3D-Aware Object Goal Navigation via Simultaneous Exploration and Identification

Jiazhao Zhang\textsuperscript{1,2,*} Liu Dai\textsuperscript{3,*} Fanpeng Meng\textsuperscript{4} Qingnan Fan\textsuperscript{5} Xuelin Chen\textsuperscript{5} Kai Xu\textsuperscript{6} He Wang\textsuperscript{1†}

\textsuperscript{1}CFCS, Peking University \textsuperscript{2}Beijing Academy of Artificial Intelligence \textsuperscript{3}CEIE, Tongji University \textsuperscript{4}Huazhong University of Science and Technology \textsuperscript{5}Tencent AI Lab \textsuperscript{6}National University of Defense Technology

Abstract

Object goal navigation (ObjectNav) in unseen environments is a fundamental task for Embodied AI. Agents in existing works learn ObjectNav policies based on 2D maps, scene graphs, or image sequences. Considering this task happens in 3D space, a 3D-aware agent can advance its ObjectNav capability via learning from fine-grained spatial information. However, leveraging 3D scene representation can be prohibitively impractical for policy learning in this floor-level task, due to low sample efficiency and expensive computational cost. In this work, we propose a framework for the challenging 3D-aware ObjectNav based on two straightforward sub-policies. The two sub-policies, namely corner-guided exploration policy and category-aware identification policy, simultaneously perform by utilizing online fused 3D points as observation. Through extensive experiments, we show that this framework can dramatically improve the performance in ObjectNav through learning from 3D scene representation. Our framework achieves the best performance among all modular-based methods on the Matterport3D and Gibson datasets, while requiring (up to 30x) less computational cost for training. The code will be released to benefit the community.\textsuperscript{1}

1. Introduction

As a vital task for intelligent embodied agents, object goal navigation (ObjectNav) \cite{38, 49} requires an agent to find an object of a particular category in an unseen and unmapped scene. Existing works tackle this task through end-to-end reinforcement learning (RL) \cite{27, 36, 47, 51} or modular-based methods \cite{9, 14, 35}. End-to-end RL based methods take as input the image sequences and directly output low-level navigation actions, achieving competitive performance while suffering from lower sample efficiency and poor generalizability across datasets \cite{3, 27}. Therefore, we favor modular-based methods, which usually contain the following modules: a semantic scene mapping module that aggregates the RGBD observations and the outputs from semantic segmentation networks to form a semantic scene map; an RL-based goal policy module that takes as input the semantic scene map and learns to online update a goal location; finally, a local path planning module that drives the agent to that goal. Under this design, the semantic accuracy and geometric structure of the scene map are crucial to the success of object goal navigation.

We observe that the existing modular-based methods mainly construct 2D maps \cite{8, 9}, scene graphs \cite{34, 56} or neural fields \cite{43} as their scene maps. Given that objects lie in 3D space, these scene maps are inevitably deficient in leveraging 3D spatial information of the environment comprehensively and thus have been a bottleneck for further improving object goal navigation. In contrast, forming a 3D scene representation naturally offers more accurate, spatially dense and consistent semantic predictions than its 2D counterpart, as proved by \cite{12, 31, 45}. Hence, if the agent could take advantage of the 3D scene understanding and

![Figure 1. We present a 3D-aware ObjectNav framework along with simultaneous exploration and identification policies: A→B, the agent was guided by an exploration policy to look for its target; B→C, the agent consistently identified a target object and finally called STOP.](image-url)
form a 3D semantic scene map, it is expected to advance the performance of ObjectNav.

However, leveraging 3D scene representation would bring great challenges to ObjectNav policy learning. First, building and querying fine-grained 3D representation across a floor-level scene requires extensive computational cost, which can significantly slow down the training of RL [7,55]. Also, 3D scene representation induces considerably more complex and high-dimensional observations to the goal policy than its 2D counterpart, leading to a lower sample efficiency and hampering the navigation policy learning [22,57]. As a result, it is demanding to design a framework to efficiently and effectively leverage powerful 3D information for ObjectNav.

