2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

TriHelper: Zero-Shot Object Navigation with Dynamic Assistance

Lingfeng Zhang[1], Qiang Zhang[1,2], Hao Wang[1], Erjia Xiao[1], Zixuan Jiang[1], Honglei Chen[1], Renjing Xu[1,†]

***Abstract*— Navigating toward specific objects in unknown environments without additional training, known as Zero-Shot object navigation, poses a significant challenge in the field of robotics, which demands high levels of auxiliary information and strategic planning. Traditional works have focused on holistic solutions, overlooking the specific challenges agents encounter during navigation such as collision, low exploration efficiency, and misidentification of targets. To address these challenges, our work proposes TriHelper, a novel framework designed to assist agents dynamically through three primary navigation challenges: collision, exploration, and detection. Specifically, our framework consists of three innovative components: (i) Collision Helper, (ii) Exploration Helper, and (iii) Detection Helper. These components work collaboratively to solve these challenges throughout the navigation process. Experiments on the Habitat-Matterport 3D (HM3D) and Gibson datasets demonstrate that TriHelper significantly outperforms all existing baseline methods in Zero-Shot object navigation, showcasing superior success rates and exploration efficiency. Our ablation studies further underscore the effectiveness of each helper in addressing their respective challenges, notably enhancing the agent's navigation capabilities. By proposing TriHelper, we offer a fresh perspective on advancing the object navigation task, paving the way for future research in the domain of Embodied AI and visual-based navigation.**

I. INTRODUCTION

Navigating in unknown environments to find a specified target object is a significant challenge in Embodied AI research. The Habitat platform[1], developed by Facebook AI Research (FAIR), provides a sophisticated simulation environment for this purpose, enabling the testing of AI agents in complex 3D indoor scenes at speeds surpassing real-time. This makes Habitat an ideal platform for tasks like the indoor Object Goal Navigation (ObjectNav) benchmark, which assesses the ability of agents to locate specific objects (e.g., bed, TV monitor) within multi-storey 3D indoor scenes using only RGBD camera captures and global pose data. In recent years, to advance ObjectNav, researchers have developed a plethora of 3D scene datasets and navigation methodologies, including reinforcement learning[2, 3, 4], imitation learning[5, 6], Zero-Shot learning[7, 8, 9], and Few-Shot learning[10], each with its specific focus and limitations. In end-to-end training, especially with reinforcement learning, researchers have designed various rewards and penalties for training on scene datasets.

Imitation learning significantly improves task success by teaching agents to navigate in the manner humans search for objects, though it requires extensive training and human demonstrations of the training dataset. Zero-Shot and Few-Shot methods, which represent training with a very small portion of the dataset or without any dataset at all, offer noticeable advantages in deployability and adaptability to different scenes, despite a slight decrease in accuracy compared to fully trained models[8]. However, the application of

Fig. 1: Challenges in Zero-Shot Object Navigation.

Zero-Shot learning in ObjectNav has encountered significant challenges. Agents often struggle with efficiently navigating towards the target object without prior exposure to similar environments, leading to issues such as frequent collisions, inefficient exploration paths, and inaccurate target identification.

In this study, we conduct a comprehensive analysis of the primary difficulties faced by agents in Zero-Shot navigation tasks like [11], leading to the development of an integrative framework designed to optimize navigation strategies through specific auxiliary modules, thereby enhancing the efficiency of object navigation in unknown multistorey 3D indoor environments. We recognize that while Zero-Shot learning offers the potential for effective navigation without extensive training data, agents still encounter issues such as collision, low exploration efficiency, and misidentification of targets in practical applications, which are shown in **Fig. 1**. To overcome these challenges, our research not only focuses on a singular navigation strategy but also takes a multi-level dynamic approach, considering the variety of situations an agent might encounter in different settings and designing specialized solutions for each specific problem. Our contributions are specific enhancements for each failure case, achieving state-of-the-art (SOTA) performance in Zero-Shot learning for the ObjectNav on the Habitat platform.

Our contributions are summarized as follows:

- **Collision Helper:** We design a clustering algorithm-based auxiliary system after analyzing the agent's motion trajectories and obstacles encountered to address collision issues.
- **Exploration Helper:** We propose a learning mechanism

[1]Authors with The Hong Kong University of Science and Technology (Guangzhou). lzhang819@conncet.hkust-gz.edu.cn
[2]Author with Beijing Innovation Center of Humanoid Robotics Co., Ltd. Jony.Zhang@x-humanoid.com
† is the Corresponding Author. renjingxu@hkust-gz.edu.cn

based on the exploration behavior of agents to tackle the problem of low exploration efficiency.
- **Detection Helper:** We propose a method employing vision-language models (VLMs) to assist in the detection of target objects from the RGB image, enhancing the accuracy of target detection.

