PONI: Potential Functions for ObjectGoal Navigation with Interaction-free Learning

Santhosh Kumar Ramakrishnan[1,2], **Devendra Singh Chaplot**[1], **Ziad Al-Halah**[2], **Jitendra Malik**[1,3], **Kristen Grauman**[1,2]
[1]Meta AI [2]UT Austin [3]UC Berkeley

Abstract

State-of-the-art approaches to ObjectGoal navigation (ObjectNav) rely on reinforcement learning and typically require significant computational resources and time for learning. We propose Potential functions for ObjectGoal Navigation with Interaction-free learning (PONI), a modular approach that disentangles the skills of 'where to look?' for an object and 'how to navigate to (x, y)*?'. Our key insight is that 'where to look?' can be treated purely as a perception problem, and learned without environment interactions. To address this, we propose a network that predicts two complementary potential functions conditioned on a semantic map and uses them to decide where to look for an unseen object. We train the potential function network using supervised learning on a passive dataset of top-down semantic maps, and integrate it into a modular framework to perform ObjectNav. Experiments on Gibson and Matterport3D demonstrate that our method achieves the state-of-the-art for ObjectNav while incurring up to* $1{,}600\times$ *less computational cost for training. Code and pre-trained models are available.*[1]

1. Introduction

Embodied visual navigation is the computer vision problem where an agent uses visual sensing to actively interact with the world and perform navigation tasks [2, 3, 6, 51]. We have witnessed substantial progress in embodied visual navigation over the past decade, fueled by the availability of large-scale photorealistic 3D scene datasets [10, 45, 60] and fast simulators for embodied navigation [33, 51, 60].

ObjectNav has gained popularity in recent years [2, 6, 50]. Here, an agent enters a novel and unmapped 3D scene, and is given an object category to navigate to (e.g., a chair). To successfully solve the task, the agent needs to efficiently navigate to the object and stop near it within a given time budget. This is a fundamental problem for embodied agents which requires semantic reasoning about the world (e.g., TV is in the living room, oven is in the kitchen, and chairs are near tables), and it serves as a precursor to more complex object manipulation tasks [33, 55].

Prior work has made good progress on this task by formulating it as a reinforcement learning (RL) problem and developing useful representations [21, 62], auxiliary tasks [63], data augmentation techniques [37], and improved reward functions [37]. Despite this progress, end-to-end RL incurs high computational cost, has poor sample efficiency, and tends to generalize poorly to new scenes [8, 13, 37] since skills like moving without collisions, exploration, and stopping near the object are all learned from scratch purely using RL. Modular navigation methods aim to address these issues by disentangling 'where to look for an object?' and 'how to navigate to (x, y)?' [13, 36]. These methods have emerged as strong competitors to end-to-end RL with good sample efficiency, better generalization to new scenes, and simulation to real-world transfer [12, 13]. However, since 'where to look?' is formulated as an RL problem with interactive reward-based learning, these methods require extensive computational resources ($\sim$8 GPUs) over multiple days for training.

We hypothesize that the 'where to look?' question for an unseen object is fundamentally a perception problem, and can be learned without any interactions. Based on this insight, we introduce a simple-yet-effective method for ObjectNav – ***P**otential functions for **O**bjectGoal **N**avigation with **I**nteraction-free learning (PONI).* A potential function is a 0-1 valued function defined at the frontiers of a 2D top-down semantic map[2], i.e., the map locations that lie on the edges of the explored and unexplored regions (see Fig. 1, right). It represents the value of visiting each location in order to find the object (higher the value, the better). Given the potential function, we can decide 'where to look?' by simply selecting the maximum potential location.

We propose the *potential function network*, a convolutional encoder-decoder model that estimates the potential function from a partially filled semantic map. Critically, we propose to train it interaction-free using a dataset of

[1]Website: https://vision.cs.utexas.edu/projects/poni/

[2]The 2D semantic map contains the object category per map location.

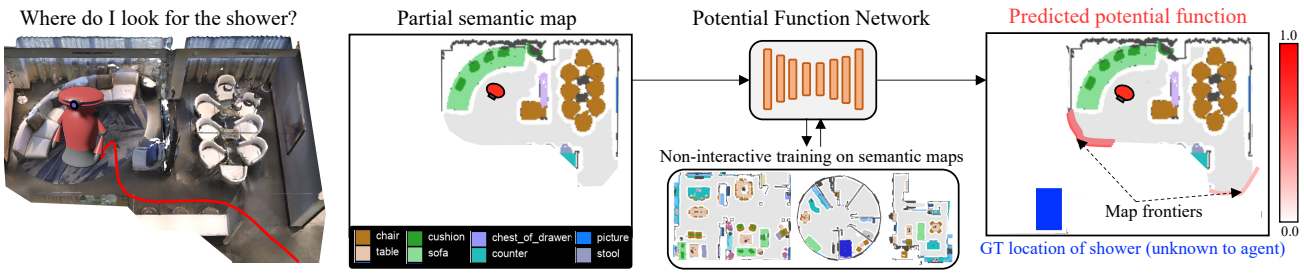

Figure 1. **Potential functions for ObjectGoal Navigation with Interaction-free learning (PONI)**. We introduce a method to decide 'where to look?' for an unseen object in indoor 3D environments. Our key insight is that this is fundamentally a perception problem that can be solved without any interactive learning. We address this problem by defining a *potential function*, which is $[0, 1]$-valued function that represents the value of visiting each location in order to find the object. Given the potential function, we can simply select its argmax location to decide where to look for the object. We propose a potential function network that uses geometric and semantic cues from a partially filled semantic map to predict the potential function for the object of interest (e.g., a shower). We train this network in a non-interactive fashion on a dataset of semantic maps, and integrate it into a modular framework for performing ObjectNav.