To tackle these challenges, we propose a novel framework composed of an online semantic point fusion module for 3D semantic scene mapping and two parallel policy networks in charge of scene exploration and object identification, along with a local path planning module. Our online semantic point fusion module extends a highly efficient online point construction algorithm [53] to enable online semantic fusion and spatial semantic consistency computation from captured RGBD sequences. This 3D scene construction empowers a comprehensive 3D scene understanding for ObjectNav. Moreover, compared to dense voxel-based methods [7,55], our point-based fusion algorithm are more memory-efficient [40,46] which makes it practically usable for floor-level navigation task. (See Figure 1)

Moreover, to ease the learning of navigation policy, we further propose to factorize the navigation policy into two sub-policies, namely exploration and identification. The two policies simultaneously perform to roll out an exploration goal and an identified object goal (if exist), respectively. Then the input for the local path planning module will switch between these two goals, depending on whether there exists an identified target object. More specifically, we propose a corner-guided exploration policy which learns to predict a long-term discrete goal at one of the four corners of the bounding box of the scene. These corner goals efficiently drive the agent to perceive the surroundings and explore regions where the target object is possibly settled. And for identification, a category-aware identification policy is proposed to dynamically learn a discrete confidence threshold to identify the semantic predictions for each category. Both of these policies are trained by RL in low-dimensional discrete action space. Through experiments, the simultaneous two-policy mechanism and discrete action space design dramatically reduce the difficulty in learning for 3D-aware ObjectNav and achieve better performance than existing modular-based navigation strategies [26,35].

Through extensive evaluation on the public benchmarks, we demonstrate that our method performs online 3D-aware ObjectNav at 15 FPS while achieving the state-of-the-art performance on navigation efficiency. Moreover, our method outperforms all other modular-based methods in both efficiency and success rate with up to 30x times less computational cost.

Our main contributions include:

- We present the first 3D-aware framework for ObjectNav task.
- We build an online point-based construction and fusion algorithm for efficient and comprehensive understanding of floor-level 3D scene representation.
- We propose a simultaneous two-policy mechanism which mitigates the problem of low sample efficiency in 3D-aware ObjectNav policy learning.

2. Related Work

**GoalNav with Visual Sequences.** There are constantly emerging researches on object goal navigation. One line of recent works directly leverages RGBD sequences, called end-to-end RL methods [47], which tends to implicitly encode the environment and predict low-level actions. These works benefit from visual representation [29, 50], auxiliary task [51], and data augmentation [27], demonstrating strong results on object goal navigation benchmarks [1,49]. However, aiming to learn all skills through one policy from scratch, e.g., avoiding collisions, exploration, and stopping, it’s well known that end-to-end RL methods suffer from low sampling efficiency for training and limited generalizability when transferred to the real world [3,35]. Instead, our work uses explicit map to represent the environment, which ensures our sample efficiency and also obtain more generalizability through a modular-based paradigm [1,35].

**GoalNav with Explicit Scene Representations.** To ease the burden of learning directly from visual sequences, another category of methods, called modular-based methods [8,9,15,17,32], use explicit representations as a proxy for robot observations. By leveraging explicit scene representations like scene graph [34, 56] or 2D top-down map [18,35], modular-based methods benefit from the modularity and shorter time horizons. They are considered to be more sample efficient and generalizable [14,35]. Recent progress in modular-based methods has proposed a frontier-based exploration strategy [35], a hallucinate-driven semantic mapping method [14], and novel verification stage [26]. In contrast with prior map-based works, our method utilizes 3D spatial knowledge, including 3D point semantic prediction and consistency, enabling a more comprehensive understanding of the environment.

**Embodied AI tasks with 3D Scene Representation.** There are considerable research leveraging 3D scene repre-
3. Method

3.1. Task Definition and Method Overview

Object Goal Navigation Task. In an unknown environment, the Object Goal Navigation task requires the agent to navigate to an instance of the specified target category. For fair comparison, we follow the previous problem setting [38, 49]. As initialization, the agent is located randomly without access to a pre-built environment map, and provided with a target category ID. At each time step $t$, the agent receives noiseless onboard sensor readings, including an egocentric RGB-D image and a 3-DoF pose (2D position and 1D orientation) relative to the starting of the episode. Then the agent estimates its action $a_t \in \mathcal{A}$ for movement in a discrete action space, consisting of move_forward, turn_left, turn_right and stop. Given a limited time budget of 500 steps, the agent terminates the movement until it is within 1 meter of an object of the specified category.