Through these innovative approaches, our method not only significantly improves the overall Zero-Shot learning performance of ObjectNav but also provides new perspectives and frameworks for future research, aiming to better address the challenges of visual-based navigation.

II. RELATED WORK

A. Object Navigation

Visual navigation is a critical task for robots, especially in unknown environments. Object Goal Navigation (ObjectNav) focuses on visual navigation within these unknown settings, leveraging semantic priors to enhance a robot's ability to locate objects[12]. Implementations of the ObjectNav often rely on reinforcement learning[2, 3, 4], imitation learning[13], or top-down map predictions[14, 15, 11]. However, these methods are predominantly based on closed dataset research, making them less applicable to different datasets and platforms. To address the challenges of applying the ObjectNav to various datasets and reduce training consumption, recent developments in Zero-Shot ObjectNav frameworks have garnered significant attention. We will also employ Zero-Shot approaches to conduct ObjectNav.

B. Zero-Shot Object Navigation

In direct supervision methods, the necessity of retraining due to variations in datasets when transitioning between them significantly increases the training consumption. Zero-Shot ObjectNav effectively addresses this issue and has seen substantial advancements in recent years. For instance, SemExp[14] constructs the semantic map to explore and navigate. CLIP on Wheels (CoW)[16], by integrating CLIP[17] technology, combines egocentric RGB-D images with verbal instructions. LGX[18], LFG[19], and ESC[20] leverage the common-sense reasoning capabilities of Large Language Models (LLMs) to support sequential navigation decisions. The PixNav[13] project, through the use of foundational models and specifying navigation targets in pixel units, achieves universal navigation for various object types.

However, these methods typically require the conversion of environmental visual information into textual information for target navigation. The application of Visual Language Models (VLMs) offers a novel solution to this issue. VLMs can directly extract semantic information from RGB images and be deployed on consumer-grade laptops, optimizing the processing workflow[8]. Furthermore, combining VLMs with LLMs for knowledge transfer and the utilization of prior information can significantly enhance the accuracy of locating targets. By integrating the capabilities of VLMs and LLMs—that is, using VLMs to directly extract rich semantic prior information from RGB images while employing LLMs to select the optimal navigation goal points we can effectively improve the success rate of navigation. Compared to existing technologies, this integrative approach demonstrates significant advantages in enhancing navigation efficiency and accuracy.

III. PRELIMINARY

A. Problem definition

In the framework of the ObjectNav, the agent is tasked with navigating towards 6 specified target object categories, like a 'chair' or 'bed' (T_i). The agent starts off at a randomly selected spot within the scenario (S_i), with the category of the object (T_i) as its target. At every discrete interval, denoted as time step t, the agent is equipped with visual data (V_t) and readings of its sensor's pose (P_t), guiding its choice of movement commands (a_t). The visual data encompasses imagery from the agent's point of view, including both color (RGB) and depth details. The space of actions ($\mathcal{A}$) available to the agent includes: *move_forward, turn_left, turn_right, look_up, look_down, and stop*. Only the *move_forward* action causes the agent to move 25 cm forward, the other actions only cause the agent to rotate 30 degrees in the corresponding direction, except stop. The agent can opt to stop once it assesses that it has neared the target of interest. Success in an episode is achieved if the agent elects to stop within a predetermined area close to the target object, specifically a distance less than $d_{success}$ (= 0.1 meters). Each episode has a preset limit of 500 time steps, marking the end of an episode.

B. Navigation Map Construction

1) Semantic Map: Semantic map $\mathcal{M}$ is created using RGB-D images and the pose of agents, which is similar to the method in [14]. Its structure is a three-dimensional tensor of size $C \times W \times H$, where $W \times H$ defines the width and length of the semantic map, and C is equal to $C_n + 5$ where n is the number of object categories, representing the number of channels. In this work, The first four channels of the semantic map respectively represent the obstacle map, explored area, current agent location, and past agent location. The next n channels are the semantic maps with n types of objects $O_{1...n}$. And we finally added a false target semantic map to save misidentified targets for later use. At the time of each episode update, we clear the semantic map, substitute and default the starting position of the agent to the center point of the semantic map $(\frac{W}{2}, \frac{H}{2})$. The semantic map is populated by converting visual data into point clouds using geometric techniques, which are then mapped onto a 2D top view. It is characterized by the inclusion of physical obstacles and areas that have been explored, as well as elements of different object species identified by semantic segmentation. The semantic mask matches the point cloud one by one, allowing precise channel mapping on the semantic map. Semantic maps provide the basis for our Zero-Shot ObjectNav.