top-down semantic maps obtained from 3D semantic annotations [10, 60] (see Fig. 1, center). This is unlike prior work on RL which interactively learns ObjectNav policies by designing reward functions using the same semantic annotations [13,37,63]. Specifically, our network predicts two complementary potential functions by leveraging geometric and semantic cues in the semantic map (e.g., the environment layout, room-object, and object-object relationships). The *area potential function* captures the unexplored areas in the map for efficient exploration, while the *object potential function* is a geodesic-distance based function that helps decide how to reach the object. Once trained, we deploy the potential function network in a modular framework for ObjectNav, where we combine the area and object potential predictions to decide where to look for a goal object.

We perform experiments on the photorealistic 3D environments of Gibson [60] and Matterport3D [10]. Our proposed method outperforms a state-of-the-art modular RL method [13] on Gibson with $7\times$ lower training cost, and a state-of-the-art end-to-end RL method [37] on MP3D with $1{,}600\times$ lower training cost. Our method sets the state-of-the-art on the Habitat ObjectNav challenge leaderboard [6] when compared to previously published methods.

2. Related Work

Visual navigation. Prior work has proposed a variety of visual navigation tasks such as PointNav [2, 50, 51], ObjectNav [6, 50], RoomNav [2, 50], ImageNav [1, 49, 65], AudioNav [15, 16], instruction following [3, 34], and question answering [19, 24]. Research into memory models such as recurrent networks [30, 51, 57], metric maps [18, 25, 29, 46], topological graphs [14, 49], and episodic memory [22] has facilitated significant improvements on these tasks. In this work, we propose a novel strategy to tackle ObjectNav.

ObjectGoal navigation. Recent work on end-to-end RL for ObjectNav has proposed improved visual representations [40, 62], auxiliary tasks [63], and data augmentation techniques [37] to improve generalization to novel scenes. Improved visual representations include semantic segmentations [40], spatial attention maps [38], and object relation graphs [20, 21, 39, 42, 62, 64]. Prior work also learns auxiliary tasks such as predicting agent dynamics, environment states, and map coverage simultaneously with ObjectNav and achieves promising results [63]. Most recently, treasure hunt data augmentation [37] achieved state-of-the-art on ObjectNav by training with artificially inserted objects and improving the RL reward [37].

Modular RL methods for ObjectNav have also emerged as strong competitors to end-to-end RL [13, 36]. They rely on individual modules for semantic mapping, high-level semantic exploration (i.e., where to look?), and low-level navigation (i.e., how to navigate to (x, y)?). The semantic exploration module is learned through RL, yet it is more sample-efficient and generalizes better than end-to-end RL due to modularity and shorter time horizons. We propose a novel strategy for the semantic exploration module. Specifically, we decide 'where to look?' using a potential function network that is trained non-interactively via supervised learning on a dataset of semantic maps. When integrated into a state-of-the-art modular pipeline [13], our method achieves better performance while incurring significantly lower computational cost for training.

Non-interactive learning for navigation. Learning from passive (non-interactive) data has emerged as a good recipe for learning navigation policies. Behavior cloning learns navigation policies from expert action supervision [3, 18, 19, 25, 56], but typically underperforms for complex navigation tasks [11, 14] and may require subsequent RL fine-tuning [18, 19]. Prior work also focuses on pre-training visual representations from image supervision [19, 52],

environment-level representations from videos [47], and learning navigation subroutines from videos [35]. However, they require subsequent policy learning to solve specific navigation tasks of interest. Recent work on self-supervised navigation directly learns distance and semantic scoring functions using passive image [14] and video [11, 26] collections, and uses analytical strategies for ImageNav and ObjectNav. In contrast, we propose a supervised strategy to learn ObjectNav policies from a passive dataset of 2D semantic maps, and demonstrate state-of-the-art performance with high computational efficiency for training.

Waypoint-based navigation. Prior work on waypoint-based navigation repeatedly predicts intermediate goals en route to the target, then uses low-level navigators to reach these intermediate goals [5, 12, 17, 23, 41]. Such policies can be learned through reinforcement learning [12, 17, 36, 44, 59], supervised learning [5, 23, 54], or just analytical planning without any learning [61]. Potential functions can be interpreted as a Q-value function for ObjectNav [36], but predicted only on frontiers and learned in a supervised fashion from collections of top-down semantic maps. The approach in [54] learns to predict value functions at the frontiers for PointGoal navigation, i.e., to a known (x, y) location, in synthetically generated mazes and lab floor-plans. In contrast, we tackle the more challenging ObjectNav task where the goal location is not known a priori, and we focus on diverse real-world indoor environments where semantic reasoning is required to find the goal (e.g., object-object and object-room relationships). We introduce the area potential function to encourage exploration and information gathering, and the object potential function to perform semantic reasoning. We empirically demonstrate the value of these components for ObjectNav in Sec. 5.

3. Approach

We next formally define the ObjectNav task and introduce our method.