Method Overview. Figure 2 provides an overview of the proposed 3D-aware ObjectNav method. Our method takes RGBD frames along with pose sensor readings as input, to online construct a point-based scene representation $\mathcal{M}_{3D}$ (Sec. 3.2), which is further projected to construct a 2D semantic map $\mathcal{M}_{2D}$. Given the structured 3D points $\mathcal{M}_{3D}$ and 2D map $\mathcal{M}_{2D}$, our framework simultaneously performs two complementary policies (Sec. 3.3), the exploration policy and identification policy at a fixed time cycle of 25 steps. The exploration policy predicts a long-term discrete corner goal $g_c^{t}$, to drive the agent to explore the surrounding environment. Meanwhile, the identification policy evaluates the 3D points $\mathcal{M}_{3D}$ at each step and outputs a target object goal $g_f^{t}$ if its semantic prediction is confident and consistent. The $g_f^{t}$ will be set as the approaching target for the agent once it exists, otherwise the agent will navigate to the long-term corner goal $g_c^{t}$. An underlying local planning module will navigate the agent towards the goal using analytical path planning.

3.2. Navigation-Driven 3D Scene Construction

During navigation, the 3D-aware agent will constantly obtain new observations and incrementally build a fine-grained 3D scene representation, integrating spatial and semantic information to drive the agent. However, given that our agent is deployed for a floor-level GoalNav task, it is fairly challenging to construct and leverage 3D representation across the entire scene while keeping an acceptable computational cost. Accordingly in this section, we extend an online point-based construction algorithm [53] to online organize the 3D points and further empower semantic fusion and consistency estimation. This design is tailored for a comprehensive scene understanding of the ObjectNav agent, requiring little computational resources.
**3D Scene Representation.** At time step \( t \), we represent the 3D scene \( \mathcal{M}_3^{\text{3D}} \) as the point clouds, denoted as \( P^{(t)}(t_i) = \{(P_{i,s}^{(t)}, P_{i,c}^{(t)})\} \in \mathbb{R}^{N(t) \times (M+4)} \), where \( N^{(t)} \) is the point number. For each point \( i \), the \( M+4 \) channels include the point position \( P_{i,i}^{(t)} \in \mathbb{R}^3 \), point semantics \( P_{i,s}^{(t)} \in \mathbb{R}^M \) and the point-wise spatial semantic consistency information \( P_{i,c}^{(t)} \in \mathbb{R}^1 \).

**Online 3D Point Fusion.** Given a new captured posed RGB image \( I_i^{(t)} \) and depth image \( I_d^{(t)} \) at time step \( t \), the agent can obtain the point position \( P_{i}^{(t)} \) by back-projecting all the depth images into the 3D world space via their corresponding poses. These points will be organized by a point-based construction algorithm [53]. Here, we briefly revisit this strategy.

The construction algorithm dynamically allocates occupied 3D blocks \( \{B_k\} \) along with their index \( k \) maintained by a tree-based method [20]. Each block \( B_k \) is defined by the boundary of constant length (10cm) along the X, Y and Z axes, e.g., \([X_{\text{min}}(B_k), X_{\text{max}}(B_k)]\). And the points \( p_{i,x} \in [X_{\text{min}}(B_k), X_{\text{max}}(B_k)] \) (the same requirement holds for Y and Z axes) be recorded by the block \( B_k \). Given any 3D point \( p_i \), the algorithm can achieve efficient neighborhood retrieval with the corresponding block index \( k \). Furthermore, a one-level octree \( \mathcal{O}_t \) for each point \( p_i \) is constructed to obtain the fine-grained spatial information among points. Specifically, we connect each point with its nearest points in the eight quadrants of the Cartesian coordinate system (see Figure 3). Powered by this point-based construction strategy, given any point, we can efficiently querying this point with its neighbor points by blocks retrieval and octree. This algorithm for organizing 3D points can run at 15 FPS while requiring reasonable memory resources (about ~ 500 MB for one entire scene). We provide more detailed description in the supplement material.

