# What Matters in RL-Based Methods for Object-Goal Navigation? An Empirical Study and A Unified Framework

Hongze Wang[1], Boyang Sun[1], Jiaxu Xing[2], Fan Yang[3], Marco Hutter[3], Dhruv Shah[4], Davide Scaramuzza[2], and Marc Pollefeys[1,5]

[1] Computer Vision and Geometry Group, ETH Zurich, Switzerland
[2] Robotics and Perception Group, University of Zurich, Switzerland
[3] Robotic Systems Lab, ETH Zurich, Switzerland
[4] Princeton Robotic Intelligence and SysteMs group, Princeton University, USA
[5] Microsoft, Switzerland

**Abstract.** Object-Goal Navigation (ObjectNav) is a key capability for deploying mobile robots in everyday environments such as homes, schools, and workplaces. In this task, an agent must locate an instance of a target object category in previously unseen environments using only onboard perception, requiring the integration of semantic understanding, spatial reasoning, and long-horizon planning. Reinforcement learning (RL) has become a dominant paradigm for ObjectNav, yet modern systems involve numerous design choices across perception modules, policy architectures, and inference-time strategies. The relative impact of these components, however, remains poorly understood. In this work, we present a large-scale empirical study of modular RL-based ObjectNav systems. We decompose the navigation pipeline into three key components: *perception*, *policy*, and *test-time enhancement*, and conduct extensive controlled experiments to analyze their individual contributions. Our results suggest that improvements in perception quality and test-time strategies often yield larger performance gains than policy improvements alone, highlighting the importance of understanding how different components interact within modular navigation systems. Motivated by these findings, we introduce a unified framework for systematically studying modular ObjectNav systems. Guided by our analysis, we build an enhanced system that achieves state-of-the-art performance on the Gibson benchmark, improving SPL by 6.6% and success rate by 2.7% over prior methods. We also introduce a human expert baseline, achieving 98% success, highlighting the significant gap between current RL agents and human-level navigation. Finally, we provide practical insights and design recommendations for each module to help guide future research. Project page: `https://honwang0054.github.io/What-matters-in-RL-ObjNav-web/`.

## 1 Introduction

Recent advances in computer vision and deep learning have inspired growing interest in interdisciplinary applications that bridge perception, reasoning, and

control, especially in robotics. Among these, vision-based navigation has emerged as a foundational capability for autonomous mobile agents. A key benchmark in this domain is *Object-Goal Navigation (ObjectNav)*, where a robot must navigate to an instance of a specified object category in an unseen environment, relying solely on its onboard sensors [2,5,18,29]. This task is both practically important and technically challenging: it requires semantic understanding, spatial reasoning, and long-horizon planning. Among many approaches, Reinforcement Learning (RL) has become a dominant paradigm for ObjectNav, offering a structured framework to learn directly through trial-and-error and showing steady progress across various benchmarks [4, 5, 21, 35]. While end-to-end RL policies are com-

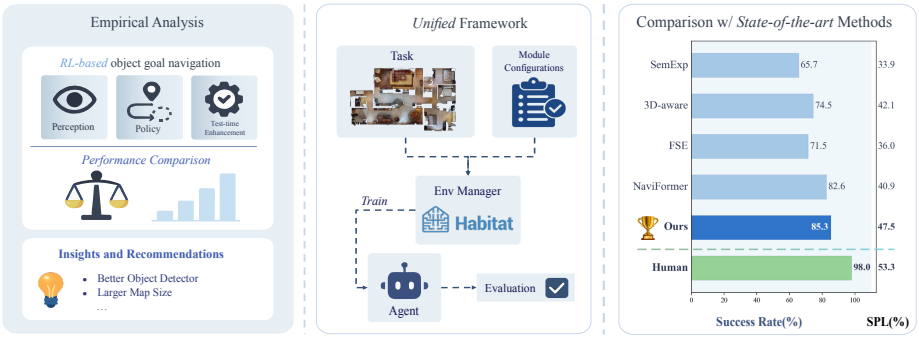


**Fig. 1:** Our framework encompasses: (1) an empirical study analyzing the impact of different modules, and (2) a unified framework with interchangeable components, enabling users to customize their own object-goal navigation policies.

mon, modular RL approaches have shown greater robustness and improved generalization. By decomposing the system into interpretable and tunable components, such as perception, mapping, policy, and action execution, these methods align with the multi-faceted nature of ObjectNav and often achieve better sim-to-real transfer [10]. However, this modular design also increases overall system complexity, and the standalone contribution of each component has not been systematically studied in recent literature. As a result, current bottlenecks remain unclear, and the lack of a unified design guide makes it difficult to fairly evaluate and advance modular RL systems. Despite its importance, this question has received limited attention in the community, leaving the relative impact of different modules largely unexplored.

With these observations, in this work, we first disentangle the components of modular RL-based ObjectNav and ask a fundamental question: *Which design choices truly matter for RL-based ObjectNav?* We establish a principled decomposition of modular ObjectNav systems into three essential parts—**perception**, **policy**, and **test-time enhancement**. For each component, we categorize representative design choices from the literature and, through extensive experiments, isolate the individual contribution of each, an overview of our approach and results is shown in Fig. 1.

Our key findings are summarized as follows: (a) The capability of the perception module has a substantial impact on overall navigation performance. (b) Test-time enhancement strategies are often overlooked, yet they can provide significant improvements during deployment. (c) By establishing a unified configuration for perception and test-time strategies, differences between policy designs can be more reliably measured and analyzed, enabling clearer evaluation and facilitating future improvements in policy learning.

Based on these findings, we offer practical recommendations for selecting and combining system components, along with insights into the reasoning behind them. Following these guidelines, we design an enhanced modular system that achieves state-of-the-art performance with 47.5% SPL (success weighted by path length) and 85.3% success rate, surpassing the best prior methods by 6.6% in SPL and 2.7% in success rate. In addition, we introduce a human baseline comparison, where expert participants operate under the same training setup, test environments, and observation modalities as the agent, achieving an average of 98.0% success rate and 53.3% SPL. This contrast reveals a clear gap between current RL-based systems and human-level navigation, emphasizing the need for new algorithms that can enhance both performance and robustness in ObjectNav.