3.1. ObjectNav Definition

An agent is tasked with navigating to an object specified by its category label (e.g., chair) in an unexplored environment [6]. At the start of an episode, the agent is spawned at a random navigable location in the environment. At each time-step t, the agent receives 640×480 RGB-D sensor readings s_t, (x, y, θ) odometer readings, and the goal category o. The odometer readings are aggregated over time to obtain the agent's pose p_t (relative to pose at $t = 0$). The agent then executes an action $a_t \sim \mathcal{A}$, where $\mathcal{A}$ consists of **`move_forward`**, **`turn_left`**, **`turn_right`**, and **`stop`**. The agent is required to navigate within $d_s = 1.0\text{m}$ of the object and execute **`stop`** to successfully complete the task. The episode terminates when the agent executes **`stop`**, or if it exceeds a time budget of $T = 500$ steps.

3.2. Method Overview

We propose Potential functions for ObjectGoal Navigation (PONI), a modular architecture for addressing ObjectNav (see Fig. 2). Our model consists of three components. The semantic mapper uses RGB-D and pose sensor readings to build an allocentric semantic map (m_t) of the world capturing which objects are where on the ground plane (Fig. 2, left). The potential function network (π_{pf}) performs geometric and semantic reasoning on top of the semantic map and samples a long-term goal location g_t to explore to find the goal object o. The local policy then navigates the agent towards the long-term goal using analytical path planning, and the process repeats until the episode ends. Our key contribution lies in the design and optimization of π_{pf}. Next, we discuss the individual components.

3.3. Potential Function Network

The potential function network addresses the 'where to look?' problem for an unseen goal object o. It uses the semantic map m_t and the goal object o to predict two potential functions that provide complementary information for ObjectNav (see Fig. 2, center). The area potential function U_t^a serves as a guide for efficient exploration, and helps find unexplored areas in the environment. Analogous to exploration rewards in RL [13, 37], it provides a useful exploration bias for ObjectNav. When the semantic map is not informative (for example, at the start of the episode), U_t^a is critical to quickly explore the environment and gather information. The object potential function U_t^o serves as a guide to efficiently search for the goal object o. When the semantic map is sufficiently informative, U_t^o is critical to perform semantic reasoning and quickly find the object. The potential function network combines these potential functions to sample the long-term goal, effectively trading-off between exploring the environment and finding the object.

We first define the area and object potential functions which are analytically computed using a complete map of the environment during training, and then describe the potential function network architecture which learns to infer them given an incomplete map.

In Fig. 3, we show an example semantic map and its corresponding potential functions. The 'complete semantic map' is the full map of an environment, and the 'partial semantic map' only contains the areas observed by an agent navigating in the environment. We define the area and object potentials only at the map frontiers, i.e., the edges between explored free-spaces (light-gray) and unexplored areas (white) on the partial semantic map (Fig. 3, column 2). Potential functions at the frontiers are sufficient for finding an unseen object since the path to any other unexplored location must pass through the frontier (by definition). In our

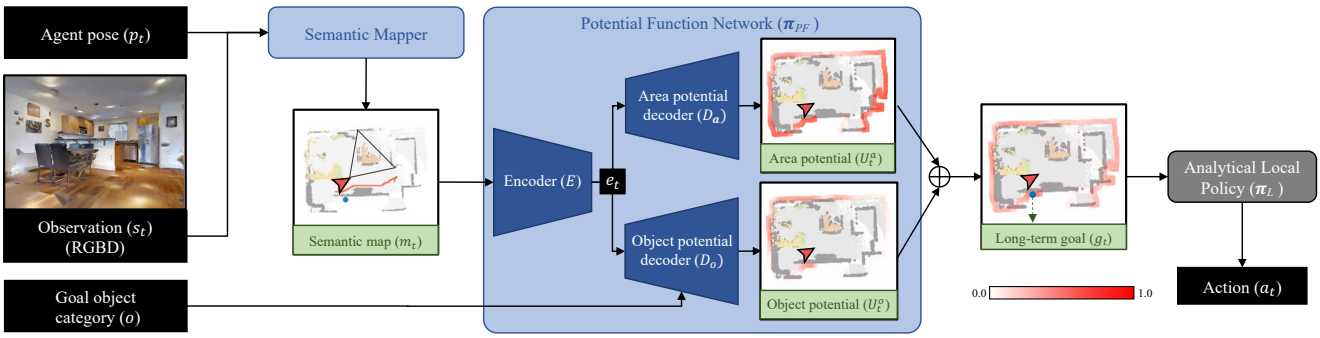

Figure 2. **Architecture for PONI:** Our model consists of three components. The semantic mapper uses RGB-D and pose sensor readings to build an allocentric map m_t of the world. The potential function network π_{pf} uses the semantic map and the goal object category o_t to predict the area and object potential functions. The two potentials are averaged and the maximum location is sampled as the long-term goal. The local policy π_L navigates the agent towards the long-term goal g_t using analytical path-planning.

experiments, we also find that predicting potential functions only at the frontiers is more effective than predicting it at all map locations (see Sec. 5).