**Online Semantic Fusion.** With an efficient reconstruction algorithm in hand, we can directly fuse temporal information, e.g., multi-view semantic predictions, to achieve more accurate and consistent scene understanding. Specifically, any point \( p_i \) which has been captured by a sequence of RGBD frames \( \{I_i^{(t)}, I_d^{(t)}\} \) could have multiple semantic predictions \( \{P_{i,s}^{(t)}(I_i^{(t)})\} \). We thus propose to online aggregate the multi-view 2D semantic predictions using a max-fusion mechanism to obtain the final 3D semantic prediction:

\[
P_{i,s}^{(t)} = \mathcal{N}(\max(\{P_{i,s}^{(t)}(I_i^{(t)}))\)),
\]

where the \( \max \) is performed on each semantic category, followed by a normalization \( \mathcal{N} \) to linearly scale the probability distribution. Note that, the alternatives to fuse semantic predictions do exist, e.g., 3D convolution [19, 24], Bayesian updating [28]. However, directly conducting 3D convolution into such a floor-level 3D representation would inevitably lead to a huge rise of computational cost, especially in the context of learning-based policy. We find that maximizing the 2D semantic prediction can already achieve impressive improvement on semantic accuracy (see Figure 8), with higher memory efficiency and time efficiency. Similar findings have also been reported and exploited in relevant works [7, 16].

**Spatial Semantic Consistency.** Based on the fact that semantic label should remain consistent for all the points in a single object, we propose to calculate the spatial semantic consistency information \( P^{(t)}_{s,i} \) as part of the navigation-driven 3D scene representation. To be specific, \( P^{(t)}_{s,i} \) is computed as the maximum semantic KL-divergence between point \( P^{(t)}_i \) and its octree \( \mathcal{O}(P^{(t)}_i) \):

\[
P^{(t)}_{s,i} = \max(\{K L(P^{(t)}_{i,s}, P^{(t)}_{j,s}) | \forall P^{(t)}_j \in \mathcal{O}(P^{(t)}_i)\}),
\]

where \( K L \) denotes the KL-divergence computation, which is a statistical distance that measures the semantic probability distribution between \( P^{(t)}_{i,s} \) and \( P^{(t)}_{j,s} \). Note for point \( P^{(t)}_i \), if we count all its spatially close points as the neighbourhood \( N(P^{(t)}_i) \), it could be time consuming to calculate Equation 2, and the spatially close points do not help relieve the issue of outlier points as mentioned above. Therefore, we use the pre-built octree \( \mathcal{O}_i \) to retrieval nearest point in the quadrants of the Cartesian coordinate system.

**3.3. Simultaneous Exploration and Identification**

With the aggregated 3D information, we expect to empower a 3D-aware agent for the ObjectNav task. However, despite the efficient 3D scene representation, the agent still suffers from the complex and high-dimensional observations, leading to a lower sample efficiency in RL and hampering the navigation policy learning. Therefore, we leverage two complementary sub-policies: corner-guided exploration policy and category-aware identification policy. Each
policy learns to predict low-dimensional discrete actions and outputs a goal location to navigate the agent, resulting in a strong performance while requiring less training time. We will detail the two policies below.

**Observation Space.** At each time step $t$, both policies take fine-grained 3D observation $x_{3D}^{(t)} = \{P^{(t)} \in (4+m \times N)\}$ based on 3D scene representation $\mathcal{M}_{3D}$. Here, the $N$ indicates the point number (we sample 4096 points) and the $m + 4$ channels are comprised of point position $p^{(t)} \in \mathbb{R}^3$, fused semantic predictions $p_s^{(t)} \in \mathbb{R}^m$ and spatial semantic consistency $p_c^{(t)} \in \mathbb{R}^3$. Following existing works [8, 9], we use an additional egocentric 2D map $\mathcal{M}_{2D}$ for exploration policy and the local path planning module, which is directly obtained by a project-to-ground operation. More detailedy, for 2D observation $x_{2D}^{(t)} \in ((2+m) \times M \times M)$ from 2D map $\mathcal{M}_{2D}$, the first two channels represent obstacles and explored area, and the rest of the channels each corresponds to an object category. Here, $\mathcal{M}_{2D}$ (in a resolution of $M = 240$ with $20\times20$ grids) is constructed to give a large perception view of the scene, while 3D points perform as a fine-grained observation of objects. In addition to the scene representations, we also pass the goal object category index $o_{1D}$ as the side input to both policies.