2) Frontiers Map: After building the semantic map, in order to find the next goal point, we also build the frontier map from the first two channels of the semantic map, using the procedure described in [21]. The process begins with

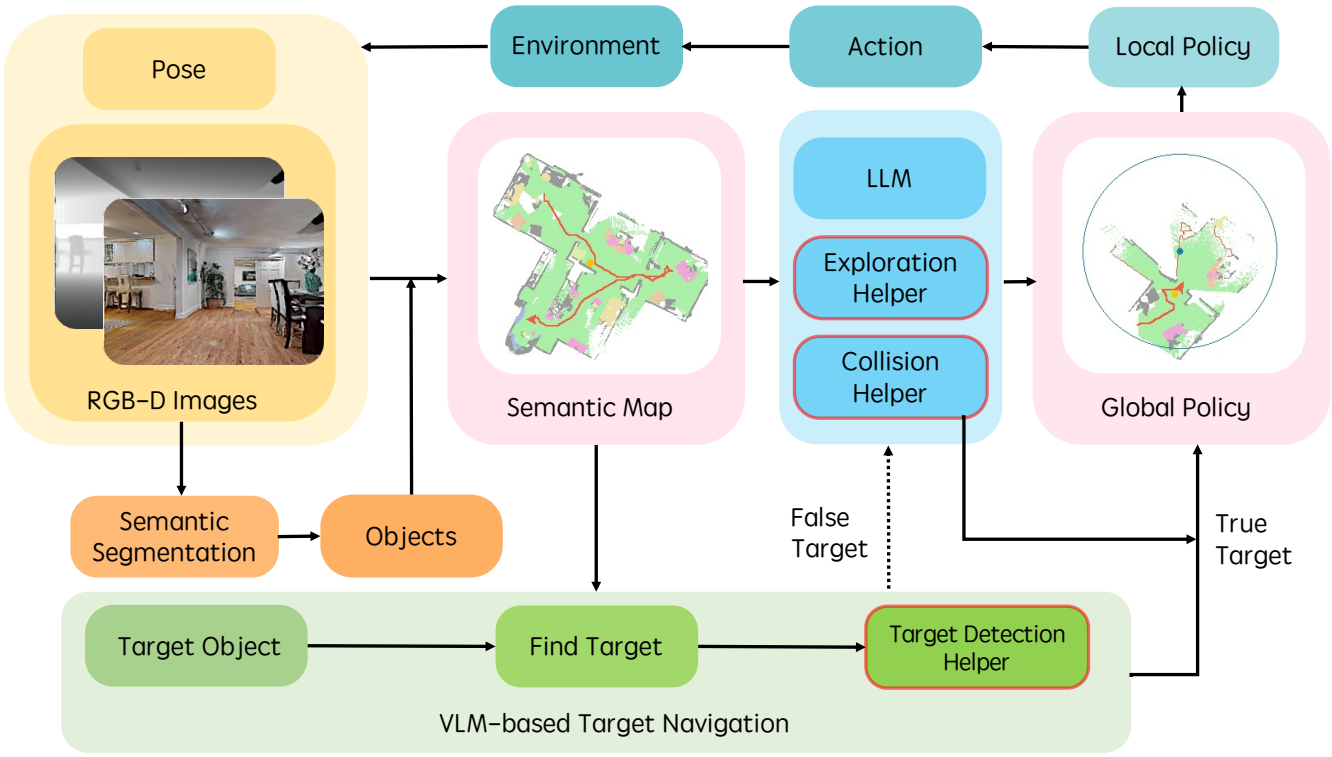

Fig. 2: The architecture of our framework. At each time step, we first input the RGB-D images into the semantic segmentation module to get the object masks, then construct the semantic map and dynamically use the global policy proposed to select a long-term goal. Finally, we use the local policy to get the action of the agent and interact with the environment. The long-term goal point and the center of the largest connected area of the explorable region are marked in the figure. The dotted line represents the re-entry of the exploration process when the target object is false.

delineating the boundaries of the explored map to find the outer edges. Expand the edges of the obstacle map, then create the frontier map by subtracting the obstacle map from the explored area. This map highlights potential areas to explore next. Small areas on the map are identified and removed, leaving only significant areas as potential exploration targets. Thus all the remaining centers of the area can be calculated as our frontiers $\mathcal{F}$. A scoring system based on cost and utility proposed in [22] helps prioritize these frontiers, for each frontier cell $F_i \in \mathcal{F}$, we can calculate the score $S^{CU}(F_i)$:

$$S^{CU}(F_i) = U(F_i) - \lambda_{CU} C(F_i) \tag{1}$$

where $U(F_i)$ is a utility function, $C(F_i)$ is a cost function and the constant λ_{CU} adjusts the relative importance between both factors. Each potential frontier cell in these areas is evaluated to determine its viability as an exploration destination, balancing the cost of reaching the destination with the expected utility.

