and Rewards

In initial experiments, we retained the default reward structure to observe the agent's behavior. Under this setup, the policy excessively selected the `Think` action. To discourage this, we introduced a small negative reward for invoking `Think`. However, this alone caused the agent to entirely avoid the action, effectively reverting to baseline behavior.

To address this, we adopt a combined reward scheme that retains the `Think` penalty while providing a small positive reward when the agent's subsequent action aligns with the direction suggested by the generated subgoal (e.g., rewarding a left turn following a subgoal containing "left"). This auxiliary reward is added to the existing task rewards and encourages selective, meaningful use of the `Think` action. The resulting interaction is illustrated in Figure 3.

4 Results and Analysis

In this section, we analyze the performance of our VLM-integrated policy under progressively simplified training setups. We begin with the full HM3D dataset to assess scalability, and then move to controlled single and multi-environment settings to isolate the impact of language-guided subgoals on navigation performance and sample efficiency.

Aside from the success rate, a key evaluation metric we used was Success weighted by Path Length (SPL), which measures both navigation success and path efficiency by computing the ratio of the shortest path length to the actual path taken, averaged over successful episodes:

$$\text{SPL} = \frac{1}{N} \sum_{i=1}^{N} S_i \frac{l_i}{max(p_i, l_i)} \tag{1}$$

where S_i is a binary success indicator, l_i is the shortest path length, and p_i is the actual path length for episode i. We also looked at Soft SPL, a relaxed variant that gives partial credit based on the agent's final distance to the goal rather than requiring strict success criteria.

4.1 Full HM3D dataset

We first attempted to train our VLM-integrated model on the full HM3D dataset on 1,000 buildings covering over $112{,}500$ m^2 of navigable space. Despite training for 30 hours to 2.6 million steps, the

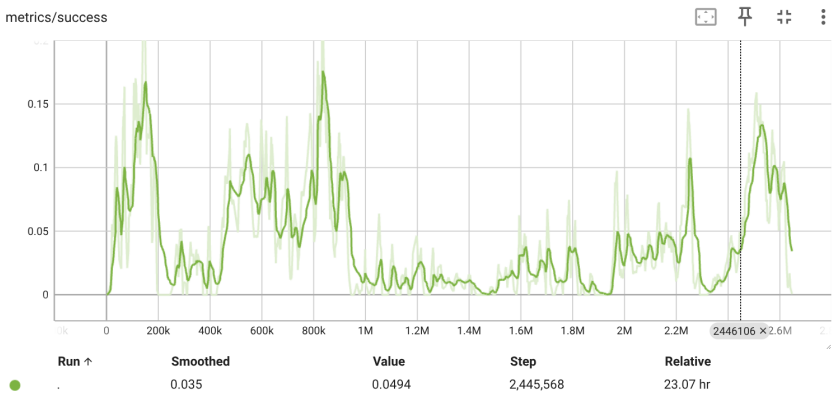

Figure 4: Full HM3D dataset success rate

agent's success rate exhibited high variance and poor convergence Figure4. Given that the HM3D baseline paper required approximately 1.6 billion environment steps for high performance [2], we determined that pursuing the full training trajectory would be computationally infeasible and ceased the full-scale experiment.

4.2 Single Environment

To isolate and better understand the differences between the baseline PPO policy and our VLM-integrated model, we restrict our evaluation to a single scene. We perform a stratified split over object categories, allocating 90% of the data for training and the remaining 10% for testing. Our results show that the VLM-integrated model matches or exceeds the baseline's success rate while requiring fewer environment steps, indicating improved sample efficiency. Additionally, the agent follows more direct and optimal navigation paths, reflected by a 35% improvement in SPL metric.