Area potential function (U_t^a) The area potential $U_t^a(f)$ at a frontier f measures the amount of free-space left to explore beyond f, i.e., navigable cells which are unexplored in the partial semantic map. To calculate $U_t^a(f)$ on training data, we first group the unexplored free-space cells into connected components $C = \{c_1, \cdots, c_n\}$ using OpenCV [7], and then associate each connected component c_i to the map frontiers. A component c is associated with frontier f only if at least one pixel in c is an 8-connected neighbor [58] of some pixel in f. For each frontier f, we can then calculate the area potential $U_t^a(f)$ as the sum of areas of connected components associated with f, normalized by the total free-space on the complete map.[3] To get the overall area potential function U_t^a for the map, we set all non-frontier pixels to 0, and set a frontier's area potential to the corresponding frontier pixels (as shown in Fig. 3, column 3).

Object potential function (U_t^o) The object potential function U_t^o for object o_t is a function of the geodesic distance between a frontier location x and o_t.

$$U_t^o(o_t, x) = \max\left(1 - \frac{d_g(o_t, x)}{d_{\max}},\ 0.0\right) \tag{1}$$

where d_g is the geodesic distance between x and the 1.0m success zone surrounding the nearest object from category o_t (similar to [6]), and $d_{\max}$ is the distance at which U_t^o decays to 0, selected via validation experiments. This object potential function facilitates efficient object search and is reminiscent of the heuristic in A* search [27].

Next, we define the potential function network that infers the area and object potential functions from the partial semantic map. It consists of three components: the semantic map encoder E, the area potential decoder D_a, and the object potential decoder D_o, as shown in Fig. 2.

[3]For MP3D, we normalize by a fixed constant since the maps are huge.

Map encoder (E) It extracts spatial features from the semantic map: $e_t = E(m_t)$ using a standard UNet encoder with 4 downsampling convolutional blocks [48].

Area potential decoder (D_a) It predicts the area potential function conditioned on the encoder features: $U_t^a = D_a(e_t)$. We use a standard UNet decoder which consists of 4 upsampling convolutional blocks with a sigmoid activation function on the last layer. The output is a 1-channel map that represents the area potential at each map location.

Object potential decoder (D_o) It predicts the object potential function for all valid object categories conditioned on the encoder features: $U_t^o = D_o(e_t)$. We use a standard UNet decoder which consists of 4 upsampling convolutional blocks with a sigmoid activation function on the last layer. The output is a N-channel map representing the object potential for each object category (1 to N) at each location. To get the potential for a specific object category o, we select the corresponding map channel from U_t^o.

Long-term goal sampling We linearly combine the area and object potentials to obtain the overall potential which trades off between discovering unexplored areas and finding the object:

$$U_t = \alpha U_t^a + (1 - \alpha) U_t^o, \tag{2}$$

where $\alpha = 0.5$ is decided via validation experiments. To sample a long-term goal, we zero-out U_t at all explored map locations (except frontiers) since we define the potentials only on the frontiers. Since the geometrically-calculated map frontiers can be noisy during navigation, we retain the predictions from the unexplored locations, providing the model some flexibility in deciding the frontier boundaries. We then sample the maximum location of the filtered U_t as the long-term goal.

3.4. Semantic Mapper

The semantic mapper is responsible for aggregating the semantic information from individual RGB-D observations

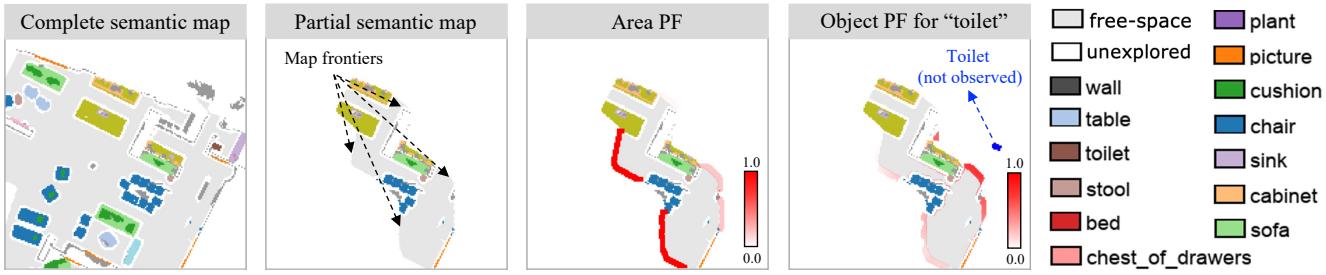

Figure 3. **An example of potential functions (PFs).** From left-to-right, we show the complete semantic map of the environment, the partial semantic map which only contains the parts observed by the agent, the area PF, and the object PF for the category 'toilet'. The area and object PFs in red are defined on the frontiers and overlayed on the partial semantic map. The intensity of red indicates the value of the PF. While the area PF is high for frontiers that lead to more unexplored areas of the environment, the object PF is high for frontiers that are closest to the object. Note that during training these maps are augmented with random translation and rotation (see Sec. 3.6).

from time 0 to t into an allocentric semantic map m_t. We use the mapping procedure from a state-of-the-art semantic exploration method [13]. The depth observations are used to compute point-clouds that are registered to an allocentric coordinate system using the agent's poses $(p_0, \cdots, p_t)$. Each point in the point-cloud is classified into one of N object classes and 1 background class by segmenting the corresponding RGB image using state-of-the-art segmentation models [28, 31]. The point-cloud is projected to the top-down map space by using differentiable geometric operations [29] to obtain the $(N+2) \times M \times M$ semantic map m_t. Channels 1 and 2 correspond to obstacles and explored areas, and the rest contain the N object categories.