3) Frontiers Select: We utilize the method described in [7] to score and select from all the frontiers, which builds a query prompt for each frontier point and utilize an LLM to query:

"What is the probability that $O_1, O_2, \cdots, O_j, T_i$ exist in a frontier area at the same time?"

Where $O_{1\ldots j}$ represents all objects contained in the frontiers to which F_i belongs and T_i represents the target object. So we can get the $P(F_i)$ from the LLM response, and we can find the F_{LLM} such that the function $P(F_i)$ reaches its maximum value:

$$F_{LLM} = \underset{F_i}{\operatorname{argmax}}\ P(F_i) \tag{2}$$

Once F_{LLM} is calculated, we utilize it dynamically later in the global policy.

IV. METHODS

A. Overview

The overview of our proposed framework is illustrated in **Fig. 2**. Upon acquiring RGB-D images and the agent's pose from the environment, we input the RGB-D images into the semantic segmentation module, obtaining masks for each object category within the images. These masks, along with the original data, are then fed into the mapping module for semantic map construction. After constructing the semantic map, we utilize a global policy mentioned in IV-C.1 to select the next long-term goal point. After that, we use the local policy for short-term goal navigation mentioned in IV-C.2. At each time step, we compute a new action for the agent to take and interact with the environment, collecting data for the next time step. Our long-term goal points are updated every 10-time steps to ensure that the agent has enough time to move toward the long-term goal. The VLM-based Target Navigation module is activated only when the target object is detected by the semantic segmentation module, to

specifically navigate to the final target object unless the target is false.

B. TriHelper

1) Collision Helper: We define two types of situations that the agent encounters during the ObjectNav as collisions: First, the agent is trapped in a certain position and collides with nearby objects or obstacles several time steps; second, after setting the long-term goal, the local policy planner mentioned in IV-C.2 cannot compute the reachable path, which hinders the agent from further exploration. To solve the collision problem, we propose a cluster-based collision helper. Upon examination, it has been determined that the majority of collision instances stem from the imprudent selection of long-term goals. This often leads to the agent becoming entrapped within confined spaces or failing to identify a navigable path. Therefore, we employ the clustering algorithm to compute the maximum connected space of the navigable area and select the centroid of the maximum connected space as the long-term goal when a collision is detected, so that the agent can reach as broad an area as possible and continue exploring after untrapping. We use the second channel M_2 of the semantic map $\mathcal{M}$ as the navigable area, and the clustering algorithm is as follows:

$$G_{max} = \underset{G_k}{\operatorname{argmax}} \, |G_k| \tag{3}$$

here G_k denotes the set of connected regions into which M_2 is decomposed, and $|G_k|$ denotes the size of the computed connected region G_k. After calculating the maximum connected region G_{max}, we calculate the center of G_{max}:

$$\begin{cases} x_{\text{centroid}} = \dfrac{1}{|G_{\max}|} \displaystyle\sum_{(x_i,y_i)\in G_{\max}} x_i \\ y_{\text{centroid}} = \dfrac{1}{|G_{\max}|} \displaystyle\sum_{(x_i,y_i)\in G_{\max}} y_i \end{cases} \tag{4}$$

the calculated center-of-mass coordinates can be used in the subsequent global policy to help the agent untrap.

2) Exploration Helper: After analyzing the ObjectNav of the agent, we observed that the baseline method [7], which utilizes an LLM for frontier selection, exhibits a prolonged failure to update the long-term goal. This issue arises because the F_{LLM} didn't update for a long time. Therefore, we calculate the distance between the agent's current location and the long-term goal point:

$$d = \sqrt{(x_g - x_a)^2 + (y_g - y_a)^2} \tag{5}$$

where (x_a, y_a) denotes the agent's current location, and (x_g, y_g) denotes the long-term goal point. d is used in the global policy to decide whether to set the LLM to sleep.

3) Detection Helper: Semantic segmentation presents a significant challenge in the ObjectNav, where misclassification of the target object can directly precipitate task failure. Consequently, we integrate a VLM to verify the identified target object. Leveraging its capacity for deeper semantic image analysis through prompt-based guidance, the VLM can execute binary classification of the target object by combining it with other objects within the image. Our prompt is set as:

"Based on all the objects in this picture and the information of this room, judge whether T_i exists in this picture, and output only yes or no."

After that, we can get a response from the VLM and give the auxiliary information to the global policy for navigation.

