Robotics: Science and Systems 2023
Daegu, Republic of Korea, July 10-July 14, 2023

How To Not Train Your Dragon: Training-free Embodied Object Goal Navigation with Semantic Frontiers

Junting Chen[1*], Guohao Li[2*], Suryansh Kumar[1†], Bernard Ghanem[2], Fisher Yu[1] ,

Abstract—**Object goal navigation is an important problem in Embodied AI that involves guiding the agent to navigate to an instance of the object category in an unknown environment—typically an indoor scene. Unfortunately, current state-of-the-art methods for this problem rely heavily on data-driven approaches, *e.g.*, end-to-end reinforcement learning, imitation learning, and others. Moreover, such methods are typically costly to train and difficult to debug, leading to a lack of transferability and explainability. Inspired by recent successes in combining classical and learning methods, we present a modular and training-free solution, which embraces more classic approaches, to tackle the object goal navigation problem. Our method builds a structured scene representation based on the classic visual simultaneous localization and mapping (V-SLAM) framework. We then inject semantics into geometric-based frontier exploration to reason about promising areas to search for a goal object. Our structured scene representation comprises a 2D occupancy map, semantic point cloud, and spatial scene graph. Our method propagates semantics on the scene graphs based on language priors and scene statistics to introduce semantic knowledge to the geometric frontiers. With injected semantic priors, the agent can reason about the most promising frontier to explore. The proposed pipeline shows strong experimental performance for object goal navigation on the Gibson benchmark dataset, outperforming the previous state-of-the-art. We also perform comprehensive ablation studies to identify the current bottleneck in the object navigation task.**

I. Introduction

In recent years, the focus on computer vision has moved from using "internet data" such as images, videos, texts, etc., towards developing an active vision system involving a robot or an agent that can perceive the 3D scene and act intelligently. Accordingly, current research trends in this direction have begun to advocate for building Artificial Intelligence (AI) that involves embodiment [16, 31, 11, 27]. Among several popular tasks in Embodied-AI, object goal navigation (ObjectNav) is one of the most important and sought-after tasks to solve [3]. ObjectNav requires an agent to search for any specific object category instance in an unknown environment. Unlike classical visual navigation [4], this task setup does not provide the target's position. As a result, it requires the agent to understand the scene's geometry and other higher-level semantics [7].

To perform ObjectNav, one possible sequence of steps similar to human solutions is: First, an agent must understand the target object such as its appearance, shape, etc. Next, the agent should explore the unseen environment while at the same time determining whether the target object is observed. If so, the agent takes the shortest path possible to reach the object while avoiding collisions. If not, it must resort to the favorable unexplored areas to unveil based on current information, *i.e.*, knowing which parts of the scene are explored and reason about the likelihood of the object based on observations. Overall, an agent needs to have capabilities such as episodic memory to remember the unexplored parts of the scene, higher-level semantic priors to reason the next exploration location, path planning to go to the object, and collision avoidance.

In the same vein, previous works attempt to solve this problem in two different ways. The *first line of work* formulates it as an end-to-end learning problem. They try to learn both robot perception and control using reinforcement or imitation learning approaches. Although end-to-end learning methods have shown some promising results on a few datasets, they typically have low sample efficiency, questionable generalization, and lack explainability; hence, it is hard to reason about its failure or success case and deploy them on real robots for practical use applications. The *second line of work* combines classic navigation approaches with learning-based methods [6, 7]. For instance, SemExp [7] uses a learning-based semantic mapping module based on Active Neural SLAM [6] to build a semantic grid map. Furthermore, a goal-oriented semantic policy is trained to predict long-term goals based on the semantic map using reinforcement learning. Fast Marching Method [14], an analytical path planner, is used to plan a path for the long-term goal. Finally, discrete actions are generated by a deterministic controller along the path. Compared to end-to-end methods, SemExp has better sample efficiency, generalization, and transferability to real-world scenarios. Yet, SemExp requires 10 million frames to train their network.

To mitigate the limitations of the current approaches, this work proposes a modular and training-free pipeline **StructNav**, which navigates an agent with a structured representation of the environment for the target object. Our method improves and enhances the current state of employing classic and learning-based modular approaches to solve this problem. Compared to SemExp [7], StructNav introduces a classic semantic visual SLAM to obtain a semantic point cloud map instead of a learned 2D semantic mapping module. Moreover, StructNav leverages a structured

[*] First two authors share equal contributions.
[1] ETH Zürich
[2] King Abdullah University of Science and Technology (KAUST)
[†] Corresponding Author (k.sur46@gmail.com)