3.5. Analytical Local Policy

The local policy π_L navigates the agent to the long-term goal sampled by the potential function network. It uses the Fast Marching Method [53] to compute the shortest path from the current location p_t to the long-term goal g_t using the obstacle channel from the semantic map m_t. The local policy then takes deterministic actions to navigate the agent along this shortest path. This was found to be as effective as a learned local policy in prior work [12, 13].

3.6. Training for Potential Function Network

Our key insight is that the 'where to look?' question for ObjectNav can be treated as a pure perception problem, and learned without any interactions in a simulated 3D environment. Specifically, we train the potential function network π_{pf} on a dataset of semantic maps that are pre-computed from semantic annotations in 3D scene datasets [10, 60].[4] First, we project the semantic point-cloud annotations for a 3D scene to per-floor 2D semantic maps using the publicly available code from Semantic MapNet [9]. We then apply random translations and rotations to the map as a form of data augmentation. This gives an augmented and complete semantic map m^c of the 3D scene with two channels for obstacles and explored regions, and N channels for objects (see column 1 in Fig. 3).

From this complete semantic map m^c, we create a training data tuple (m^p, U^a, U^o) which consists of the partial semantic map, and the area and object potential functions (see columns 2 to 4 in Fig. 3). The partial map m^p is a subset of m^c, and serves as a proxy for the semantic map an embodied agent would observe while navigating in the 3D environment. It is computed as follows. We initialize an all-zeroes 'exploration mask' m^e with the same size as m^c. We then randomly pick two free-space locations on the m^c and compute the shortest path between them using a classical planner [53]. For each location x on the shortest-path, we set a $S \times S$ square patch centered around x to 1 in m^e, indicating that these parts of the map have been explored. We copy values from the m^c to m^p only for locations that are set to 'explored' in m^e. The remaining locations in m^p are left unexplored. The distances to each object from the frontiers in m^p are obtained using shortest-path planning [27, 53] on the complete map m^c. The area and object potential functions are computed as discussed in Sec. 3.3.

Given a set of complete semantic maps $\{m_1^c, m_2^c, \cdots\}$ obtained from 3D scenes, we create the dataset $\mathcal{D}$ for training the potential function network as described above:

$$\mathcal{D} = \{(m_1^p, U_1^a, U_1^o), (m_2^p, U_2^a, U_2^o), \cdots\} \tag{3}$$

The potential function network is then trained to predict (U^a, U^o) from m^p using $\mathcal{D}$. Given a partial map m^p, the map encoder extracts features e, and the area and object potential decoders infer potentials $\hat{U}^a$ and $\hat{U}^o$, respectively. The models are trained end-to-end using the loss $L = L_a + L_c$, where L_a and L_c are pixel-wise mean-squared errors for the area and object potential functions.

$$L_a = \frac{1}{|\mathcal{F}|} \sum_{x \in \mathcal{F}} \left|\left|\hat{U}^a(x) - U^a(x)\right|\right|_2^2 \tag{4}$$

$$L_c = \frac{1}{|\mathcal{F}|N} \sum_{x \in \mathcal{F}} \sum_{n=1}^{N} \left|\left|\hat{U}^o(x,n) - U^o(x,n)\right|\right|_2^2 \tag{5}$$

[4] The same semantic annotations are used for most ObjectNav methods.

Here, $\mathcal{F}$ is the set of frontier pixels, and N is the number of object categories. The potential function network trained here is directly transferred to our PONI model in Fig. 2.

4. Experimental Setup

We perform experiments on Gibson [60] and Matterport3D (MP3D) [10] datasets with the Habitat simulator [51]. 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 which have associated semantic annotations [4]. For MP3D, we use the standard splits of 61 train / 11 val / 18 test scenes.

We use the ObjectNav setup defined in Sec. 3.1. These design choices are consistent with the CVPR 2021 ObjectNav Challenge [6]. Note that the depth and odometer are noise-free in simulation. Only the semantic mapper relies on depth and pose, and this was shown to work well in the real world with noisy pose and depth in prior work [13].

For Gibson experiments, we use the ObjectNav dataset from SemExp [13] which consists of 6 goal categories: 'chair', 'couch', 'potted plant', 'bed', 'toilet', and 'tv'. For MP3D experiments, we use the Habitat ObjectNav dataset [51] which consists of 21 goal categories (listed in supplementary). The semantic maps for training the potential function network use these categories as well.

Evaluation metrics We measure ObjectNav performance using four metrics. **Success** is the ratio of episodes where a method succeeded. **SPL** is the success weighted by the path length and measures the efficiency of the agent's path relative to the oracle shortest path length [6]. **DTS** is the distance (in m) of the agent from the success threshold of the goal object at the end of the episode [6]. **SoftSPL** is a softer version of SPL which measures efficiency based on the progress towards the goal (even with 0 success). It was introduced in the Habitat 2020 PointNav challenge.

4.1. Baselines

We use three types of baselines: non-interactive, end-to-end RL, and modular.

Non-interactive Baselines.

`BC`: We use behavior cloning to train a ResNet-50 based recurrent policy [57] which uses RGB-D, agent pose, and goal object category as inputs.

`Predict-`θ: It classifies the direction to the nearest object. The directions from 0° to 360° are discretized into 8 classes. During ObjectNav, it navigates to the closest frontier along the predicted direction.