C. Dynamic Policy

1) Global Policy: Here we introduce our global navigation policy, purposed to find the next long-term goal point in the semantic map to direct the agent's trajectory. We use a dynamic policy to select the long-term goal point. First, we get the best frontier goal point F_{LLM} given by LLM from the baseline policy and use it as one of the candidate goal points. Then we check the agent state at the current time step, if the agent chooses the *move_forward* action and has moved less than 25 cm, or the local policy described later cannot find a feasible path, the agent is put into collision state, and then we use the collision helper to calculate the G_{max} and the $(x_{centroid}, y_{centroid})$ as the long term goal point. If not in collision state, we check if d is less than the $dormant\_thresold$ and F_{LLM} does not change $(F_{LLM\_t} = F_{LLM\_t-1})$, then the exploration helper is enabled to let the LLM sleep for $sleep\_time$ time steps and the agent is free to explore to increase the exploration space. Finally, if the target object is detected by the semantic segmentation module, we use the detection helper to double check the current RGB image. If the target is true, we directly select the detected target object as the final target point and continue to use the collision helper in this period. If the target is false, we mask it out, record it, and let the agent continue to explore. Finally, if the target object is not found after a certain time step $detection\_threshold$ and a false target is recorded, then the false target is marked as the final goal point.

The three modules we proposed collaborate and complement each other in global policy, and their synergy is illustrated in **Fig. 3**, where they are dynamically utilized throughout the exploration and navigation process to better assist our agent in the ObjectNav.

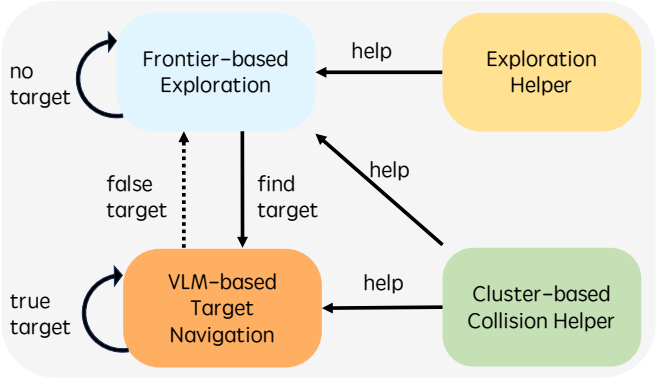

Fig. 3: The framework of dynamic global policy.

2) Local Policy: After finding the long-term goal via the global policy, we deploy the local policy to navigate the agent to the long-term goal point. We use the Fast Marching Method (FMM) [23] for point-to-point navigation configuring the start point at the agent's current locations and the end point at the long-term goal point. FMM calculates the current navigation path at each timestep and then selects the action from the action space based on the agent's current locations, which ensures real-time and effective path planning and navigation. FMM substitutes end-to-end methods such as reinforcement learning, so that our Zero-Shot ObjectNav framework operates without training.

V. EXPERIMENT

In this section, we evaluate the performance of our methodology by comparing it to other Zero-Shot ObjectNav baselines. Additionally, We demonstrate the ability of the three agent helpers to solve three failed cases respectively.

A. Datasets

Our methodology is evaluated using the Habitat Simulator[1] by applying it to the validation segments of two different datasets: HM3D and Gibson. Specifically, the validation segment for HM3D includes 2000 episodes, spread over 20 scenes. For Gibson, we employ the ObjectNav validation segment developed in SemExp[14] within 5 scenes. The two datasets both encompass 6 target object categories: couch, chair, bed, toilet, TV monitor and potted plant.

B. Experiment Details

We evaluate our model on the 3D indoor simulator Habitat platform[1] with an RGB-D image size of (480,640) and an environmental input that also includes the base odometer sensor, a target object represents as an integer of $0-5$. Our implementation is based on Open-Set code in [7]. The LLM[24] used in selecting frontiers is calibrated in alignment with the baseline framework. At the same time, we use RedNet[25] and yolov8[26] to conduct semantic segmentation to predict all existing objects in the RGB-D image. For the VLM used in detection helper, we choose the quantized Qwen-VL-Chat-Int4[27] model to ensure effectiveness while improving our inference speed and reducing memory footprint. For other constants, we set the $dormant\_thresold = 50cm$, $sleep\_time = 20$ and $detection\_thresold = 400$.