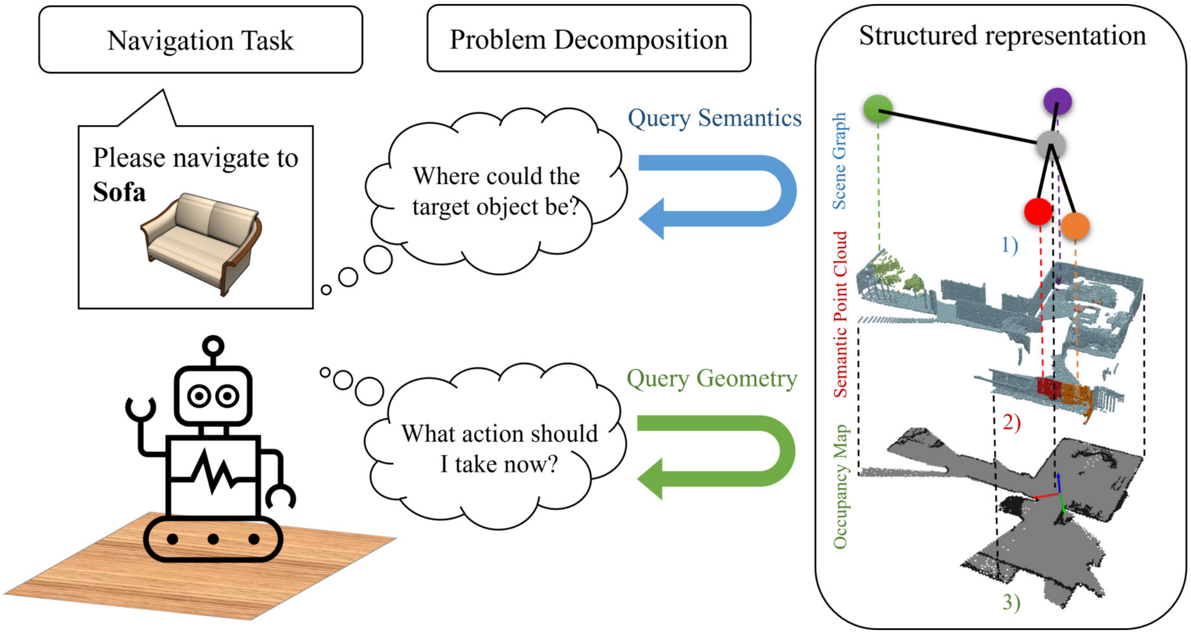


Figure 1: **Object Navigation with Structured Scene Representation.** ObjectNav can be decomposed into inferring the potential position of the target object in the scene and point-to-point planning. Provided a structured representation of the scene, which is composed of *(i)* a spatial scene graph, *(ii)* semantic point clouds, and *(iii)* a 2D occupancy map, an agent can handle the two sub-tasks by querying semantic and geometric information from the scene graph and occupancy map separately. For clarity, the scene graph and occupancy map are computed from the semantic points cloud, thus the semantic point cloud is also considered part of our structured scene representation.

scene representation in which a 2D occupancy map for planning and a spatial scene graph for reasoning is constructed from the point cloud map. To avoid training an exploration policy with reinforcement learning, we propose a semantic frontier (SemFrontier) module, which combines classic frontier-based exploration with semantic priors. Our structured scene representation allows easy generation of proposal frontiers and propagating semantics to the frontiers. Specifically, we introduce language and scene-based priors to reason the promising unexplored areas by scoring geometric-based frontiers with semantics via the spatial scene graph. The language priors are acquired from pre-trained large-scale language models that encode knowledge from large-scale natural language inference datasets. The scene priors are obtained from the training split of 25 Gibson scenes, following the Batra et al. [3] experiment setting.

From the perspective of real-world robotic systems setup, there is still an interesting sim-to-real gap for recent approaches in the object goal navigation domain, many of which adopt the ideal problem setting defined in [1, 3], where ground truth localization is provided. However, as a fundamental building block of robotic systems, localization significantly impacts downstream applications, including mapping and path planning. In this work, we also want to understand how this gap affects general visual navigation performance and how noise in other building blocks of the robotic pipeline affects performance. Thus, we propose a comprehensive benchmark for our modular method in which the ground truth output for each of our building blocks is turned on/off, including robot location, to assess how noise in the different building blocks of a robotic system affects the task of object goal navigation.

To summarize our key contributions are:

- We introduce a modularized pipeline that reaches state-of-the-art performance for Object Goal Navigation, with a simple learning-free policy module, easily deployable to ROS-based robots.
- Our work propose a more realistic and comprehensive benchmark for visual navigation and discuss the impact of common building blocks of a robotic system on the performance of object goal navigation.

II. Related Work

Our proposed method contributes to several modules in the overall pipeline for object goal navigation, including different perception and control aspects such as scene representation, navigation, *etc.*. Therefore, we discuss the works closely related to our proposed pipeline for brevity.

***(a)* Classical Vision-Based Navigation.** These methods rely on the explicit map of the scene and generally perform point-to-point navigation. The agent explores the scene using frontier-based exploration algorithm [32] and navigates along the computed optimal path to the goal using the well-crafted path planning algorithm [18, 28].

***(b)* Object Goal Navigation.** The end-to-end reinforcement learning (RL) based method has gained popularity for solving object goal navigation tasks in recent years [26, 3, 1, 29].

Current attempts have tried to improve the performance of such methods generally by using data-augmentation [20], and higher-level scene representation [8, 23, 9]. By higher-level scene representation, we mean object-level relation graph [33], spatial attention map [21], semantic segmentation map [22], and others [34]. Lately, a combination of RL-based learning with classical methods has emerged, showing excellent performance accuracy compared to end-to-end learning methods [25, 7, 19]. Nevertheless, as mentioned before, popular methods along this line of approaches rely on RL-based semantic exploration module, which requires extensive training data. On the contrary, [25] proposed a convolutional encoder-decoder module which is trained on 3D semantic segmentation dataset [31, 5] to overcome the computational overhead with [7, 19].

III. Methodology

Section III-A provides a formal definition of the ObjectNav problem. In Section III-B an overview of the pipeline is outlined, specifically focusing on how data are processed through the pipeline. Section III-C, contains details about how the structured representation is constructed. Lastly, Section III-D describes how navigation goals and actions are calculated based on our structured scene representation.

A. Problem Definition of Object Navigation

We formally define ObjectNav following [3]. An agent is randomly initialized on the floor in an unknown environment $\mathcal{E}$ with the initial pose $\boldsymbol{s}_0^w = (\boldsymbol{x}_0^w, \boldsymbol{r}_0^w)$, where $\boldsymbol{x}_0^w \in \mathbb{R}^3$ and $\boldsymbol{r}_0^w \in SO(3)$ represent the initial position and the initial rotation in the world frame w, respectively. The agent is required to navigate to an instance of the goal object category g specified by the category name (such as "chair"). We use the Habitat simulator [27] as our testbed, in which the action space $\mathcal{A}$ is discretized into four actions: `turn_left`, `turn_right`, `move_forward`, and `stop`. At time step t, the agent executes action a_t and receives visual observation $\boldsymbol{I}_t = (\boldsymbol{I}_t^{\text{rgb}}, \boldsymbol{I}_t^{\text{depth}})$ from a noiseless RGBD camera, where $\boldsymbol{I}_t^{\text{rgb}} \in \mathbb{R}^{H \times W \times 3}$ for the RGB image and $\boldsymbol{I}_t^{\text{depth}} \in \mathbb{R}^{H \times W}$ for the depth image.