Beyond empirical improvements, our goal is to provide broader insights for the vision-based navigation community, from guiding the design of future algorithms to enabling fairer evaluation of modular ObjectNav systems. To facilitate further research, we publicly release our code and evaluation tools at `https://github.com/honwang0054/What-matters-in-RL-ObjNav-release`.

## 2 Related Work

**End-to-end Object-Goal Navigation.** These approaches leverage reinforcement learning or imitation learning to train a policy that directly maps visual observations to low-level actions. Early works by [22, 23, 41] followed the architectural blueprint of DD-PPO [30], employing convolutional neural networks (CNNs) coupled with recurrent neural networks (RNNs). Subsequent methods [15, 38] enhanced this framework by replacing the RGB encoder with more powerful self-supervised vision models, leading to improved performance. More recent efforts [8, 45] have adopted transformer-based policy architectures to better capture long-term spatial dependencies, achieving state-of-the-art results. To further enrich spatial reasoning, several works have proposed integrating structured scene representations [9] into the policy network. These methods [11, 19, 40, 47, 50] build scene graphs from RGB inputs, encode them with Graph Neural Networks (GNNs), and feed the features into the policy. One major downside of end-to-end methods is that they require large-scale data and struggle to generalize across the sim-to-real gap [10].

**Modular RL Methods for Object-Goal Navigation.** These approaches integrate learning-based components with classical map-based navigation to achieve improved data efficiency, lower computational overhead, and competitive perfor-

mance. One representative approach is ANS by [6], which employs a reinforcement learning-based pipeline for navigation in 3D environments. ANS consists of a mapping module that projects RGB-D observations into a top-down occupancy map, along with a long-term goal policy trained via reinforcement learning to guide exploration. Building on ANS, SemExp [5] introduces a semantic mapping module that augments the top-down map with object-level semantic annotations. Subsequent work largely follows this modular reinforcement learning framework with various enhancements. More recently, vision-based navigation has attracted increasing interest from the computer vision community, with several works focusing on improving visual representations and spatial reasoning for embodied navigation [7, 28, 45, 46, 49]. For instance, FSE by [42] incorporates frontier detection into the mapping process and selects sub-goals based on the centers of frontier regions. 3D-Aware [46] further utilizes 3D structural cues to improve scene understanding and predicts corner-based sub-goals, inspired by heuristic strategies like those in [17]. More recently, NaviFormer [35] achieves state-of-the-art performance by replacing traditional map encoders with a transformer-based architecture, enabling richer spatial-semantic representation learning. While existing modular methods primarily focus on improving navigation policies, the relative contribution of different system components remains insufficiently understood. In this work, we present a comprehensive study of modular ObjectNav architectures, systematically evaluating how individual components influence overall navigation performance.

## 3 Study Design

**Problem Formulation.** In the context of ObjectGoal navigation, the agent is required to navigate to an instance of a specified object category within an unseen environment. At each timestep $t$, the agent receives an RGB-D observation $o_t \in \mathbb{R}^{4\times H\times W}$, a goal category label $g$, and its current pose $p_t$. Based on these inputs, the agent generates actions to move towards the goal. We use Habitat [18] as a unified simulation testbed, which has a built-in low-level controller that outputs one of four discrete actions: `move forward`, `turn left`, `turn right`, and `stop`.

**System Overview.** Figure 1 outlines the structure of our study and analysis. We follow the natural pipeline of robot navigation and decompose a typical RL-based approach into three core modules: perception, policy, and test-time enhancement. The perception module covers the process from raw sensor inputs to building representations of the environment. The policy module takes the output of the perception module and predicts the next action based on its action space.

The test-time enhancement module contains inference-time strategies that improve performance during execution by adjusting the agent's behavior without retraining. These strategies operate alongside the learned policy to correct common failure patterns and strengthen the overall perception–action loop.

For each of these three components, we survey a wide range of design choices found in recent literature and conduct extensive experiments under controlled

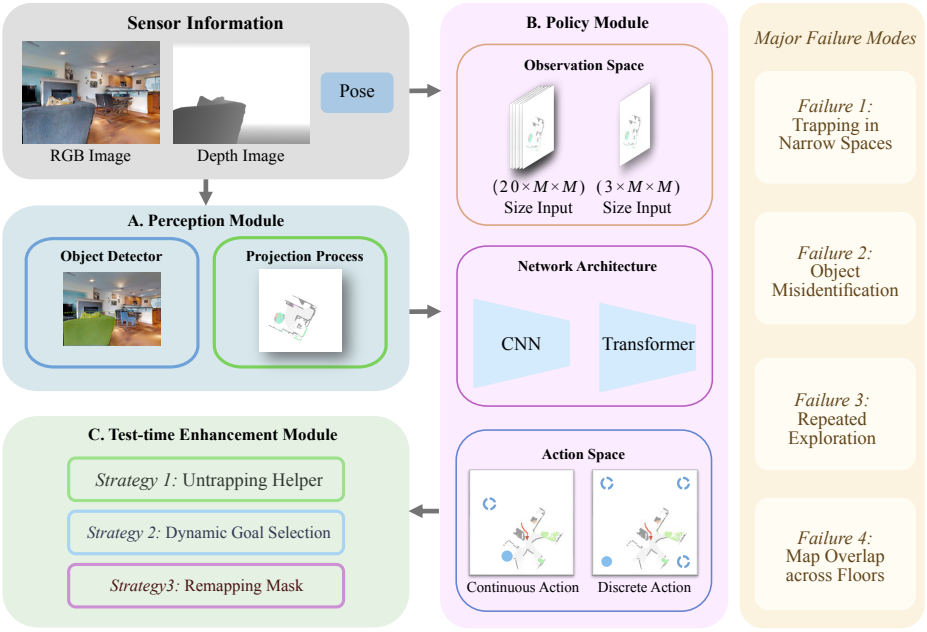


**Fig. 2:** Unified Framework of our experimental setting. *A. Perception:* RGB-D and pose are fused into a top-down semantic map. *B. Policy:* The map and auxiliary inputs (e.g., category, orientation) guide action prediction. *C. Test-Time Enhancement:* Plug-and-play strategies applied at evaluation to boost performance without retraining. The figure also highlights the major failure modes addressed by these strategies.

settings to isolate and quantify their individual contributions to the final navigation performance.