**Corner-Guided Exploration Policy.** The exploration policy attempts to guide the agent to explore and perceive the surrounding environment where it could access any instance of the target object category. We observe that existing learning-based exploration policies predict goal locations over the 2D map in continuous or large-dimensional discrete action space (Figure 4 Left), suffering from low sample efficiency. Therefore, we define a corner-guided exploration policy $g_e = \pi_e(x_{3D}, x_{2D}, o_{1D}; \theta_e)$ that predicts a corner goal $g_e$ to drive the agent(Figure 4 Right). Here, the $\theta_e$ indicates the parameters of the policy, and $g_e$ is one of the four pre-defined corner goals [Top Left, Top Right, Bottom Left, Bottom Right] of the 2D map.

Compared to predicting goals in a continuous or high-dimensional action space, learning to predict the four corner goals significantly reduces the learning difficulty. More-over, as noted by previous studies [4, 26], the corner-goal-based exploration strategy exhibits the capacity to achieve efficient exploration through avoiding back-and-forth pacing. Superior to using other heuristic corner goal exploration strategies (Figure 4 Middle), our agent can learn from the 3D scene priors to behave more intelligently. Demonstrations of our corner-guided exploration can be found in the attached video.

**Category-Aware Identification Policy.** During navigation, the agent consistently makes semantic predictions to identify an instance of target object category. Most works [9, 14] simply use a preset hard confidence threshold for target identification. However, this strategy is inherently sub-optimal due to the considerable variability in semantic prediction results across different categories and observation angles. As a result, a preset threshold would be unable to adequately adapt to the ever-changing nature of these scenarios. Also, it ignores to consider the consistency of the semantic prediction in 3D space.

To tackle this issues, we propose to leverage both dynamic confidence threshold and spatial semantic label consistency for target identification. We define a policy $s = \pi_f(x_{3D}, o_{1D}; \theta_f)$ which takes the 3D observation $x_{3D}$ and target category index $o_{1D}$ and outputs a threshold-indicating action $s \in \{0, 1, ..., 9\}$. And the dynamic threshold $\tau$ can be obtained by:

$$\tau = \tau_{low} + \frac{1 - \tau_{low}}{10},$$

where the $\tau_{low}$ is set to 0.5 in our implementation for a threshold range $\tau \in [0.5, 0.95]$. The $\tau$ will be used to dynamically identify the points belonging to the target object (Figure 5 Middle). It is worth mentioning that that this policy also utilizes a low-dimensional discrete action space, which is fairly easy for the agent to learn.

To obtain the final target goal $g_f$, our method further checks the spatial semantic label consistency. Specifically, we use the points $\{p_l, p_r \in \mathcal{O}_p\}$ connected by the per-point octree $\mathcal{O}_p$ to approximately represent the 3D surface of the target object. Our insight is that the points along the target’s surface should have consistent semantic.
tags. Therefore, we only identify those points which have at least 2-ring neighbors across the octree \((p_i|p_j,p_j)\in O_p, |p_j,p|\in O_p\) as the target object goal \(g_f\) (Figure 5 Right). See Figure 5 for visualized illustration and more details can be found in supplemental material.

**Local Planning Module.** The goals \(g_e\) and \(g_f\) from two polices will be consistently updated during navigation. Our method will preferentially utilize the target goal \(g_f\) if it exists, otherwise take the long-term corner goal \(g_e\) to explore. To navigate to the given location, we use the Fast Marching Method [42] to analytically plan the shortest path from the agent location. The agent then takes deterministic actions to follow this path.

**Rewards.** For the exploration policy, we share a similar reward design as [1, 51]. The agent receives a sparse success reward \(r_{success} = 2.5\), a slack reward \(r_{slack} = 10^{-2}\) and an exploration reward \(r_{explore}\). The exploration reward is a dense reward, defined by the number of new inserted points \(n_p^{new}\) as \(r_{explore} = n_p^{new} \times 10^{-3}\). The slack reward and exploration reward encourage the agent to take the most effective direction to the unobserved area. And for the identification policy, we combine the same success reward and slack reward borrowed from the exploration policy.