In the experimental setting of SemExp [6], the agent is equipped with a noiseless GPS+Compass sensor that provides the true relative pose information $\boldsymbol{s}_t^{l*} = (\boldsymbol{x}_t^{l*}, \boldsymbol{r}_t^{l*})$ to the initial state in the local frame l at time t. In this paper, we also propose to evaluate ObjectNav in a more realistic setting, in which the localization from visual odometry on the fly replaces ground-truth localization. In this setting, the pose of an agent is annotated $\hat{\boldsymbol{s}}_t^l = (\hat{\boldsymbol{x}}_t^l, \hat{\boldsymbol{r}}_t^l)$.

B. Structured Navigation Pipeline

We propose a modular and *training-free* pipeline named *StructNav* to tackle the ObjectNav problem. StructNav uses a structured scene representation that consists of a semantic point cloud, a 2D occupancy map, and a spatial scene graph. With this structured representation, the semantic information for target position inference and the geometric information for planning on the 2D map can be decomposed and queried by different modules separately.

Fig. 2 depicts how our training-free pipeline interacts with the 3D physical environment via receiving observations from sensors and executing planned actions. At each time step t, data flow starts at observations from the agent sensor and ends at actions sent to the agent controller. The data flow consists of two stages: perceiving to generate a structured scene representation and exploiting the structured representation to perform navigation actions.

In the first stage, our pipeline updates the structured representation of the scene by the current observation: 1) Given the visual observation $\boldsymbol{I}_t$, the semantic segmentation image $\boldsymbol{I}_t^{\text{sem}}$ is predicted from the input RGB image $\boldsymbol{I}_t^{\text{rgb}}$ by a pre-trained Mask R-CNN [13]; 2) The visual SLAM module receives the latest visual observation $\boldsymbol{I}_t$, predicts the current agent pose $\hat{\boldsymbol{s}}_t^l$, and updates the dense RGB reconstruction of the scene $\boldsymbol{P}_t^{\text{rgb}}$; 3) The dense RGB reconstruction of the scene is projected onto $xy - plane$ to generate a 2D occupancy map $\boldsymbol{M}_t$; 4) Taken into the agent pose $\hat{\boldsymbol{s}}_t^l$, depth image $\boldsymbol{I}_t^{\text{depth}}$, and the semantic image $\boldsymbol{I}_t^{\text{sem}}$, the semantic point cloud $\boldsymbol{P}_t^{\text{sem}}$ is updated by the 3D fusion module via back projection; 5) Spatial scene graph $\mathcal{G}_t = (\boldsymbol{V}_t, \boldsymbol{E}_t)$ is extracted from the semantic point cloud $\boldsymbol{P}_t^{\text{sem}}$.

In the second stage, our pipeline computes an intermediate navigation goal in the map and generates an action to execute: 6) Frontiers $\boldsymbol{F}_t = \{\boldsymbol{f}_t^0, \cdots, \boldsymbol{f}_t^{N_t}\}, \boldsymbol{f}_t \in \mathbb{R}^2$ are cluster centers of the boundary pixels between the explored area and the unexplored area on the 2D occupancy map, as Yamauchi *et al.* [32] do; 7) Utility module combines the geometric information from the frontiers $\boldsymbol{F}_t$ and semantic information from the spatial scene graph $\mathcal{G}_t$ to select the most promising frontier as the navigation point goal $\boldsymbol{x}_t^{\text{goal}} \in \mathbb{R}^2$. 8) The global planner estimate the path from the current agent pose $\hat{\boldsymbol{s}}_t^l$ to the point goal $\boldsymbol{x}_t^{\text{goal}}$ and the local planner selects the correct action a_t to follow the path at time t.

For clarity, we provide our algorithmic approach in Pseudo Code 1, demonstrating how data is processed in the pipeline.

C. Structured Scene Representation

As discussed in section III-B, our proposed structured scene representation has the following components: 1) 2D occupancy map $\boldsymbol{M}_t \in \mathbb{R}^{h_t \times w_t}$, where h_t and w_t are the height and width of the occupancy map, which is a slack bound of the explored area, automatically expanded by the SLAM module; 2) semantic point cloud $\boldsymbol{P}_t^{\text{sem}} \in \mathbb{R}^{k_t \times 4}$, where k_t is the number of points in the point cloud at time t. Each point has four channels. The first three channels are point coordinates, and the last channel is the semantic label; 3) spatial scene graph $\mathcal{G}_t = (\boldsymbol{V}_t, \boldsymbol{E}_t)$. $\boldsymbol{V}_t = \{\boldsymbol{v}_t^i\}$ are object nodes in the graph, and each node $\boldsymbol{v}_t^i \in \mathbb{R}^4$ has the same 4 channels as the semantic point cloud $\boldsymbol{P}_t^{\text{sem}}$, representing the center of the object and its label. Edges