**Perception Module Design Choices.** A typical representation built by the perception module is a top-down semantic map, which captures both geometry and semantic information based on historical input images. At each timestep, an object detector extracts semantic labels from RGB images, which are then fused with depth to build a structured 3D representation. Aggregating this representation along the height axis yields a 2D semantic map suitable for navigation.

In our study, we focus on three widely adopted design choices for the perception module—*object detector*, *map augmentation*, and *map size*, as these have been repeatedly emphasized in prior ObjectNav systems [5, 6, 35, 42, 46]. A detailed description of these components is provided in the Supplementary Materials.

**Policy Module Design Choices.** With the top-down semantic map in place, the policy module infers a goal location that guides the agent's navigation. Typical RL-based policy for long-term goal prediction and identification consists of four key design choices: the *observation space*, the *action space*, the *network architecture*, and the *reward design*. Each of these components influences how the agent interprets the semantic map and decides on navigation strategies. In our experiments, we adopt the common setup established in prior work [5,6,32,35,42].

We refer the readers to the Supplementary Materials for the implementation details.

**Policy Learning Algorithm.** We adopt PPO [27] as the default learning algorithm for the policy module because it is the dominant choice in prior ObjectNav work and consistently offers stable performance under sparse rewards and long-horizon navigation. This observation is consistent with broader trends in robotics RL, where PPO is the dominant choice across quadrupedal locomotion [13, 16], agile aerial navigation [36, 37], and humanoid control [20] due to its reliability and ease of tuning. To illustrate how policy-learning choices affect performance, we also include a small ablation on PPO's clipping threshold in the Supplementary Materials. The standard setting ($\epsilon = 0.2$) clearly outperforms an extremely restrictive value ($\epsilon = 0.001$), which severely limits learning progress. This highlights the importance of maintaining a reasonable update range for effective policy optimization.

**Test-time Enhancement Module Design Choices.** Common failure modes found during evaluations are categorized into: (a) trapping in narrow spaces, (b) object misidentification, (c) repeated exploration, and (d) map overlap across floors due to undetected staircases (Figure 2). To mitigate these issues, several strategies have been used in previous work, but were not formally documented. We summarize them into three plug-and-play strategies: (i) an *untrapping helper* to prevent the agent from getting stuck, (ii) *dynamic goal selection* to avoid redundant exploration, and (iii) a *remapping mask* to handle multi-floor map overlap. We refer readers to the Supplementary Materials for further details.

## 4 Experiments, Results, and Recommendations

Following our definition of three main modules and different design choices for each one, we organize our experiments by analyzing the contribution of individual design choices to overall navigation performance. For each experiment, we provide a detailed setup and report quantitative results, followed by in-depth interpretation and analysis. Alongside our findings, we offer actionable insights and practical recommendations, which can serve as a foundation for designing effective pipelines in future research.

To evaluate the performance of the proposed methods, we consider two evaluation settings: *Fixed timestep* and *Dynamic timestep*. In the fixed-timestep setting, the agent is allowed to interact with the environment for up to 500 steps. In the dynamic timestep setting, the maximum number of steps is set proportional to the number of steps required by an optimal planner to reach the goal. In both cases, the task is considered successful if the agent is within 1 meter of the target object instance at the end of the episode. Further details are provided in the Supplementary Materials.

We adopt three standard evaluation metrics proposed by [1]: Success Rate (SR): the ratio of successful episodes to the total number of episodes. Success weighted by Path Length (SPL): the ratio between the shortest-path distance and the actual path length taken by the agent, averaged over successful episodes. Distance to Success (DTS): the agent's distance to the target object at the end

of an episode. In the Dynamic timestep setting, we report the corresponding variants: D-SR, D-SPL, and D-DTS.

### 4.1 Effect of Perception Module

To isolate the influence of the perception module itself, we implement two heuristic-based long-term goal policies that use heuristic rules for navigation instead of a neural network policy conditioned on the top-down semantic map. This allows us to evaluate the Perception Module independently of long-term goal policy effects.

(i) *Corner Goal Policy*: This method adopts a similar action prediction strategy as Stubborn [17]. Instead of selecting goals in a deterministic or fixed pattern, we introduce randomness into the action selection process, similar to MOPA [25]. This enables more exploratory behavior in the environment, allowing us to better evaluate the influence of components in the Perception Module under varied navigation conditions.

(ii) *Frontier-Based Policy*: As there is currently no standard implementation of the Frontier-Based Policy [39], we adopt a similar setup to FSE [42], but replace the learned policy with a random action policy. Since FSE already includes strong heuristic mechanisms for frontier selection, randomly selecting a goal from the list of frontiers serves as a reasonable approximation of the classical Frontier Exploration method.

**Object Detector** The object detector takes an RGB or RGB-D image as input and outputs the semantic segmentation of the image. It has a dominant impact on overall performance, as the semantic map depends on its output and it directly serves as input to the policy module. We summarize all the object detectors used in the prior works: *Mask R-CNN with Default Checkpoint*: This variant uses the R50-FPN checkpoint from the Detectron2 model zoo [33] without any domain-specific fine-tuning.

*Mask R-CNN with Gibson-Finetuned Checkpoint*: This checkpoint, originally introduced in PONI [21], is fine-tuned on the Gibson dataset [34] to better adapt to the domain characteristics of indoor navigation environments. *RedNet*: A network specifically designed for indoor RGB-D semantic segmentation. It is fine-tuned on 100K randomly sampled views from the MP3D [3] dataset. This model was first utilized in Stubborn [17]. Additionally, we evaluate two recent object detectors, *YOLOv11* [14] and *RF-DETR* [26], and further explore combining them with the foundation model *SAM2* [24] to obtain segmentation masks through an alternative approach.