### 4. Experiments

#### 4.1. Experiment Setup.

We perform experiments on the Matterport3D (MP3D) [6] and Gibson [48] datasets with the Habitat simulator [39]. Both Gibson and MP3D contain photorealistic 3D reconstructions of real-world environments. For Gibson, we use 25 train / 5 val scenes from the Gibson tiny split. And we follow the same setting as in [9, 35] where we consider 6 goal categories, including chair, couch, potted plant, bed, toilet and TV. For MP3D, we use the standard split of 61 train / 11 val scenes with Habitat ObjectNav dataset [38], which consists of 21 goal categories (the full list can be found in the supplemental material). Note that, the RGB-D and pose readings are noise-free from simulation (follow the definition of [1]). Estimation of the pose from noisy sensor readings is out of the scope of this work and can be addressed if necessary, by incorporating off-the-shelf robust odometry [52,54].

**Implementation Details.** On MP3D, we use the same pre-trained 2D semantic model RedNet [21] as [35,51]. On Gibson, we leverage a Mask R-CNN [18], which is trained with COCO dataset [23]. For each frame, we randomly sample 512 points for point-based construction. Moreover, we use PointNet [33] and fully convolutional networks [25] to obtain the feature of 3D points and the 2D map, respectively. During training, we sample actions every 25 steps and use the Proximal Policy Optimization (PPO) [41] for both execution and identification policies. More implementation details can be found in the supplemental material.

**Evaluation Metrics.** Following existing works [2,14,35], we adopt the following evaluation metrics: 1) SPL: success weighted by path length. It measures the efficiency of the agent over oracle path length, which serves as the primary evaluation metric for Habitat Challenge [49]. 2) Success rate: the percentage of successful episodes 3) Soft SPL: a softer version of SPL measure the progress towards the goal (even with 0 success). 4) DTS: geodesic distance (in m) to the success at the end of the episode.

**Baselines.** We consider mainstream baselines in the ObjectNav task. For end-to-end RL methods, we cover DD-PPO [47], Red-Rabbi [51], THDA [27], and Habitat-Web [36]. For modular based methods, we cover FBE [37], ANS [8], L2M [14], SemExp [9], Stubborn [26] and PONI [35]. Note that, some works use additional data to improve the performance, e.g. Habitat-web leverages human demonstration trajectories, and THDA utilizes data augmentation. It is challenging to compare all the methods fairly. Therefore, we are particularly interested in the three most relevant baselines: SemExp, Stubborn, and PONI. These three methods share the same 2D semantic predictors [18,21] as our method.

#### 4.2. Results

**Comparison on MP3D and Gibson.** We evaluate our approach on MP3D (val) and Gibson (val) with other baselines, including end-to-end RL(rows 1 - 4) and modular-
Figure 6. An qualitative visualization of the trajectory of the proposed method. We visualize an episode from MP3D where an agent is expected to find a bed. The semantic prediction $p_s$ and spatial semantic consistency $p_c$ of points are visualized on the left. During navigation, the agent can successfully dismiss the wrong prediction and approach and finally call stop around the target object.

Table 3. Comparison of different exploration policies. Here, all methods share the same identification strategy from [9] for fair comparison.

<table>
<thead>
<tr>
<th>Method</th>
<th>SPL(%)</th>
<th>Succ.(%)</th>
<th>DTS(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn Continuous Goal.</td>
<td>11.1</td>
<td>28.6</td>
<td>6.354</td>
</tr>
<tr>
<td>Learn dense Grid Goal.</td>
<td>12.7</td>
<td>29.5</td>
<td>5.635</td>
</tr>
<tr>
<td>Learn 8 corner goal.</td>
<td>12.9</td>
<td>30.7</td>
<td>5.112</td>
</tr>
<tr>
<td>Heuristic. 4 corner goal.</td>
<td>13.5</td>
<td>33.0</td>
<td>4.995</td>
</tr>
<tr>
<td>Learn 4 corner goal. (Ours)</td>
<td><strong>13.9</strong></td>
<td><strong>33.5</strong></td>
<td><strong>4.931</strong></td>
</tr>
</tbody>
</table>

Table 4. Comparison on different identification policies.