`Predict-`xy: It predicts the (x, y) map location of the nearest object (same action space as [13]). During ObjectNav, it navigates to the predicted (x, y) location.

`Predict-`$\mathcal{A}$: It classifies the navigation action to take to reach the nearest object along the shortest-path. The predicted action is executed at each step during ObjectNav.

The **`Predict-*`** baselines are obtained by changing the output parameterization of the potential function network in our PONI model (see Fig. 2). These are trained on the same dataset of semantic maps from Sec. 3.6.

End-to-end RL Baselines.

`DD-PPO` [57]: It represents vanilla end-to-end RL with distributed training over several nodes.

`Red-Rabbit` [63]: It augments **`DD-PPO`** with multiple auxiliary tasks that improve sample efficiency and generalization to unseen environments. This was the winning entry to the Habitat ObjectNav challenge held at CVPR 2021.

`THDA` [37]: It introduces 'Treasure Hunt Data Augmentation' and improves the RL reward and model inputs, which results in better generalization to new scenes—at the time of submission, the state-of-the-art on MP3D.

Modular Baselines.

`SemExp` [13]: This is the state-of-the-art modular method for ObjectNav. It uses RL-based interactive training to learn a policy for sampling long-term goals. It was the winning entry to the ObjectNav challenge held at CVPR 2020.

`FBE` [61]: This is a classical frontier-based exploration approach that builds a 2D occupancy map of the world and navigates to the nearest map frontiers. When the semantic segmentation model detects the goal object, it navigates to the goal using an analytical local policy and executes stop.

`ANS` [12]: This is a modular RL policy trained to maximize area coverage. It uses the same heuristic as **`FBE`** for goal detecting and stopping.

`FBE` and **`ANS`** perform goal-agnostic exploration and help benchmark the value of goal-driven behavior for ObjectNav. For **`DD-PPO`**, **`Red-Rabbit`**, **`THDA`** and **`SemExp`**, we use the publicly available MP3D results on the Habitat ObjectNav challenge leaderboard. For **`SemExp`** on Gibson, and **`ANS`**, we evaluate the pre-trained models released by the authors.

4.2. Implementation Details

On Gibson, we finetune a COCO-pretrained Mask-RCNN [28] on images from the training split of Gibson tiny with 15 object categories from [13]. On MP3D, we use a RedNet [31] segmentation model that is trained in [37], and predict 21 object categories.

The potential function network uses a UNet-based encoder-decoder architecture [48]. This model is trained on a dataset of semantic maps as described in Sec. 3.6. For Gibson, we extract 63 train / 13 val maps from each floor in Gibson tiny. For MP3D, we extract 153 train / 21 val maps from each floor. For each dataset, we pre-compute 400,000 train / 1,000 val (m^p, U^a, U^o) tuples as discussed in Sec. 3.6. We train the model using PyTorch [43] for 3 epochs. We use the Adam optimizer [32] with a learning

Method	Gibson (val)			MP3D (val)		
	Succ. ↑	SPL ↑	DTS ↓	Succ. ↑	SPL ↑	DTS ↓
BC	12.2	8.3	3.90	3.8	2.1	7.5
Predict-$\mathcal{A}$	14.7	13.6	3.45	2.7	1.6	7.8
Predict-θ	69.9	35.7	1.44	29.0	10.6	5.7
Predict-xy	66.9	34.2	1.69	29.4	10.7	5.5
DD-PPO [57]	15.0	10.7	3.24	8.0	1.8	6.9
Red-Rabbit [63]	-	-	-	**34.6**	7.9	-
THDA [37]	-	-	-	28.4	11.0	**5.6**
FBE [61]	64.3	28.3	1.78	22.7	7.2	6.7
ANS [12]	67.1	34.9	1.66	27.3	9.2	5.8
SemExp [13]	71.7	39.6	1.39	-	-	-
PONI (ours)	**73.6**	**41.0**	**1.25**	31.8	**12.1**	**5.1**

Table 1. **ObjectNav validation results on Gibson and MP3D.** We train with 3 seeds and report the average performance. The missing results were not reported in the corresponding papers.

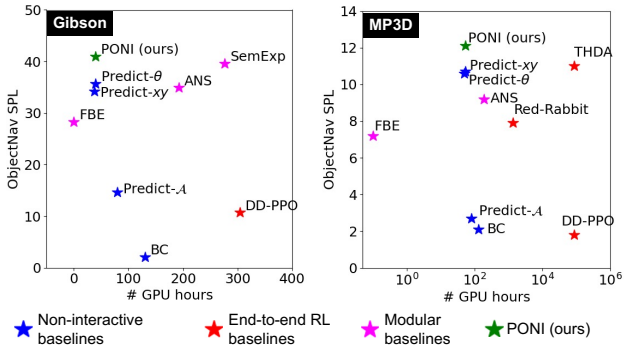

Figure 4. **ObjectNav performance vs. training cost.** We quantify training cost using # GPU hours used to train the model. PONI achieves state-of-the-art performance with up to 1,600× lower training cost. Note: the MP3D plot uses a log-scale for X axis.

rate of 0.001 that is decayed by a factor of 10 after 2 epochs.

During ObjectNav transfer, we found it beneficial to sample a long-term goal with a frequency of T=1 steps for **PONI** (based on val performance). This is unlike prior modular methods [12, 13, 44] which learn to sample long-term goals with T=25 steps. For **Predict-θ** and **Predict-xy**, we found T=25 to work better.