**Interpretation.** From Table 1, we observe that both the Corner Goal Policy and Frontier-Based Policy experiments exhibit similar trends. More advanced perception models generally yield better performance. And a fine-tuned object detector consistently leads to an obvious performance boost across all evaluation metrics. Specifically, for the Corner Goal Policy, using a fine-tuned Mask R-CNN model results in a 6.8% increase in Success Rate, a 6.7% improvement in Success weighted by Path Length, and a 0.247m reduction in Distance to Success. Similar improvement occurs when using our Dynamic metrics. Additionally, although RedNet is not trained on Gibson, fine-tuning on indoor scenes also helps

**Table 1:** Study results of perception module for Corner (C) and Frontier (F) goal policies. MRCNN denotes Mask R-CNN with the default checkpoint, while FT-MRCNN denotes Mask R-CNN with the Gibson-finetuned checkpoint.

| Pol. | Det. | Size | Aug. | SR (%) | SPL (%) | DTS (m) | D-SR (%) | D-SPL (%) | D-DTS (m) |
|---|---|---|---|---|---|---|---|---|---|
| C | MRCNN | 480 | ✓ | 77.7 | 40.9 | 1.047 | 61.6 | 38.6 | 1.641 |
| C | RedNet | 480 | ✓ | 80.9 | 43.9 | 0.851 | 65.5 | 41.6 | 1.488 |
| C | RF-DETR-Seg | 480 | ✓ | 77.6 | 44.2 | 0.991 | 64.1 | 42.4 | 1.512 |
| C | RF-DETR+SAM2 | 480 | ✓ | 77.1 | 44.7 | 1.112 | 63.8 | 42.6 | 1.669 |
| C | YOLO11-XL | 480 | ✓ | 76.0 | 42.6 | 1.147 | 62.8 | 40.8 | 1.668 |
| C | FT-MRCNN | 480 | ✓ | 83.8 | 47.6 | 0.800 | 69.5 | 45.3 | 1.425 |
| C | FT-MRCNN | 240 | ✓ | 78.7 | 45.2 | 1.135 | 64.8 | 43.1 | 1.658 |
| C | FT-MRCNN | 480 | ✗ | 83.3 | 44.2 | 0.771 | 63.5 | 41.2 | 1.727 |
| F | MRCNN | 480 | ✓ | 71.2 | 38.5 | 1.428 | 49.1 | 33.5 | 2.086 |
| F | RedNet | 480 | ✓ | 75.0 | 40.9 | 1.114 | 53.1 | 35.7 | 1.876 |
| F | FT-MRCNN | 480 | ✓ | 79.9 | 45.8 | 0.892 | 57.2 | 39.7 | 1.714 |
| F | RF-DETR-Seg | 480 | ✓ | 72.9 | 40.3 | 1.183 | 53.6 | 36.6 | 1.871 |
| F | RF-DETR+SAM2 | 480 | ✓ | 72.8 | 43.7 | 1.416 | 53.6 | 39.2 | 2.006 |
| F | YOLO11-XL | 480 | ✓ | 69.5 | 39.0 | 1.514 | 49.8 | 34.6 | 2.106 |
| F | FT-MRCNN | 240 | ✓ | 79.6 | 43.6 | 1.033 | 61.0 | 40.1 | 1.742 |
| F | FT-MRCNN | 480 | ✗ | 78.0 | 42.5 | 1.000 | 53.4 | 36.3 | 1.791 |

reduce distribution errors. More detailed performance analysis can be found in the Supplementary Materials.

**Recommendation.** Our results highlight that the perception module plays a critical role in the overall navigation pipeline, as errors in object detection directly propagate to the semantic map and subsequently influence policy decisions. This suggests that improving the reliability and domain alignment of perception models can substantially enhance downstream navigation performance. In practice, prioritizing robust perception models and ensuring reasonable alignment between the detector's training data and the deployment environment can provide consistent gains in navigation robustness.

**Map Size** Map size defines the agent's field of view. Larger maps reveal more of the environment, but increase computational cost. To investigate its effect, we experiment with two settings: $240 \times 240$ and $480 \times 480$. We focus on the local ego-centric map, as the global map is only used to update the local map and is not directly used for navigation. Therefore, when we refer to the top-down semantic map in this context, we mean the local top-down semantic map.

**Interpretation.** From Table 1, we observe that for the Corner Goal Policy, a larger top-down semantic map generally yields better performance than a smaller one. However, this trend does not hold for the Frontier-Based Policy. In this setting, the performance differences between the two map sizes are minimal. In fact, for some Dynamic metrics such as D-SR and D-SPL, the smaller map size slightly outperforms the larger one. We find that map size influences policies differently: larger maps encourage the Corner Goal Policy to explore wider areas

and thus improve performance, while the Frontier-Based Policy is less sensitive to map size and can even degrade with higher-resolution maps due to frontier extraction errors. Details are provided in the Supplementary Materials.

**Recommendation.** Our results suggest that the impact of map size depends on how the navigation policy utilizes spatial context. For goal-based policies such as the Corner Goal Policy, larger maps provide a broader spatial context that encourages wider exploration and can improve performance. In contrast, frontier-based policies rely primarily on local frontier extraction and therefore benefit less from increased map resolution. This observation highlights that the optimal map representation should be considered jointly with the policy design rather than treated as an independent system parameter.

**Map Augmentation** Map augmentation typically refers to the techniques applied during the projection from 3D voxels to 2D maps [35, 42, 43]. It often involves fine-grained enhancements that aim to make the top-down map more accurately reflect the real-world scene. To study its impact, we conduct two sets of experiments with and without augmentation.

**Interpretation.** As shown in Table 1, under the standard evaluation setting, both the Corner Goal Policy and the Frontier-based Policy achieve slightly better or comparable performance when map augmentation is applied. However, under our proposed evaluation framework, map augmentation leads to a significant improvement in performance for both policies. We find that map augmentation mainly improves the quality of the constructed maps, enabling richer and more accurate representations within the same number of steps. This leads to clear gains under stricter evaluation, while the benefit diminishes under standard evaluation, where agents have more time to explore (see Supplementary Materials).

**Recommendation.** Our results indicate that map augmentation improves the fidelity of the constructed semantic map, leading to more accurate spatial representations for navigation. This benefit becomes particularly evident under stricter evaluation settings, where the agent must build reliable scene representations within a limited number of steps. Under more relaxed settings, where the agent has sufficient time to explore and correct mapping errors, the impact of map augmentation becomes less pronounced. These findings suggest that map augmentation is especially valuable in scenarios requiring efficient exploration and rapid environment understanding.