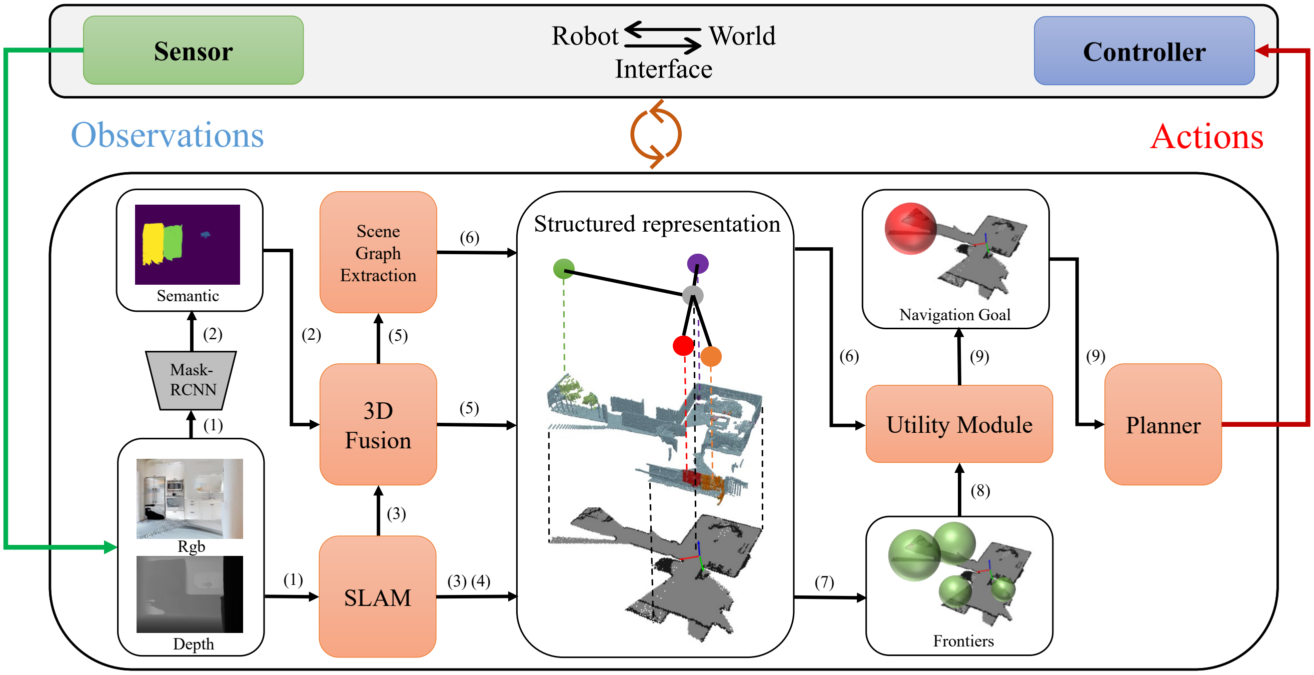


Figure 2: **Overview of the StructNav pipeline.** Our pipeline runs in the loop of receiving observations and generating actions to navigate an agent to the goal object in an unknown scene. Colored boxes shows functional modules and arrows represent data flows: (1) RGBD observation $\boldsymbol{I}_t = (\boldsymbol{I}_t^{\text{rgb}}, \boldsymbol{I}_t^{\text{depth}})$ (2) Semantic Image $\boldsymbol{I}_t^{sem}$ (3) Estimated pose $\hat{\boldsymbol{s}}_t^l$ (4) RGB point cloud $\boldsymbol{P}_t^{\text{RGB}}$ (5) Semantic point cloud $\boldsymbol{P}_t^{\text{sem}}$ (6) Spatial scene graph $\mathcal{G}_t$ (7) 2D occupancy map $\boldsymbol{M}_t$ (8) Frontiers $\boldsymbol{F}_t$ (9) Intermediate navigation goal $\boldsymbol{x}_t^{\text{goal}}$.

$\boldsymbol{E}_t = (\boldsymbol{e}_t^{ij})$ are translations between object nodes, $\boldsymbol{e}_t^{ij} \in \mathbb{R}^3$ represents the translation from $\boldsymbol{v}_t^i$ to $\boldsymbol{v}_t^j$. To construct this structured representation, we build our pipeline on top of a popular visual SLAM system RTAB-Map [17], which takes per-frame RGBD observations to predict the agent pose and construct the semantic point cloud on the fly. We use a purely geometry-based algorithm DBSCAN [10] to cluster the semantic point cloud, which clusters points based on the point density in the neighborhood. It is worth mentioning that we take the semantic label weighted by a large factor as the fourth dimension in DBSCAN to avoid points of different categories being classified into one cluster. The semantic segmentation model could produce false predictions and inter-frame inconsistency in 2d semantic images, leading to faulty object nodes in 3D. To reduce the false predictions, we compute the bounding boxes of all clusters and then use 3D non-maximum suppression (3DNMS) to filter out small clusters which share a high IOU rate with other large ones.

D. Exploration with Semantic Frontiers

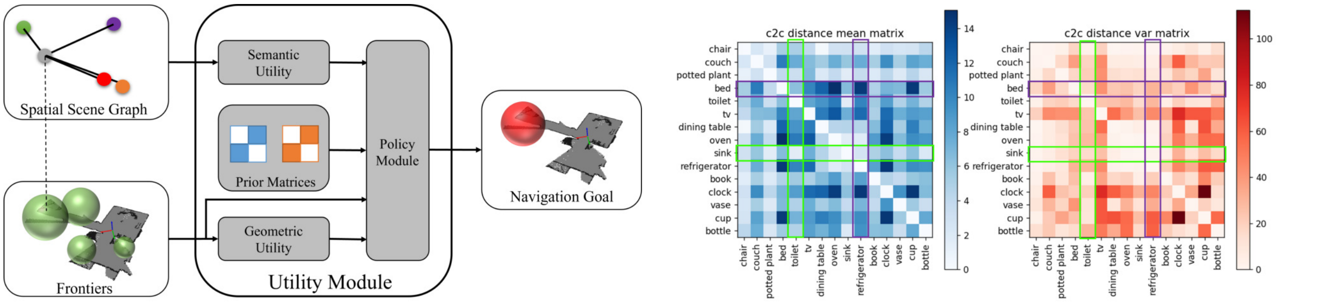
With the structured scene representation, an agent explores the unknown environment with semantic reasoning on the spatial scene graph for target inference and path planning on the 2D grid map until the target is detected in the current observations. Then it simply navigates to the target object with the planner. In the rest of this section, we will focus on frontier-based exploration with semantic utilities.