C. Evaluation Metrics

We evaluate our method using Success Rate(SR), Success weighted by Path Length (SPL) and Distance to Goal (DTG) based on previous work [28]. SR is defined as: $SR = \frac{1}{N}\sum_{i=1}^{N} S_i$, SPL is defined as: $SPL = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{\max(p_i, l_i)}$, and DTG is defined as: $DTG = max(||x_T - G||_2 - d_s, 0)$. Here N represents the total number of evaluated tasks. S_i denotes a binary success indicator for the task, with 1 indicating success (i.e., the agent reached the goal) and 0 indicating failure. l_i is the shortest path length from the starting point to the goal for the i^{th} episode. p_i is the actual path length traveled by the agent to complete the i^{th} episode. $||x_T - G||_2$ is the L2 distance of the agent from the target object location at the end of the episode, d_s is the success threshold boundary. The SPL metric, therefore, not only considers whether the agent successfully completes the task but also evaluates the efficiency of the path taken by the agent. The efficiency is gauged by comparing the actual path length (p_i) against the shortest possible path length (l_i). Thus, higher values are preferable for SR and SPL. DTG represents the distance between the agent and the target objects at the end of episodes, so a lower value is preferable.

D. Baselines

To evaluate the Zero-Shot ObjectNav performance of our model, we compare it to several baselines containing the state-of-the-art (SOTA) baseline.

- **Randomly Walking:** The agent selects an action randomly from all available actions space $\mathcal{A}$.
- **FBE[9]:** This baseline utilizes a classical robotics framework for constructing maps and selecting frontiers.
- **SemExp[14]:** This baseline constructs the semantic map and utilizes reinforcement learning.
- **ZSON[29]:** This work transfers the agent trained in the Image Navigation (ImageNav) task to the ObjectNav.
- **CoW[16]:** CoW applies CLIP[30] to detect objects and let the agent always explore the closest frontiers until it finds the target object.
- **ESC[20]:** This work uses the semantic map to navigate and CLIP to detect objects, and implemented LLM to select frontiers.
- **L3MVN[7]:** We follow this work as the baseline to construct the semantic map and utilize LLM to select frontiers.
- **PixNav[13]:** This baseline uses pixels as navigation targets, trains models for pixel goal navigation, and uses LLMs for long-term navigation planning.
- **PONI[21]:** This baseline determines the location of an unseen target by predicting two latent functions.
- **Stubborn[11]:** This work proposes methods to address collision and detection.
- **VLFM[8]:** In this latest work, researchers built a semantic value map to evaluate frontiers to select exploration directions.

E. Results

The performance of our model on the two datasets HM3D and Gibson is shown in Table I, with empty data indicating that the work was not experimented on that dataset or did not provide the appropriate metrics. TriHelper clearly stands out from all the Zero-Shot ObjectNav methods and achieves SOTA on both datasets. Compared to the baseline we used, our proposed method achieves +6% SR improvement and +9.5% SPL on the HM3D dataset; +9.1% SR improvement and +9.6% SPL improvement on the Gibson dataset. Compared with the current SOTA model, we achieve +4% SR improvement on the HM3D dataset and +1.2% SR improvement on the Gibson dataset. The SR of our method compared to the baseline by object category is illustrated in **Fig. 4**. The reason why our SPL is slightly lower than that

of the SOTA method is that the proposed dynamic global policy focuses on improving the most important metric SR, which indicates the percentage of agents that can successfully navigate to the target object. Since we prevent the agent from choosing a false target for a certain time step, the path length is increased. In addition, since Gibson's scene dataset is more spacious and the task dataset does not contain episodes on the upper and lower floors, our effectiveness in solving the collision problem is diminished, and a small improvement is achieved. We also found that some detection fail cases were caused by simulator misjudgments, so we also carried out a manual double-check on the HM3D, we can see that after removing the misjudgments, our method achieved 5.9% improvement of baseline and reaches 60%+ SR. See VII Appendix for detailed analysis.

TABLE I: Results of comparative experiment. Our model reaches the SOTA across all the Zero-Shot ObjectNav baselines. We use two color scales of blue to denote the **first** and second best performance. The * denotes that we manually double-check.

Method	Zero-Shot	HM3D			Gibson		
		SR↑	SPL↑	DTG↓	SR↑	SPL↑	DTG↓
Randomly	✓	0.000	0.000	7.600	0.030	0.030	2.580
FBE[9]	✓	0.237	0.123	5.414	0.417	0.214	2.634
SemExp[14]	✗	0.379	0.188	**2.943**	0.652	0.336	1.520
ZSON[29]	✗	0.255	0.126	-	0.313	0.120	-
CoW[16]	✓	0.320	0.181	-	-	-	-
ESC[20]	✓	0.385	0.220	-	-	-	-
PONI[21]	✗	-	-	-	0.736	0.410	1.250
VLFM[8]	✓	0.525	**0.304**	-	0.840	**0.522**	-
PixNav[13]	✗	0.379	0.205	-	-	-	-
Stubborn[11]	✗	0.237	0.098	-	-	-	-
L3MVN[7]	✓	0.504	0.231	4.427	0.761	0.377	1.101
L3MVN[7]*	✓	0.561*	-	-	-	-	-
TriHelper (Ours)	✓	**0.565**	0.253	3.873	**0.852**	0.431	**0.600**
TriHelper (Ours)*	✓	**0.620***	-	-	-	-	-