### 4.2 Effect of Policy Module

The policy module plays a central role in predicting goals for the agent to reach, and its decision-making process directly influences the agent's ability to explore the environment effectively. To better isolate the contribution of different components within the policy module, we fix the parameters of the perception module. Additionally, when analyzing the effect of a single component, we keep all other factors constant.

**Table 2:** Study results of key architectural and training choices, including observation/action spaces, network architectures, and reward design.

| Observation Space | Action Space | Network Arch. | Reward Design | SR (%) | SPL (%) | DTS (m) | D-SR (%) | D-SPL (%) | D-DTS (m) |
|---|---|---|---|---|---|---|---|---|---|
| Standard | Discrete | Transformer | Type 2 | 83.4 | 43.8 | 0.717 | 66.3 | 41.0 | 1.501 |
| Compressed | Discrete | Transformer | Type 2 | 83.8 | 43.3 | 0.685 | 65.8 | 40.3 | 1.572 |
| Compressed | Continuous | Transformer | Type 2 | 84.4 | 48.2 | 0.741 | 69.6 | 45.8 | 1.415 |
| Compressed | Discrete | CNN | Type 2 | 83.7 | 43.3 | 0.706 | 65.8 | 40.5 | 1.548 |
| Compressed | Discrete | Transformer | Type 1 | 85.2 | 45.8 | 0.664 | 67.8 | 43.0 | 1.346 |

**Observation Space** To investigate the impact of different observation space choices on performance, we consider two options: *Top-down semantic map (standard)*: Following most prior work, this setup uses multiple channels to store different types of information, including the exploration map, obstacle map, frontier map, and semantic maps. *Top-down semantic map (compressed)*: In this setting, we propose to use a compressed version of the top-down semantic map. Specifically, we compress the original 20 channel semantic map into a single 3-channel RGB image.

**Interpretation.** Table 2 shows that the performance differences between the standard and compressed configurations are minimal: less than 0.5% in SR and D-SR, less than 0.7% in SPL and D-SPL, and less than 0.1m in DTS and D-DTS. These negligible differences suggest that the compressed top-down semantic map configuration retains sufficient spatial information for effective policy learning, while reducing the observation size by more than a factor of six.

**Recommendation.** Our results indicate that compressed observations preserve the spatial information required for policy learning while significantly reducing the observation dimensionality, suggesting that compact representations can improve efficiency without degrading navigation performance.

**Action Space** The action space determines how the policy generates goals on the map. We investigate the impact of different action space designs by considering the following two options: *Continuous Action Space*: The policy predicts a goal location directly on the map. The goal can be any coordinate within the map boundaries. *Discrete Action Space*: A set of possible goal locations—typically the four corners of the map—is predefined as candidate actions. The policy then predicts the index of one of these predefined goal locations.

**Interpretation.** The 2nd and 3rd row of Table 2 shows that continuous action space outperforms the discrete one. Compared with the continuous action space, the discrete action space is more akin to predicting a direction for the agent rather than an exact goal. Given the scale of the map, the agent's movement range during local policy navigation is limited and often insufficient to directly reach the final goal. As a result, the selected goal in the discrete setting effectively serves as a directional signal, guiding the agent toward further exploration. Therefore, the discrete action space is coarser compared to the continuous action space, as it provides less precise control over goal selection.

**Recommendation.** Without additional strategies to further enhance the discrete action space, the continuous action space is a more favorable choice.

**Network Architecture** Previous works commonly used CNNs to process the observations, more recent studies have shifted to using Transformers. Specifically, in our study, we choose: *CNN*: For the CNN-based model, we adopt ResNet-18 [12] as the policy backbone. *Transformer*: For our transformer model, we adopt the ViT-Base architecture [31] as the backbone for map feature extraction.

**Interpretation.** The results in Table 2 indicate no significant difference in performance between the two network architectures. These findings indicate that, when keeping other modules unchanged, varying the network architecture for map feature extraction does not substantially affect the final outcomes. This also indicates that the overall performance is sensitive to the policy network as other modules.

**Recommendation.** Our results show no significant performance difference between CNN and Transformer backbones for map feature extraction. This may be because the structured top-down semantic map already encodes much of the spatial information required for navigation, reducing the need for highly expressive feature extractors. Consequently, lightweight CNN backbones can offer a more efficient alternative without noticeable performance degradation.

**Reward Design** Reward design is always a critical challenge for reinforcement learning algorithms. In this study, We consider two types of reward designs: $r_1 = r_{\text{exploration}} + r_{\text{distance to target}}$, and $r_2 = r_{\text{exploration}} + r_{\text{success}} + r_{\text{step penalty}}$.

**Interpretation.** Table 2 shows that the policy trained with $r_1$ outperforms that trained with $r_2$, yielding a 0.8% improvement in Success Rate and a 2.5% improvement in SPL. With $r_1$, the agent is rewarded for moving closer to the target object, with semantic information implicitly embedded in the reward signal. In contrast, $r_2$ is designed solely for exploration, where only task success is rewarded and additional steps are penalized; the agent receives no reward for approaching the target.

**Recommendation.** Our results suggest that incorporating a distance-to-target component in the reward function provides a denser and more informative learning signal during policy optimization. Compared to purely exploration-driven rewards, this intermediate progress signal helps reduce ambiguity in the navigation objective and encourages the policy to align exploration with goal-directed behavior. These findings highlight the importance of reward structures that provide meaningful progress signals in long-horizon navigation tasks.

### 4.3 Effect of Test-time Enhancement Module

To improve robustness during deployment, many ObjectNav systems incorporate simple test-time strategies alongside the learned policy. Although these tech-

**Table 3:** Study results of test-time enhancement strategies, including untrapping helper, dynamic goal selection and remapping mask.

| Policy | Untrapping Helper | Dynamic Goal Selection | Remapping Mask | SR (%) | SPL (%) | DTS (m) | D-SR (%) | D-SPL (%) | D-DTS (m) |
|---|---|---|---|---|---|---|---|---|---|
| RL (Continuous) | ✗ | ✓ | ✓ | 81.3 | 47.3 | 0.741 | 69.1 | 45.3 | 1.509 |
| RL (Continuous) | ✓ | ✓ | ✗ | 82.6 | 48.0 | 0.817 | 70.1 | 45.9 | 1.384 |
| RL (Continuous) | ✓ | ✓ | ✓ | 84.4 | 48.2 | 0.741 | 69.6 | 45.8 | 1.415 |
| RL (Discrete) | ✓ | ✗ | ✓ | 83.8 | 43.3 | 0.685 | 65.8 | 40.3 | 1.572 |
| RL (Discrete) | ✓ | ✓ | ✓ | 85.3 | 47.5 | 0.632 | 69.8 | 44.9 | 1.397 |

niques are widely implemented in practice, they are rarely discussed or systematically analyzed in prior work, as they are often treated as engineering details rather than research contributions. In this section, we summarize several commonly used strategies and investigate their impact on navigation performance through controlled experiments.