Fig. 3(a) demonstrates how our utility module generates an intermediate navigation goal $\boldsymbol{x}_t^{\text{goal}}$ in the exploration stage. Inspired by the idea of frontier-based exploration [32], our pipeline generates the frontiers $\boldsymbol{F}_t = \{\boldsymbol{f}_t^0, \cdots, \boldsymbol{f}_t^{N_t}\}$ from the 2D occupancy map $\boldsymbol{M}_t$ as intermediate navigation goal candidates. Each frontier $\boldsymbol{f}_t^i = [x_t^i, y_t^i, l_t^i] \in \mathbb{R}^3$ is composed of its position $[x_t^i, y_t^i]$ and frontier length l_t^i. At the beginning of exploration, the limited observations could provide very little semantic information about the scene so that the agent will explore the environment with the geometric utility, that is, to take the frontier with the maximum utility as the navigation goal. The geometric utility of frontier $\boldsymbol{f}_t^i$ is defined as

$$u_{t,geo}^i = \frac{l_t^i}{dist(\hat{\boldsymbol{s}}_t^l, \boldsymbol{f}_t^i, \boldsymbol{M}_t)} \quad (1)$$

where $dist(\hat{\boldsymbol{s}}_t^l, \boldsymbol{f}_t^i, \boldsymbol{M}_t)$ is the geodesic distance from the current agent state to frontier $\boldsymbol{f}_t^i$ on the 2D occupancy map $\boldsymbol{M}_t$. This heuristic function, which derives a larger value with a larger frontier size and a shorter distance, describes the score of a frontier in greedy exploration policy.

However, since our task is to navigate to the target object with the shortest path, instead of exploring as much area of the scene as possible, we propose to use novel utility functions to exploit the semantic information in the explored partial scene, to avoid unnecessary exploration. For this purpose, We calculate the semantic utility for each frontier by propagating



(a) Utility Module

(b) Prior Matrices

Figure 3: **Utility Module and Prior Matrices.** a) Our utility module calculates the semantic utility from the spatial scene graph and the geometric utility from the frontiers, respectively. A policy module calculates the most promising frontier as the temporary navigation goal, based on the utilities and prior matrices. b) Prior matrices comprise a category-to-category prior distance matrix $\boldsymbol{D}_{prior}$ and a category-to-category prior distance variance matrix $\boldsymbol{V}_{prior}$. The green cross highlights the relationship between *sink* and *toilet*, that they have close proximity with high confidence. The purple cross highlights the relationship between *bed* and *refridgerator*, that they are always far from each other.

semantic information of object nodes near the frontier to the frontier as its semantic utility. Specifically, we try to leverage the spatial relations between the observed objects in the scene with the target object category. For example, a frontier close to a basin and a toilet should be promising for finding a bathtub. The semantic utility of frontier $\boldsymbol{f}^i_t$ is defined as

$$u^i_{t,sem} = \frac{\boldsymbol{r}^i_t \cdot \boldsymbol{w}^i_t}{k \cdot dist(\hat{\boldsymbol{s}}^L_t, \boldsymbol{f}^i_t, \boldsymbol{M}_t)} \qquad (2)$$

where $\boldsymbol{r}^i_t \in \mathbb{R}^k$ is the relation scores vector of the k objects around frontier $\boldsymbol{f}^i_t$. The relation score for object category j to the target category c is $r_{jc} = 1/d_{jc}$, that is, the inverse of the prior distance between category j and c. Similarly, $\boldsymbol{w}^i_t \in \mathbb{R}^k$ is the discount weights vector of the k objects, and the weight of object category j with target category c is $w^{jc} = 1/sqrt(v_{jc})$. $\boldsymbol{D}_{prior} = \{d_{jc}\} \in \mathbb{R}^{N_c \times N_c}$ is the pre-computed category-to-category prior distance matrix. $\boldsymbol{V}_{prior} = \{v_{jc}\} \in \mathbb{R}^{N_c \times N_c}$ is the pre-computed category-to-category distance variance matrix, which serves to reduce the weight of the objects having little spatial relations to the target category. Fig. 3(b) also intuitively demonstrates how the pre-computed prior matrices help in general. The green cross highlights the relationship between *sink* and *toilet*, that they have a relatively small average distance d_{jc} and a relatively small variance v_{jc}. This indicates the prior that *sink* and *toilet* are close with high confidence. For the same reason, the purple cross highlights the relationship between *bed* and *refrigerator*, that they are far away from each other with high probability, since they have a large prior distance d_{jc} and small variance v_{jc}.

For the prior matrices used by the utility module, we collect data from three different sources *i.e.*, BERT [15], CLIP [24] and Gibson [31]. With BERT and CLIP, we use pre-trained models to embed the class name strings and calculate the inter-class distance in the word embedding space as the prior distance. With Gibson, we calculate inter-class distance *only on the train scenes* as the prior distance. Besides, we also calculate the inter-class distance variance matrix on Gibson, which is used to weigh the prior distances.

When it comes to the question of which objects to propagate semantic information to a frontier, we investigate two methods: 1) only select objects within a radius to a frontier and 2) use all objects while adding the inverse distance from an object to the frontier center as an extra weight factor to the semantic utility (soft radius method). For the policy module, we simply have the agent navigate by the geometric utility for the first 50 steps to collect enough observations, and then the agent navigates by semantic utility purely until the end of the episode.

IV. Experiments

A. Experiment Setup

We use Gibson [31] dataset in AI Habitat [27] simulator in our experiments. Specifically, we follow the same settings as SemExp [7], using Gibson tiny split with 25 training scenes and 5 test scenes. Each test scene has 200 episodes. Each episode contains the starting point and the goal category. The episodes are provided by SemExp [7]. The Habitat simulator APIs compute the ground truth path from a starting point to the closest goal category. For the perception setting, we use the simulated RGBD camera with resolution 640x480, field of view 79°, and max depth range of 5 meters. We run all experiments on a Ubuntu 20.04 workstation with a single Nvidia RTX3080 graphics card.