<table>
<thead>
<tr>
<th>Method</th>
<th>Type</th>
<th>SPL(%)</th>
<th>Succ.(%)</th>
<th>DTS(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deterministic</td>
<td>2D</td>
<td>0.85</td>
<td>12.8</td>
<td>30.1</td>
</tr>
<tr>
<td></td>
<td>3D</td>
<td>0.85</td>
<td>13.8</td>
<td>32.5</td>
</tr>
<tr>
<td>Learning (Ours)</td>
<td>3D</td>
<td>-</td>
<td><strong>14.6</strong></td>
<td><strong>34.0</strong></td>
</tr>
</tbody>
</table>

We also provide a qualitative visualization of MP3D episodes in Figure 6. Here, our method online updates the semantic prediction and successfully dismisses the wrong target goal. For more qualitative results, please refer to the supplemental material.

Comparison on Exploration Policy. We conduct an experiment to verify the efficiency of our corner-guided exploration policy on MP3D. To remove the effect of the 2D semantic predictor and identification policy, all competitors share the same semantic predictor and a heuristic identification policy proposed in SemExp [9]. The results are reported in Table 3. Our corner-guided exploration policy outperforms the mainstream existing methods, including learning-based ones [8, 14] and heuristic ones [26]. Our findings indicate that the best performance is achieved through learning to predict discrete corner goals from the four corners of the scene. This suggests that the four-corner design, which benefits from a small, discrete action space, is already capable of efficiently guiding the agent in exploring the environment.

Comparison on Identification Policy. Another critical challenge in ObjectNav is how to properly identify an
Figure 7. An comparison of predicted threshold distribution between different categories by our category-aware policy. We report the ratio of the each predicted threshold.

stance of target object category. Therefore, We evaluate our identification policy on MP3D along with other identifying strategies, including a 2D frame-based policy adopted in [9] and 3D point-based methods proposed by our approach. The results are shown in Table 4. We observe a performance improvement (rows 1 - 2) by simply leveraging 3D point-based construction and fusion algorithm. It can demonstrate that the multi-view observations provide more accurate semantic prediction, which effectively reduces false positive prediction (see examples in Figure 8). Moreover, our category-aware identification policy, through predicting dynamic threshold, demonstrates an even better performance.

To further investigate the effect of our identification policy, We conduct a break down study in Figure 7 by plotting the distribution of predicted semantic confidence thresholds. Specifically, we plot the distribution of three different categories (table, cushion, plant). For a relatively easy-to-recognize category, such as table with 52.6% success rate (SR), our policy predict a broad threshold distribution. However, for more challenging categories, such as cushion (36.9% SR) and plant (16.1% SR), the policy tends to be more conservative through setting a higher threshold. The results demonstrate the category-aware characteristic of our identification policy which adapts well to different difficulty levels across categories.

Ablation Study. We also perform an ablation study to verify the effectiveness of different components of our method. The results are demonstrated in Table 5. The cooperation of the 2D top-down map and 3D points (row 4) shows significant improvement by incorporating extensive scene perception (in 2D) and fine-grained object perception (in 3D). Moreover, rows (3-4) and (4-5) proved the effectiveness of leveraging consistency information and the identification policy, respectively.

Analysis of Computational Cost. Our framework is extremely memory efficient, which requires about 0.5GB for one scene, and can perform online construction and semantic fusion at a frame rate of 15 FPS. Moreover, our method requires only 48 GPU hours to train a 3D-aware agent on MP3D dataset to achieve the SOTA performance among all modular-based methods. This is significantly faster (30x) than other existing reinforcement learning based methods [9, 51], and is comparable to supervised learning modular-based methods [35].

5. Conclusion

In this work, we present a 3D-aware framework for object goal navigation. Our method is based on a 3D point-based construction algorithm to observe the 3D scenes and simultaneously perform exploration and identification policies to navigate the agent. Our method achieve SOTA performance among all modular-based methods, while requiring less training time. In the future, we would like to exploit this 3D-aware framework in other embodied AI tasks, e.g. mobile manipulation, robotic nurses.

Acknowledgements. We thank anonymous reviewers for their valuable suggestions. This work was supported by National Key Research and Development Program of China (2018AAA0102200), NSFC (62132021), and Beijing Academy of Artificial Intelligence (BAAI).
References


Thang Vu, Kookhoo Kim, Tung Minh Luu, Xuan Thanh Nguyen, and Chang-Dong Yoo. Softgroup for 3d instance segmentation on point clouds. ArXiv, abs/2203.01509, 2022.