**Untrapping helper.** The untrapping helper uses predefined rules to help the agent escape from stuck situations. To assess its impact, we compare our continuous-action RL policy with and without the helper enabled.

**Interpretation.** As shown in Table 3, the use of the untrapping helper clearly improves performance in both evaluation settings. The improvement is more obvious in the standard evaluation. Furthermore, we observe that the untrapping helper has a greater impact on long-distance navigation. When the target goal is farther from the agent, the likelihood of the agent getting trapped increases due to the extended navigation path. This explains why the performance improvement is more pronounced in the standard evaluation setting.

**Recommendation.** Our results show that simple recovery mechanisms can significantly improve navigation robustness. In particular, the untrapping helper helps the agent recover from local navigation failures without modifying the learned policy.

**Dynamic goal selection** Dynamic goal selection is a strategy originally designed for heuristic discrete policies to prevent the agent from engaging in redundant exploration. We further demonstrate that it can also be incorporated into RL-based discrete action space policies to guide the local policy more efficiently. To assess its impact, we compare the performance of our trained discrete policy with and without this mechanism.

**Interpretation.** As shown in Table 3, the improvement provided by dynamic goal selection is significant. In both evaluation settings, we observe substantial performance gains. This is primarily because dynamic goal selection effectively reduces repeated exploration, thereby enhancing navigation efficiency.

**Recommendation.** Our analysis shows that dynamic goal selection reduces repeated exploration by discouraging the agent from revisiting previously explored regions. This highlights the benefit of lightweight exploration management strategies alongside learned policies.

| Method | SR(%) | SPL(%) | DTS(m) |
|---|---|---|---|
| SemExp [5] | 65.7 | 33.9 | 1.47 |
| SemExp* [21] | 71.7 | 39.6 | 1.39 |
| PONI [21] | 73.6 | 41.0 | 1.25 |
| 3D-aware [46] | 74.5 | 42.1 | 1.16 |
| FSE [42] | 71.5 | 36.0 | 1.35 |
| SGM [49] | 78.0 | 44.0 | 1.11 |
| L3MVN [43] | 76.8 | 38.8 | 1.01 |
| NaviFormer [35] | 82.6 | 40.9 | 0.76 |
| T-Diff [44] | 79.6 | 44.9 | 1.00 |
| HOZ++ [48] | 78.2 | 44.9 | 1.13 |
| Ours | 84.4 | 48.2 | 0.74 |
| Human Experts | 98.0 | 53.3 | 0.26 |

**Table 4:** Comparisons with prior methods on Gibson. The metrics of the two best-performing methods are highlighted in green and orange. Our method outperforms all prior approaches.

**Remapping mask** The remapping mask assists the agent in regenerating the map when overlapping issues occur during navigation. To evaluate the effectiveness of this mechanism, we perform an ablation study using our trained continuous action space policy.

**Interpretation.** Similar to the effect of the untrapping helper, the remapping mask has a more significant impact in the standard evaluation setting, as shown in Table 3, compared to our more stringent setting. This is because long distance navigation increases the likelihood of the agent encountering map overlap issues, such as revisiting previously seen areas from different floors (e.g., via stairs). In contrast, during short distance navigation tasks, if the agent can reach the target object within just a few steps, it is unlikely to traverse into stair regions that could cause overlaps in the top-down semantic map. As a result, the benefit of the remapping mask in these short scenarios is minimal.

**Recommendation.** Our results suggest that the remapping mask improves robustness when map inconsistencies occur, such as during multi-floor navigation. Maintaining reliable map representations is therefore important for stable navigation.

### 4.4 Comparisons with the State-of-the-art and Human experts

After clearly understanding the impact of different components from various modules on overall performance, we selected four policies, each configured with the best-performing components, to compare against prior work. Standard evaluation metrics were used for this comparison.

To enable comparison with state-of-the-art methods, we design four customized policies: (i) Corner Goal Policy, (ii) Frontier-Based Policy, (iii) RL policy with a discrete action space, and (iv) RL policy with a continuous action

space. The detailed results are shown in the supplementary materials. We report in Table 4 the results from the policies using a continuous action space. For all policies, we adopt the best-performing configurations across all modules. From Table 4, we observe that after thoroughly analyzing the impact of different components, our configured policy outperforms previous state-of-the-art algorithms on the Gibson benchmark. Since modular methods consist of multiple components that jointly contribute to overall performance, it is essential to understand the impact of each component. Without a clear analysis of these components, focusing solely on designing complex policy networks is unlikely to achieve optimal performance. Moreover, complex policy designs often introduce additional challenges, such as increased computational cost and reduced inference speed. Hence, a clear understanding of modular components is essential and can provide valuable insights for future research.

To better contextualize system performance, we introduce a human expert baseline under the same training setup, test environments, and observation modalities as our agents. Five researchers with robotics expertise (but no prior ObjectNav experience) were given practice on the training scenes before evaluation. As shown in Table 4, they achieved near-perfect performance, with an average SR of 98%, consistently outperforming all existing autonomous methods. This highlights the substantial *algorithmic* gap between human and system performance, underscoring the need for more intelligent perception, reasoning, and exploration strategies to approach human-level robustness.