F. Ablation study

In order to evaluate the capability of the three proposed helpers for the navigation problem. We conduct ablation experiments on the HM3D dataset: the percentage of corresponding failure cases with different helpers, as well as the improvement in SR. The results in Table II show that all three of our proposed helpers have significant effects on the corresponding failure cases: The collision helper reduces 6% of collisions; the exploration helper reduces 2.60% of failed exploration; and the detection helper reduces 3.05% of false detection. The best performance is reached when the three helpers are used together: 11.05% reduction in collision, 2.60% reduction in false detection, and 6% improvement in SR compared to baseline. Table II shows that employing all three helpers simultaneously reduces most collision and many recognition failures, yet exploration failures significantly increase. This issue arises primarily from episodes where the target object is located on a different floor than the agent's initial position, making it challenging for the agent to transition between floors after exploring the initial one. Additionally, the detection helper's masking of false targets can sometimes prevent the agent from successfully locating the target object before the episode ends, contributing to task failure.

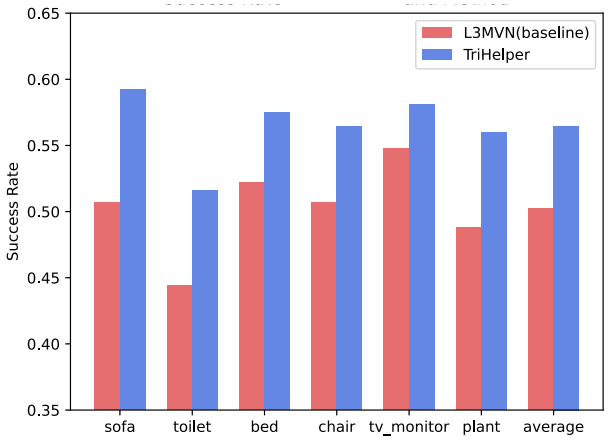

Fig. 4: The Success Rate by Category and Method.

TABLE II: Results of ablation study on HM3D. The $\downarrow$ *and* $\uparrow$ respectively denote the decrease and increase in the number of corresponding failures. The * denotes that we manually double-check.

Ablation	Collision Failure(%)↓	Exploration Failure(%)↓	Detection Failure(%)↓	HM3D Results SR↑
No Helper	19.10	13.65	16.95	0.505
Collision	13.10↓	16.65↓	17.30↑	0.530
Exploration	18.45↓	**11.05**↓	17.35↑	0.531
Detection	20.00↑	12.05↓	**13.90**↓	0.540
TriHelper	**8.05**↓	21.25↑	14.30↓	**0.565**
TriHelper*	**8.05**↓	21.25↑	**8.80***↓	**0.620***

VI. CONCLUSIONS

In this work, we presented TriHelper, a novel framework aimed at addressing the key challenges of Zero-Shot ObjectNav. Our framework substantially enhances the navigational proficiency of autonomous agents operating within unfamiliar environments by dynamically incorporating a trio of synergistic components: Collision Helper, Exploration Helper, and Detection Helper. Each component is meticulously designed to tackle specific problems encountered during navigation, namely collision avoidance, efficient exploration, and accurate target detection.

Our extensive experiments conducted on the HM3D and Gibson datasets have demonstrated the superior performance of TriHelper against all the existing baseline methods. The ablation studies further validated the critical role of each helper component, highlighting their collective importance in enhancing the agent's navigation success.

By integrating these innovative solutions, TriHelper offers a robust and dynamic framework that significantly advances the field of Zero-Shot ObjectNav. Our approach underscores the necessity of targeted assistance and strategic planning in overcoming navigation challenges, highlighting the importance of targeted assistance for overcoming specific navigation challenges. Further research could focus on solving the exploration problem caused by exploring different floors.

REFERENCES

[1] Manolis Savva et al. "Habitat: A platform for embodied ai research". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9339–9347.

[2] Bogdan Mazoure et al. "Improving zero-shot generalization in offline reinforcement learning using generalized similarity functions". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 25088–25101.

[3] Angelos Filos et al. "Psiphi-learning: Reinforcement learning with demonstrations using successor features and inverse temporal difference learning". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 3305–3317.

[4] Matt Deitke et al. "ProcTHOR: Large-Scale Embodied AI Using Procedural Generation". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 5982–5994.