5. Results

Tab. 1 presents the ObjectNav performance on the Gibson and MP3D validation splits. We group the baselines into non-interactive (rows 1 - 4), end-to-end RL (rows 5 - 7), and modular methods (rows 8 - 10).

Comparison to non-interactive baselines. **PONI** is the best method for learning ObjectNav without interactions (see rows 1 - 4, 11 in Tab. 1). On Gibson (val), **PONI** outperforms the next best non-interactive method (i.e., **Predict-θ**) with 3.7% higher success, 5.2% higher SPL, and 0.19m lower DTS. On MP3D (val), **PONI** outperforms the next-best method (i.e., **Predict-xy**) with 2.4% higher success, 1.4% higher SPL, and 0.4m lower DTS. Note that **PONI** is better than **Predict-θ** and **Predict-xy** even though they are all trained on the same dataset of semantic maps with the same encoder backbone. This indicates that it is better to explicitly predict the area and object potential functions for navigation, instead of directly predicting where the objects are. **BC** and **Predict-$\mathcal{A}$** perform poorly suggesting that directly learning to classify shortest-path actions is inadequate.

Comparison to end-to-end RL baselines. **PONI** also outperforms the state-of-the-art in end-to-end RL (see rows 5 - 7 in Tab. 1). **PONI** is significantly better than vanilla RL in **DD-PPO** on both Gibson (val) and MP3D (val). **Red-Rabbit** and **THDA** improve upon **DD-PPO** using techniques such as auxiliary tasks, data augmentation, and better reward-designs. When compared to these, **PONI** achieves more efficient navigation with $1-4\%$ higher SPL and competitive success rates on MP3D (val).

Comparison to modular baselines. **PONI** is the best modular method for ObjectNav (see rows 8 - 11 in Tab. 1). **PONI** convincingly outperforms the goal-agnostic baselines **FBE** and **ANS** on all metrics and datasets, confirming the value of goal-oriented search for ObjectNav. **PONI** is also better than **SemExp**, the previous state-of-the-art modular method, on Gibson (val) even though they rely on the same semantic mapper and analytical local policy. This confirms our hypothesis that the 'where to look?' question can be fundamentally treated as a perception problem and learned without any interactions. See Fig. 5 for a qualitative visualization of **PONI**.

Analysis of computational cost. In Fig. 4, we plot the ObjectNav SPL of different methods as a function of the computational cost for training. We quantify computational cost using the effective number of GPU hours (i.e., # GPUs × training time).[5] See Fig. 4. **PONI** achieves the state-of-the-art on Gibson (val) and MP3D (val) while having one of the lowest computational costs for training. In particular, **PONI** outperforms the prior SoTA on Gibson (**SemExp**) with 7× lower training cost, and the prior SoTA on MP3D (**THDA**) with 1,600× lower training cost. This highlights the value of treating 'where to look?' as a perception problem.

Performance on Habitat Leaderboard. We submitted our best-performing model to the Habitat challenge leaderboard. The results are in Tab. 2. At the time of submission, our method achieved the state-of-the-art relative to prior published entries, confirming our validation results.

Ablation study. We perform an ablation study to understand the impact of different components of **PONI**. There are three key components that contribute to our performance: the object potential function (U^o), the area potential function (U^a), and the fact that they are defined only at the

[5] For end-to-end RL, we use the training cost reported in the papers.

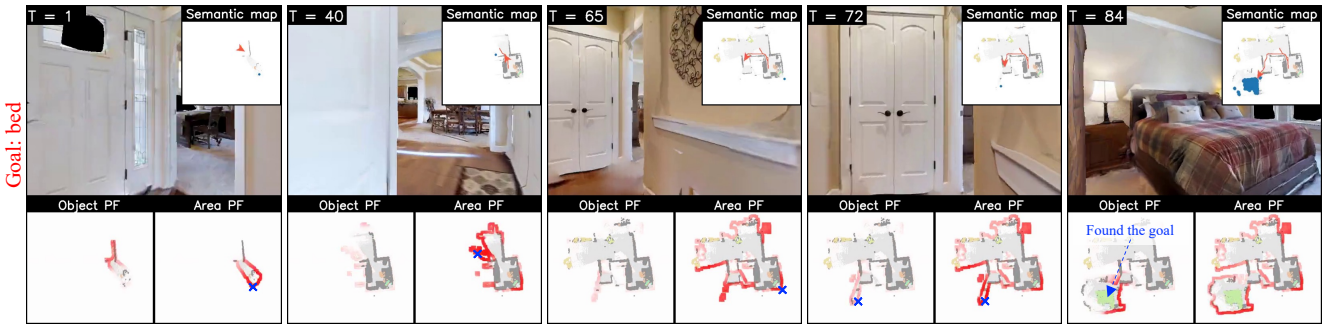

Figure 5. **Qualitative example of navigation using potential functions**. We visualize parts of an ObjectNav episode on Gibson (val), starting from T=1 until the agent finds the goal object (bed). For each step, we show the egocentric RGB view, the predicted semantic map, object and area potential functions. We indicate the maximum location that the agent navigates to using a blue cross on the PF map(s) responsible for the maximum. At the episode start (T=1 to 65), the agent is guided by the area PF which is high near frontiers leading to unexplored areas, allowing it to explore and gather information. The object PF plays a limited role here. After having gathered information, the model predicts higher object PF near the bedroom entrance at T=72, while the area PF remains high at multiple frontiers unrelated to the object location. The agent uses the new signal from the object PF to enter the bedroom and find the bed at T=84. This highlights the value of the two potential functions and how they are combined to perform ObjectNav. Please see supplementary for more examples.