## 5 Conclusion

In this work, we present a large-scale empirical study that systematically analyzes the contributions of different components in RL-based Object-Goal Navigation systems. By decomposing these systems into three key modules—perception, policy, and test-time strategies—we provide a clearer understanding of how different design choices influence navigation performance. Our analysis shows that improvements in perception quality and test-time strategies can lead to substantial gains in navigation robustness and efficiency, while controlled configurations enable more reliable evaluation of policy designs. We further demonstrate that simple test-time strategies, such as dynamic goal selection and untrapping helpers, can significantly improve system robustness without requiring additional training. Guided by these findings, we develop an enhanced modular navigation system that achieves state-of-the-art performance on standard benchmarks. In addition, our human baseline evaluation highlights the significant gap between current autonomous agents and human-level navigation. Beyond empirical improvements, this study provides practical insights and design recommendations for modular ObjectNav systems, offering a foundation for future research on more robust evaluation, improved system design, and real-world deployment of embodied navigation agents.

## References


1. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., Zamir, A.R.: On evaluation of embodied navigation agents (2018), https://arxiv.org/abs/1807.06757
2. Cao, Y., Zhang, J., Yu, Z., Liu, S., Qin, Z., Zou, Q., Du, B., Xu, K.: Cognav: Cognitive process modeling for object goal navigation with llms. In: 2025 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9550–9560 (2025). https://doi.org/10.1109/ICCV51701.2025.00891
3. Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., Zhang, Y.: Matterport3d: Learning from rgb-d data in indoor environments. International Conference on 3D Vision (3DV) (2017)
4. Chaplot, D.S., Dalal, M., Gupta, S., Malik, J., Salakhutdinov, R.R.: Seal: Self-supervised embodied active learning using exploration and 3d consistency. Advances in neural information processing systems **34**, 13086–13098 (2021)
5. Chaplot, D.S., Gandhi, D., Gupta, A., Salakhutdinov, R.: Object goal navigation using goal-oriented semantic exploration. In: In Neural Information Processing Systems (2020)
6. Chaplot, D.S., Gandhi, D., Gupta, S., Gupta, A., Salakhutdinov, R.: Learning to explore using active neural slam. In: International Conference on Learning Representations (ICLR) (2020)
7. Du, H., Li, L., Huang, Z., Yu, X.: Object-goal visual navigation via effective exploration of relations among historical navigation states. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2563–2573 (2023)
8. Ehsani, K., Gupta, T., Hendrix, R., Salvador, J., Weihs, L., Zeng, K.H., Singh, K.P., Kim, Y., Han, W., Herrasti, A., Krishna, R., Schwenk, D., VanderBilt, E., Kembhavi, A.: Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. In: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16238–16250 (2024). https://doi.org/10.1109/CVPR52733.2024.01537
9. Gadre, S.Y., Ehsani, K., Song, S., Mottaghi, R.: Continuous scene representations for embodied ai. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 14849–14859 (2022)
10. Gervet, T., Chintala, S., Batra, D., Malik, J., Chaplot, D.S.: Navigating to objects in the real world. Science Robotics **8**(79), eadf6991 (2023). https://doi.org/10.1126/scirobotics.adf6991, https://www.science.org/doi/abs/10.1126/scirobotics.adf6991
11. Guo, J., Lu, Z., Wang, T., Huang, W., Liu, H.: Object goal visual navigation using semantic spatial relationships. In: CAAI international conference on artificial intelligence. pp. 77–88. Springer (2021)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
13. Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., Hutter, M.: Learning agile and dynamic motor skills for legged robots. Science Robotics **4**(26), eaau5872 (2019)
14. Jocher, G., Qiu, J., Liu, M., Lyu, S., Akyon, F.C., Kalfaoglu, M.E.: Ultralytics YOLO26: Unified Real-Time End-to-End Vision Models (2026). https://doi.org/10.48550/arXiv.2606.03748, https://arxiv.org/abs/2606.03748

15. Khandelwal, A., Weihs, L., Mottaghi, R., Kembhavi, A.: Simple but effective: Clip embeddings for embodied ai. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2022)
16. Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., Hutter, M.: Learning quadrupedal locomotion over challenging terrain. Science robotics **5**(47), eabc5986 (2020)
17. Luo, H., Yue, A., Hong, Z.W., Agrawal, P.: Stubborn: A strong baseline for indoor object navigation. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3287–3293 (2022). https://doi.org/10.1109/IROS47612.2022.9981646
18. Puig, X., Undersander, E., Szot, A., Cote, M.D., Yang, T.Y., Partsey, R., Desai, R., Clegg, A.W., Hlavac, M., Min, S.Y., Vondruš, V., Gervet, T., Berges, V.P., Turner, J.M., Maksymets, O., Kira, Z., Kalakrishnan, M., Malik, J., Chaplot, D.S., Jain, U., Batra, D., Rai, A., Mottaghi, R.: Habitat 3.0: A co-habitat for humans, avatars and robots (2023), https://arxiv.org/abs/2310.13724
19. Qiu, Y., Pal, A., Christensen, H.I.: Learning hierarchical relationships for object-goal navigation. In: The Conference on Robotic Learning, (CoRL). Boston, MA (November 2020), https://proceedings.mlr.press/v155/pal21a/pal21a.pdf
20. Radosavovic, I., Xiao, T., Zhang, B., Darrell, T., Malik, J., Sreenath, K.: Real-world humanoid locomotion with reinforcement learning. Science Robotics **9**(89), eadi9579 (2024)
21. Ramakrishnan, S.K., Chaplot, D.S., Al-Halah, Z., Malik, J., Grauman, K.: Poni: Potential functions for objectgoal navigation with interaction-free learning (2022), https://arxiv.org/abs/2201.10029
22. Ramrakhya, R., Batra, D., Wijmans, E., Das, A.: Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 17896–17906 (2023). https://doi.org/10.1109/CVPR52729.2023.01716
23. Ramrakhya, R., Undersander, E., Batra, D., Das, A.: Habitat-web: Learning embodied object-search strategies from human demonstrations at scale. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5163–5173 (2022). https://doi.org/10.1109/CVPR52688.2022.00511
24. Ravi, N., Gabeur, V., Hu, Y.T., Hu, R., Ryali, C., Ma, T., Khedr, H., Rädle, R., Rolland, C., Gustafson, L., Mintun, E., Pan, J., Alwala, K.V., Carion, N., Wu, C.Y., Girshick, R., Dollár, P., Feichtenhofer, C.: Sam 2: Segment anything in images and videos. arXiv preprint arXiv:2408.00714 (2024), https://arxiv.org/abs/2408.00714
25. Raychaudhuri, S., Campari, T., Jain, U., Savva, M., Chang, A.X.: Mopa: Modular object navigation with pointgoal agents. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5763–5773 (2024)
26. Robinson, I., Robicheaux, P., Popov, M., Ramanan, D., Peri, N.: Rf-detr: Neural architecture search for real-time detection transformers (2025), https://arxiv.org/abs/2511.09554
27. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms (2017), https://arxiv.org/abs/1707.06347
28. Shah, H., Xing, J., Messikommer, N., Sun, B., Pollefeys, M., Scaramuzza, D.: Foresightnav: Learning scene imagination for efficient exploration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. pp. 5275–5284 (June 2025)

29. Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., Batra, D.: Habitat 2.0: Training home assistants to rearrange their habitat (2022), https://arxiv.org/abs/2106.14405
30. Wijmans, E., Kadian, A., Morcos, A.S., Lee, S., Essa, I., Parikh, D., Savva, M., Batra, D.: Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In: International Conference on Learning Representations (2019), https://api.semanticscholar.org/CorpusID:210839350
31. Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., Tomizuka, M., Gonzalez, J., Keutzer, K., Vajda, P.: Visual transformers: Token-based image representation and processing for computer vision (2020)
32. Wu, Q., Wang, J., Liang, J., Gong, X., Manocha, D.: Image-goal navigation in complex environments via modular learning. IEEE Robotics and Automation Letters **7**(3), 6902–6909 (2022)
33. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
34. Xia, F., R. Zamir, A., He, Z.Y., Sax, A., Malik, J., Savarese, S.: Gibson env: real-world perception for embodied agents. In: Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on. IEEE (2018)
35. Xie, W., Jiang, H., Zhu, Y., Qian, J., Xie, J.: Naviformer: A spatio-temporal context-aware transformer for object navigation. Proceedings of the AAAI Conference on Artificial Intelligence **39**(14), 14708–14716 (Apr 2025). https://doi.org/10.1609/aaai.v39i14.33612, https://ojs.aaai.org/index.php/AAAI/article/view/33612
36. Xing, J., Geles, I., Song, Y., Aljalbout, E., Scaramuzza, D.: Multi-task reinforcement learning for quadrotors. IEEE Robotics and Automation Letters (2024)
37. Xing, J., Romero, A., Bauersfeld, L., Scaramuzza, D.: Bootstrapping reinforcement learning with imitation for vision-based agile flight. arXiv preprint arXiv:2403.12203 (2024)
38. Yadav, K., Ramrakhya, R., Majumdar, A., Berges, V.P., Kuhar, S., Batra, D., Baevski, A., Maksymets, O.: Offline visual representation learning for embodied navigation (2022), https://arxiv.org/abs/2204.13226
39. Yamauchi, B.: A frontier-based approach for autonomous exploration. Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation' pp. 146–151 (1997), https://api.semanticscholar.org/CorpusID:206561621
40. Yang, W., Wang, X., Farhadi, A., Gupta, A., Mottaghi, R.: Visual semantic navigation using scene priors (2018), https://arxiv.org/abs/1810.06543
41. Ye, J., Batra, D., Das, A., Wijmans, E.: Auxiliary tasks and exploration enable objectgoal navigation. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 16097–16106 (2021). https://doi.org/10.1109/ICCV48922.2021.01581
42. Yu, B., Kasaei, H., Cao, M.: Frontier semantic exploration for visual target navigation. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). p. 4099–4105. IEEE (May 2023). https://doi.org/10.1109/icra48891.2023.10161059, http://dx.doi.org/10.1109/ICRA48891.2023.10161059
43. Yu, B., Kasaei, H., Cao, M.: L3mvn: Leveraging large language models for visual target navigation. In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). p. 3554–3560. IEEE (Oct 2023). https:

//doi.org/10.1109/iros55552.2023.10342512, http://dx.doi.org/10.1109/IROS55552.2023.10342512
44. Yu, X., Zhang, S., Song, X., Qin, X., Jiang, S.: Trajectory diffusion for objectgoal navigation. In: Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., Zhang, C. (eds.) Advances in Neural Information Processing Systems. vol. 37, pp. 110388–110411. Curran Associates, Inc. (2024), https://proceedings.neurips.cc/paper_files/paper/2024/file/c72861451d6fa9dfa64831102b9bb71a-Paper-Conference.pdf
45. Zeng, K.H., Zhang, Z., Ehsani, K., Hendrix, R., Salvador, J., Herrasti, A., Girshick, R., Kembhavi, A., Weihs, L.: Poliformer: Scaling on-policy rl with transformers results in masterful navigators. In: Agrawal, P., Kroemer, O., Burgard, W. (eds.) Proceedings of The 8th Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 270, pp. 408–432. PMLR (06–09 Nov 2025), https://proceedings.mlr.press/v270/zeng25a.html
46. Zhang, J., Dai, L., Meng, F., Fan, Q., Chen, X., Xu, K., Wang, H.: 3d-aware object goal navigation via simultaneous exploration and identification. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6672–6682 (2023). https://doi.org/10.1109/CVPR52729.2023.00645
47. Zhang, S., Song, X., Bai, Y., Li, W., Chu, Y., Jiang, S.: Hierarchical object-to-zone graph for object navigation. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 15110–15120 (2021). https://doi.org/10.1109/ICCV48922.2021.01485
48. Zhang, S., Song, X., Yu, X., Bai, Y., Guo, X., Li, W., Jiang, S.: Hoz++: Versatile hierarchical object-to-zone graph for object navigation. IEEE Transactions on Pattern Analysis and Machine Intelligence pp. 1–18 (2025). https://doi.org/10.1109/TPAMI.2025.3552987
49. Zhang, S., Yu, X., Song, X., Wang, X., Jiang, S.: Imagine before go: Self-supervised generative map for object goal navigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 16414–16425 (June 2024)
50. Zhou, K., Guo, C., Zhang, H.: Visual navigation via reinforcement learning and relational reasoning. In: 2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI). pp. 131–138 (2021). https://doi.org/10.1109/SWC50871.2021.00027