In our experiments, we first evaluate our pipeline under the same setting with SemExp [7] to demonstrate that our method achieves state-of-the-art performance on the task of ObjectNav, even without performing any training on the given 25 training scenes. Then we present a comprehensive ablation study in two aspects. 1) Firstly, We conduct ablation studies on our semantic utility method to demonstrate how

```python
def struct_nav(Env, SemSeg, SLAM, Fusion3D,
  Planner, target):

  Finished = 0
  while not Finished:
    I_rgb, I_depth = Env.get_observation()
    # first stage, update struct. representation
    I_sem, map, sg, pose = update_struct_repr(
    SemSeg, SLAM, Fusion3D, I_rgb, I_depth,target)
    # second stage, do navigation
    nav_goal, Finished = navigate(I_depth, I_sem,
      target, map, sg, pose)
    if not Finished:
      action = Planner.navigate(
        map, pose, nav_goal)
    else:
      action = "STOP"
    Env.execute_action(action)
  return

def update_struct_repr(SemSeg, SLAM, Fusion3D,
  I_rgb, I_depth):

  # process geometric data
  pcl_rgb, pose = SLAM.update(I_rgb, I_depth)
  map = SLAM.project_to_map(P_rgb)
  frontiers = frontier_detect(map)
  # process semantic data
  I_sem = SemSeg.predict(I_rgb)
  pcl_sem = Fusion3D.update(I_sem, pose)
  sg = build_scene_graph(pcl_sem)
  return I_sem, map, sg, pose

def navigate(I_depth, I_sem,
  target, map, sg, pose):

  Finished = 0
  if target in I_sem:
    # if target in this frame, simply moves to it
    nav_goal=get_object_goal(I_depth, I_sem, pose)
    Finished=check_goal_reached(pose, nav_goal)
  else:
    # explore the scene to look for the target
    frontiers=get_frontiers(map)
    nav_goal=select_frontier(frontiers, sg, pose)
  return nav_goal, Finished
```

Pseudo Code 1: **StructNav** (Python script). After receiving the RGBD images from camera sensors, StructNav first updates the structured representation by processing the geometric and semantic information. Then, StructNav enters the *navigation* stage. The agent will move to the goal if the target is in this frame. Otherwise, the agent will navigate to the most promising frontier obtained from our structured representation.

semantic information and other components help in this task. 2) Secondly, to better understand the problem essence and spot the bottleneck of ObjectNav, we present a comprehensive analysis of our modular pipeline: we provide ground truth data to different components of our pipeline and analyze how error accumulates through the data flow.

In the results, we report the results on the metrics proposed by Anderson *et al.* [1] and also used by Habitat Challenge [3], the *Success Rate*, *Success weighted by Path Length* (SPL) and *Distance to Goal* (DTG). The *Success Rate* is the ratio of episodes that the agent successfully navigates to the goal among all test episodes. SPL is defined as

$$\text{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \cdot \frac{\ell_i}{\max\left(p_i, \ell_i\right)} \tag{3}$$

where, l_i is the length of the shortest path between the goal and the target for an episode, p_i is the length of the path taken by the agent in an episode, and S_i is the binary success indicator of the episode. DTG is the average agent's distance to the target object at the end of the episode.

B. Implementation Details

We use RTAB-Map [17] to predict the agent pose and generate a 3D semantic reconstruction of the scene and a 2D occupancy map. When external ground truth odometry is provided, only the map assembler and the map optimizer nodes in RTAB-Map, which are used for point cloud registration and global optimization, are used. Since RTAB-Map does not have native support for semantic SLAM, we launch another standalone RTAB-Map instance to reconstruct the semantic point cloud by 3D fusion.

For the scene graph extraction module, we use DBSCAN [10] to cluster the semantic point cloud to object instances. The maximum distance between two points to be considered neighbors is 1.0, and the minimum number of points in a neighborhood for a point to be considered a core point is 5. All clusters with less than 5 points will be filtered out as outliers. Then we further reduce the false detected instances by non-maximum suppression on all 3D bounding boxes with a threshold of 0.4.

C. Object Goal Navigation on Gibson

In this section, at each step, all agents are given the ground truth pose and share the same Mask R-CNN [13] model with pre-trained checkpoints provided by Detectron2 [30] to predict the corresponding semantic image from the RGB input, and share the same local planner to generate an instant action for the current step, to make a fair comparison. We report our experimental results along with two other recent works, as shown in table I. For SemExp[7], we report their improved results updated in Github. For RegQLearn [12], we report the numbers in their paper.

As demonstrated in the table, our method (StructNav) achieves state-of-the-art performance on all three metrics with a fundamental margin on *Success Rate* and *Success weighted by Path Length* (SPL). Our method outperforms the best previous method SemExp by 6.6% relatively on the success rate metric, by 22.7% relatively on the SPL metric, and by 6.9% relatively on the DTG metric.

D. Ablation Study on Semantic Utility Method

In this section, we present the comprehensive ablations study on the utility modules of semantic frontiers. We provide

Table I: **Results and Method Ablations.** Evaluated on 1000 episodes in Gibson validation split. Numbers reported on three metrics: success rate (Success), success-weighted path length (SPL), distance to goal in meters (DTG)

Method	**Success**	**SPL**	**DTG**
SemExp [7]	0.650	0.330	1.576
RegQLearn [12]	0.637	0.313	1.568
Ours (BERT SemUtil)	**0.693**	**0.405**	1.488
Ours (Gibson SemUtil)	**0.693**	0.383	**1.468**
Ablations			
(GeoUtil)	0.623	0.359	1.890
(CLIP SemUtil)	0.677	0.375	1.606
(SemUtil w/o.Var)	0.689	0.383	1.528
(SemUtil w/o.3DNMS)	0.684	0.379	1.519
(SemUtil w/o.SoftR)	0.624	0.344	1.839

the following ablations: ***(i)*** GeoUtil: only use geometric utility for exploration, *i.e.*, classical frontier-based exploration [32]; ***(ii)*** CLIP SemUtil: Use Language prior computed from CLIP [24]; ***(iii)*** SemUtil w/o. Var: remove variance discount from BERT SemUtil; ***(iv)*** SemUtil w/o. 3DNMS: remove 3D non-maximum suppression from BERT SemUtil and ***(v)*** SemUtil w/o. SoftR: instead of using the soft radius method, compute the semantic utility of a frontier on objects within a radius to it with equal weights.

The result in Table I shows how different components in the utility module contribute to the final performance. Interestingly, our baseline of pure geometric-based frontier exploration achieves a very competitive result on Gibson. Besides, we observe a clear performance drop on (CLIP SemUtil) compared to (BERT SemUtil), which is also interesting since CLIP is trained by both texts and corresponding images. This could imply that the similarity in image features between categories does not necessarily relate to the spatial proximity in indoor scenes.