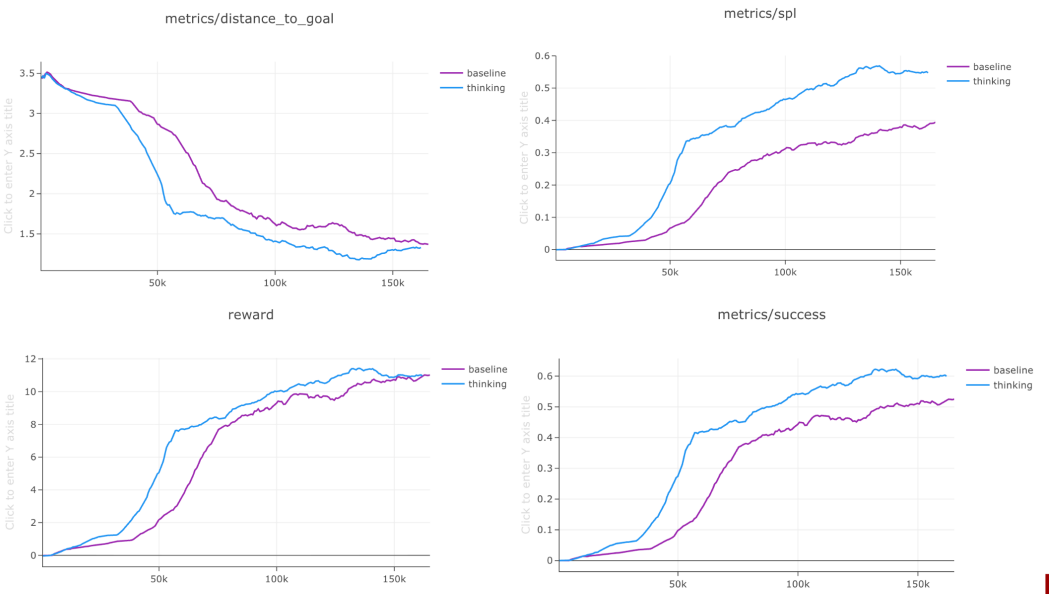

Figure 5: Metrics from single environment training (160k steps). Blue line is PPO baseline and purple is VLM-integrated model.

4.3 Multiple Environments

For our second experiment, we extend the single-environment setup to five scenes with the lowest navigation complexity. The evaluation protocol remains the same, using a 90%–10% train–test split. As shown in Figure 6, the relative trends across metrics are consistent with those observed in the single-environment setting. We improve 11% on the success rate and 35 % on the SPL metric over the baseline. However, the absolute metric values are significantly lower, and convergence requires substantially longer training (up to 300k steps). This supports our initial hypothesis that scaling to multiple scenes increases sample and compute requirements, particularly given the inherent navigation difficulty of the HM3D dataset.

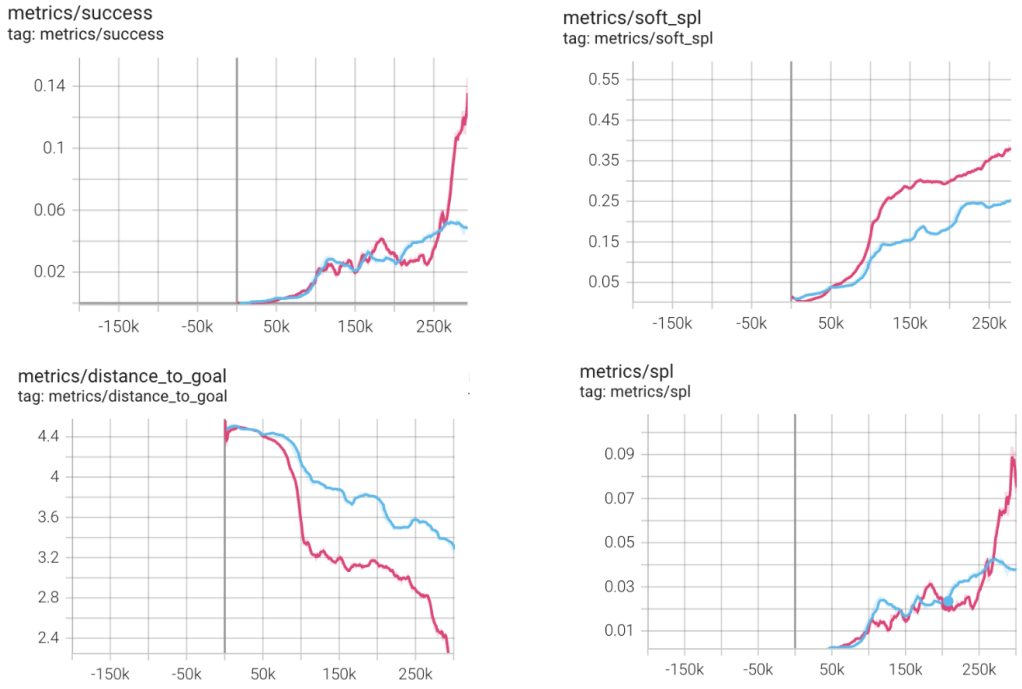

Figure 6: Metrics from 5 environment training (300k steps). The magenta line is our VLM-integrated policy, and the red line is our baseline.