[5] Zhao Mandi et al. "Towards more generalizable one-shot visual imitation learning". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2434–2444.

[6] Ram Ramrakhya et al. "PIRLNav: Pretraining With Imitation and RL Finetuning for ObjectNav". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 17896–17906.

[7] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. "L3mvn: Leveraging large language models for visual target navigation". In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pp. 3554–3560.

[8] Naoki Harrison Yokoyama et al. "Vlfm: Vision-language frontier maps for zero-shot semantic navigation". In: *2nd Workshop on Language and Robot Learning: Language as Grounding*. 2023.

[9] Brian Yamauchi. "A frontier-based approach for autonomous exploration". In: *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*. IEEE. 1997, pp. 146–151.

[10] Vernon Kok, Micheal Olusanya, and Absalom Ezugwu. "A few-shot learning-based reward estimation for mapless navigation of mobile robots using a siamese convolutional neural network". In: *Applied Sciences* 12.11 (2022), p. 5323.

[11] Haokuan Luo et al. "Stubborn: A strong baseline for indoor object navigation". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 3287–3293.

[12] Dhruv Batra et al. "Objectnav revisited: On evaluation of embodied agents navigating to objects". In: *arXiv preprint arXiv:2006.13171* (2020).

[13] Wenzhe Cai et al. "Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill". In: *arXiv preprint arXiv:2309.10309* (2023).

[14] Devendra Singh Chaplot et al. "Object goal navigation using goal-oriented semantic exploration". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4247–4258.

[15] Sixian Zhang et al. "Hierarchical object-to-zone graph for object navigation". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 15130–15140.

[16] Samir Yitzhak Gadre et al. "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*. 2023, pp. 23171–23181.

[17] Alec Radford et al. "Learning transferable visual models from natural language supervision". In: *International conference on machine learning*. PMLR. 2021, pp. 8748–8763.

[18] Vishnu Sashank Dorbala, James F Mullen Jr, and Dinesh Manocha. "Can an Embodied Agent Find Your "Cat-shaped Mug"? LLM-Based Zero-Shot Object Navigation". In: *IEEE Robotics and Automation Letters* (2023).

[19] Dhruv Shah et al. "Navigation with large language models: Semantic guesswork as a heuristic for planning". In: *Conference on Robot Learning*. PMLR. 2023, pp. 2683–2699.

[20] Kaiwen Zhou et al. "Esc: Exploration with soft commonsense constraints for zero-shot object navigation". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 42829–42842.

[21] Santhosh Kumar Ramakrishnan et al. "Poni: Potential functions for objectgoal navigation with interaction-free learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 18890–18900.

[22] Miguel Juliá, Arturo Gil, and Oscar Reinoso. "A comparison of path planning strategies for autonomous exploration and mapping of unknown environments". In: *Autonomous Robots* 33 (2012), pp. 427–444.

[23] James A Sethian. "A fast marching level set method for monotonically advancing fronts." In: *proceedings of the National Academy of Sciences* 93.4 (1996), pp. 1591–1595.

[24] Yinhan Liu et al. "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692* (2019).

[25] Jindong Jiang et al. "Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation". In: *arXiv preprint arXiv:1806.01054* (2018).

[26] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. Version 8.0.0. 2023. URL: https://github.com/ultralytics/ultralytics.

[27] Jinze Bai et al. "Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond". In: (2023).

[28] Peter Anderson et al. "On evaluation of embodied navigation agents". In: *arXiv preprint arXiv:1807.06757* (2018).

[29] Arjun Majumdar et al. "Zson: Zero-shot object-goal navigation using multimodal goal embeddings". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 32340–32352.

[30] Liunian Harold Li et al. "Grounded language-image pre-training". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10965–10975.

VII. APPENDIX

A. *Manually Double-check*

After an in-depth analysis of the results of each episode, we found that some of the failures due to detection were caused by misjudgment of the simulator, as shown in **Fig. 5**. In order to correct these misjudgments, we manually double-check the HM3D experiment results of the TriHelper and baseline for detection cases. The results are shown in the Table III below.

Fig. 5: The misjudgments of the simulator.

TABLE III: Results of manual double-check on HM3D.

Method	Detection Cases/ Corrected Cases	Detection Ratio Corrected(%)	HM3D SR
L3MVN[7]	339/112	33.04	0.561
TriHelper (Ours)	286/110	38.41	**0.620**

Through the table data, we can analyze that there are 33.04%, 38.41% of simulator misjudgment cases respectively in the baseline and our model, and after subtracting these cases, we achieve 0.561, 0.620 of SR respectively. At this time, our model improves 5.91% of SR compared to the baseline. We can see that after adding the detection helper, our model is able to solve the problem of target object misrecognition very well.