E. Qualitative Result of Semantic Utility Method

To further explain why and how semantic reasoning is useful in object goal navigation, we visualize and compare the trajectories of the navigation process with and without semantic utility, i.e. GeoUtil and SemUtil as defined in sections IV-CIV-D. The Fig. 4 shows the visualization result of an episode to find *couch* in the test scene Wiconisco.

Compared with the trajectory from frontier-based exploration 4(a), which is designed to explore as much as possible with the shortest path, we could see a clear shortcut from the result of our method 4(b). In fact, at the beginning of the episode, both agents rotate themselves to get observations from their surroundings. Then our method tries to follow the clue from semantic reasoning, navigating to the frontier most related to the target object, to avoid unnecessary exploration in the unknown environment. This qualitative result explains why our method gains a fantastic boost on the metric of *Success weighted by Path Length* (SPL). This could also indicate that semantic reasoning and knowledge transferring are helpful for object goal navigation, and probably other indoor robotic tasks involving common sense knowledge.

F. Ablation Study on StructNav Pipeline

ObjectNav is an intricate problem that involves perception, understanding and interacting with the complex 3D physical world. It remains unclear what is the key bottleneck to solving this problem. In this section, we present the ablation studies on the pipeline scale, i.e., how noise and errors in each module undermine the overall performance of the pipeline in the task of ObjectNav. Our modular pipeline allows us to do comprehensive ablation studies on each module. We inject the stepped ground truth data into the data flow of the pipeline (Fig. 2) and analyze how the performance improves if the pipeline uses ground truth data instead of predicted data from a module. The result is shown in Table II. In the table, **Odom.**, **SemSeg.**, **SG.** columns indicate the following experimental setting:

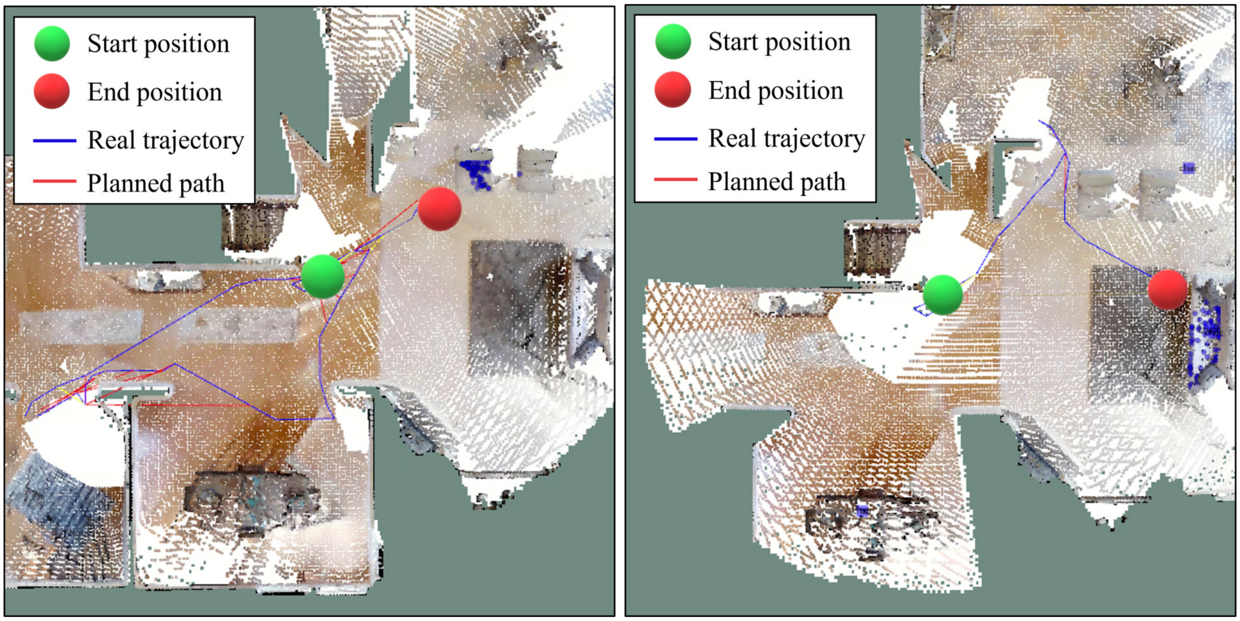
- **Odom.**: GT. indicates using the ground truth pose of the agent; Pred. indicates using the predicted pose by RTAB-Map[17].
- **SemSeg.**: GT. indicates using the ground truth semantic segmentation provided by 3DSceneGraph[2] dataset; Pred. indicates using the predicted semantic segmentation by Mask R-CNN[13].
- **SG.**: GT. indicates using the ground truth scene graph from the Habitat simulator; Pred. indicates using the constructed scene graph by our method. For a fair comparison, objects in the unexplored area will be masked. Otherwise, the agent knows the ground-truth object's position in the unexplored area, which makes it close to the oracle.

It is also worth mentioning that our methods in Table II are directly comparable to the oracle. This is because the ground truth path is calculated by the Fast Marching Method (FMM) with the ground truth 2D occupancy map and ground truth start and end points. Thus, the methods presented in the Table II share the same global planner.

Table II: **Pipeline Ablations**. Results on ObjectNav with stepped ground truth data provided to the pipeline.

Method	**Odom.**	**SemSeg.**	**SG.**	**Success**	**SPL**	**DTG**
StructNav	Pred.	Pred.	Pred.	0.576	0.340	2.123
StructNav	GT.	Pred.	Pred.	0.693	0.405	1.488
StructNav	GT.	GT.	Pred.	0.842	0.563	1.098
StructNav	GT.	GT.	GT.	0.849	0.563	0.947
Oracle	\	\	\	1.000	1.000	0.000

As highlighted in Table II, among the three ablated modules, the major bottleneck of StructNav is the semantic segmentation module which is a pre-trained MaskRCNN [13] in our experiments. With a perfect semantic segmentation model, the success rate boosts about 15% from 0.693 to



(a) GeoUtil; Frontier-based Exploration[32]

(b) SemUtil; Our Method

Figure 4: **Visualization of navigation trajectories in Rviz.** A bird-eye view of the maps and trajectories on a test scene recovered using our approach. For this example, the result is obtained by running GeoUtil and SemUtil on the same episode to navigate to *couch* on the test scene Wiconisco. The green sphere indicates the start of the episode. The red sphere suggests the end of this episode where the agent returns *STOP* action to the simulator. The map's blue dots indicate this episode's detected target object. The blue lines are the real trajectory recorded from the TF frame attached to the agent base, while the red lines are the planned path from the planner.