4.4 Analysis

In our initial reward schemes, the agent quickly learned to exploit the custom VLM-consistency rewards. Since this positive signal (for taking an action aligned with the VLM's generated subgoal) was immediate and dense, it often became the dominant training objective, outweighing the sparse terminal success reward.

While we introduced a negative penalty on the `Think` action and rescaled the reward function, the tendency toward short-term optimization persists. The policy, while now utilizing the VLM for guidance, still exhibits a bias toward maximizing immediate VLM-alignment rewards, rather than prioritizing the long-horizon `distance_to_goal` minimization.

This trade off is particularly evident as we scale from the single environment setting to five environments. The increase in navigational complexity makes the global reward signal sparser, causing the agent to over-rely on the local VLM-consistency signal. We noticed in our multiple environment training that the agent follows suboptimal trajectories, performing unnecessary turns or maneuvers that satisfy the immediate VLM subgoal but do not efficiently advance the agent toward the final object goal. This suggests a direction for future work in designing more sophisticated reward shaping to effectively bridge the gap between high-level semantic subgoals and the low-level geometric navigation objective.

5 Impact

With the current paradigm, RL agents that act in the real world must necessarily also learn to interpret and reason about the world (such as where things are commonly found in a household in our case). Offloading this burden to modern VLMs trained on internet-scale data means that the RL policy has lesser to learn, and is much more capable.

This is especially important in tacking long-tail scenarios that are underrepresented in policy training data. Consider self-driving, where unique safety-critical scenarios can occur that are not present in the training data for the RL policy driving the car. A modern VLM, however, may be able to reason soundly about these situations and be able to guide the policy to take the correct steps. To an unassisted policy, such a situation could be out-of-distribution, causing the policy to fail when faced with that situation.

6 Conclusion and Future Work

Our results show that enabling RL policies to selectively leverage the reasoning capabilities of VLMs leads to faster training and improved navigation performance. Across both single and multi-environment settings, the VLM-integrated policy achieves higher path efficiency and competitive success rates compared to a PPO baseline. Notably, these gains are achieved through a simple integration that reuses existing action and observation abstractions, requiring no changes to the underlying PPO architecture.

A promising direction for future work is enabling the agent to more effectively *learn when to think*. In this work, we partially address this through reward shaping that balances a penalty for invoking the `Think` action with auxiliary rewards for following useful subgoal guidance. More principled approaches could include further tuning of this reward structure or introducing an explicit regularization term or auxiliary loss that encourages the agent to invoke `Think` only when the expected value of high-level reasoning outweighs its cost. Such mechanisms could improve sample efficiency and reduce unnecessary VLM queries, further strengthening the practicality of VLM-guided embodied agents.

References

[1] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. 2020.

[2] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *Advances in Neural Information Processing Systems*, volume 33, pages 4247–4258, 2020.

[3] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[4] S. K. Ramakrishnan, D. S. Chaplot, Z. Al-Halah, J. Malik, and K. Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18890–18900, 2022.

[5] S. K. Ramakrishnan, A. Gokaslan, E. Wijmans, O. Maksymets, A. Clegg, J. Turner, E. Undersander, W. Galuba, A. Westbury, A. X. Chang, M. Savva, Y. Zhao, and D. Batra. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[6] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019.

[7] D. Shah, B. Osiński, B. Ichter, and S. Levine. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 492–504. PMLR, 2023.

[8] Q. Team. Qwen3 technical report, 2025.

[9] J. Zhang, K. Wang, R. Xu, G. Zhou, Y. Hong, X. Fang, Q. Wu, Z. Zhang, and H. Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. In *Robotics: Science and Systems*, 2024.

[10] D. Zheng, S. Huang, L. Zhao, Y. Zhong, and L. Wang. Towards learning a generalist model for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11200–11210, 2024.