0.842, which indicates that a better semantic segmentation method is crucial to tackling the ObjectNav problem.

The impact of the visual odometry module is also comparably large. The ground truth odometry lifts the success rate by 11.7% from 0.576 to 0.693 and SPL by 0.065 from 0.340 to 0.405. It is worth mentioning that although our experiment sheds some light on how visual odometry might impact the overall performance of object goal navigation to investigate the sim-to-real gap problem, there is still a gap between the experiment setting and the real robotic system. The simulation's action space is discrete, and it allows sliding when an agent moves against the boundary of the traversable space with a certain heading angle range. This gives more tolerance to localization errors in the simulation than in the real robotic system.

However, the scene graph extraction module shows little impact on the result, which indicates that with a perfect semantic segmentation model, the extracted spatial scene graph is "good enough" for navigation. The second last row shows the result with ground truth odometry, semantic segmentation, and scene graph, which still has a gap of 15.1% in success rate and 0.437 in SPL compared to the Oracle. The gap in success rate could be attributed to the errors in the SLAM system, especially the errors in the 2D occupancy map projected by the 3D reconstruction. The gap in SPL could be more attributed to the variance in scene layout.

V. Conclusion

This paper proposes a training-free pipeline **StructNav** to solve the object goal navigation problem. Built on top of a visual SLAM system, our introduced pipeline constructs a structured representation of the scene, then navigates the agent to the target object by leveraging the information in the observed partial scene and the prior knowledge of the category-to-category spatial relations. Extensive experiments demonstrate that our training-free pipeline can reach state-of-the-art performance on the Gibson dataset with a large margin compared to previous training-heavy methods. Yet, from the pipeline ablations, we find that the semantic model is the major bottleneck of this task. Therefore, it would be interesting to explore the semantic segmentation model and semantic frontier module as a future extension of our work.

References

[1] Peter Anderson, Angel Chang, Devendra Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Košecká, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Zamir. On evaluation of embodied navigation agents. *ArXiv*, 07 2018.

[2] Iro Armeni, Zhi-Yang He, JunYoung Gwak, Amir R Zamir, Martin Fischer, Jitendra Malik, and Silvio Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5664–5673, 2019.

[3] Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*, 2020.

[4] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. Visual navigation for mobile robots: A survey. *Journal of Intelligent and Robotic Systems*, 53:263–296, 11 2008. doi: 10.1007/s10846-008-9235-4.

[5] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[6] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations (ICLR)*, 2020.

[7] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258, 2020.

[8] Raphael Druon, Yusuke Yoshiyasu, Asako Kanezaki, and Alassane Watt. Visual object search by learning spatial context. *IEEE Robotics and Automation Letters*, 5(2):1279–1286, 2020.

[9] Heming Du, Xin Yu, and Liang Zheng. Learning object relation graph and tentative policy for visual navigation. In *European Conference on Computer Vision*, pages 19–34. Springer, 2020.

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[11] Chuang Gan, Jeremy Schwartz, Seth Alter, Damian Mrowca, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, et al. Threedworld: A platform for interactive multi-modal physical simulation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[12] Nandiraju Gireesh, D. A. Sasi Kiran, Snehasis Banerjee, Mohan Sridharan, Brojeshwar Bhowmick, and Madhava Krishna. Object goal navigation using data regularized q-learning, 2022. URL https://arxiv.org/abs/2208.13009.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[14] Lucas Janson and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research*, 34:883 – 921, 2013.

[15] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[16] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[17] Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, 2019. doi: https://doi.org/10.1002/rob.21831.

[18] Steven M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.

[19] Yiqing Liang, Boyuan Chen, and Shuran Song. Sscnav: Confidence-aware semantic scene completion for visual semantic navigation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13194–13200. IEEE, 2021.

[20] Oleksandr Maksymets, Vincent Cartillier, Aaron Gokaslan, Erik Wijmans, Wojciech Galuba, Stefan Lee, and Dhruv Batra. Thda: Treasure hunt data augmentation for semantic navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15374–15383, 2021.

[21] Bar Mayo, Tamir Hazan, and Ayellet Tal. Visual navigation with spatial attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16898–16907, 2021.

[22] Arsalan Mousavian, Alexander Toshev, Marek Fišer, Jana Košecká, Ayzaan Wahid, and James Davidson. Visual representations for semantic target driven navigation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8846–8852. IEEE, 2019.

[23] Anwesan Pal, Yiding Qiu, and Henrik Christensen. Learning hierarchical relationships for object-goal navigation. In *Conference on Robot Learning*, pages 517–528. PMLR, 2021.

[24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[25] Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18890–18900, 2022.

[26] Manolis Savva, Angel X Chang, Alexey Dosovitskiy,

Thomas Funkhouser, and Vladlen Koltun. Minos: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931*, 2017.

[27] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9338–9346, 2019.

[28] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[29] Ayzaan Wahid, Austin Stone, Kevin Chen, Brian Ichter, and Alexander Toshev. Learning object-conditioned exploration using distributed soft actor critic. In *Conference on Robot Learning*, pages 1684–1695. PMLR, 2021.

[30] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[31] F. Xia, Amir Roshan Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9068–9079, 2018.

[32] B. 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'*, pages 146–151, 1997. doi: 10.1109/CIRA.1997.613851.

[33] Sixian Zhang, Xinhang Song, Yubing Bai, Weijie Li, Yakui Chu, and Shuqiang Jiang. Hierarchical object-to-zone graph for object navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15130–15140, 2021.

[34] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699, 2021.