	MP3D (test-standard)			
Method	SPL ↑	SoftSPL ↑	DTS ↓	Success ↑
PONI (ours)	**8.82**	**17.08**	**8.68**	20.01
THDA [37]	8.75	16.96	9.20	21.08
Red-Rabbit [63]	6.22	12.14	9.14	**23.67**
SemExp [13]	7.07	14.50	8.82	17.85
DD-PPO [57]	0.00	0.94	10.32	0.00

Table 2. **Habitat ObjectNav challenge results**. We report the test-standard results from the top-performing *published* methods on the EvalAI leaderboard (as of November 14th, 2021). **PONI** is the state-of-the-art on 3 out of 4 metrics.

PONI ablations				Gibson (val)			MP3D (val)		
U^o	F-only	U^a	GT†	Succ. ↑	SPL ↑	DTS ↓	Succ. ↑	SPL ↑	DTS ↓
✓				58.8	34.9	2.18	30.5	11.6	**5.1**
✓	✓			65.1	37.9	1.76	30.8	12.0	5.2
	✓	✓		72.7	39.4	**1.20**	31.1	11.8	5.3
✓	✓	✓		**73.6**	**41.0**	1.25	**31.8**	**12.1**	**5.1**
✓	✓	✓	✓	86.5	51.5	0.76	58.2	27.5	3.4

Table 3. **Ablation study of PONI.** We study the impact of the object potential function U^o, area potential function U^a, the choice to define potential functions only at the frontiers (F-only), and ground-truth image segmentation (GT). † This is privileged.

frontiers (F-only). We additionally study the impact of using ground-truth image segmentation (GT). In Tab. 3 (rows 1-4), we compare the performance of our complete model with variants that have one or more components missing. The complete model with the 3 components achieves the best performance (row 4, Tab. 3). When U^o is removed (row 3, Tab. 3), both success rate and SPL drop by a good margin, indicating the value of goal-oriented search within **PONI**. When U^a is removed (row 2, Tab. 3), the performance drops even more, which shows the importance of exploration for ObjectNav, echoing findings from recent work that encourage exploration for ObjectNav via rewards [13,37] and tethered policies [63]. In row 5, we augment our complete model with the ground-truth semantic segmentation (GT). We observe that the performance improves significantly on all cases. Since the image segmentation impacts semantic mapping and the stopping behavior for the local policy, segmentation failures are a major source of error for **PONI**.

6. Conclusion

We presented PONI, a modular approach for ObjectNav. Our key idea is to treat 'where to look for an unseen object?' purely as a perception problem and address it without any interactions. To this end, we proposed the potential function network, an encoder-decoder model that predicts two complementary potential functions to decide 'where to look?' for a goal object. We proposed a novel strategy to train this model in a supervised fashion using a dataset of semantic maps obtained from 3D semantic annotations, unlike existing ObjectNav methods which design reward functions for RL-based policy learning. Through experiments on Gibson and Matterport3D, we demonstrated that our method achieves the state-of-the-art for ObjectNav while incurring significantly lower training cost. We hope that our work will spur future research into compute-efficient training for embodied navigation.

7. Acknowledgements

UT Austin is supported in part by the IFML NSF AI Institute, the FRL Cog. Sci. Consortium, and DARPA L2M. K.G is paid as a Research Scientist by Meta AI. We thank the CVPR reviewers and meta-reviewers for their valuable feedback and suggestions. We thank Vincent Cartiller for sharing image segmentation models for MP3D.

References

[1] Ziad Al-Halah, Santhosh K. Ramakrishnan, and Kristen Grauman. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. In *Computer Vision and Pattern Recognition (CVPR), 2022 IEEE Conference on*. IEEE, 2022. 2

[2] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 1, 2

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683, 2018. 1, 2

[4] 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. 6

[5] Somil Bansal, Varun Tolani, Saurabh Gupta, Jitendra Malik, and Claire Tomlin. Combining optimal control and learning for visual navigation in novel environments. In *Conference on Robot Learning*, pages 420–429. PMLR, 2020. 3

[6] 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. 1, 2, 3, 4, 6

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 4

[8] Tommaso Campari, Paolo Eccher, Luciano Serafini, and Lamberto Ballan. Exploiting scene-specific features for object goal navigation. In *European Conference on Computer Vision*, pages 406–421. Springer, 2020. 1

[9] Vincent Cartillier, Zhile Ren, Neha Jain, Stefan Lee, Irfan Essa, and Dhruv Batra. Semantic mapnet: Building allocentric semantic maps and representations from egocentric views. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 964–972, 2021. 5

[10] 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. *Fifth International Conference on 3D Vision (3DV)*, 2017. Matterport3D license available at http://kaldir.vc.in.tum.de/matterport/MP_TOS.pdf. 1, 2, 5, 6

[11] Matthew Chang, Arjun Gupta, and Saurabh Gupta. Semantic visual navigation by watching youtube videos. *arXiv preprint arXiv:2006.10034*, 2020. 2, 3

[12] 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*, 2019. 1, 3, 5, 6, 7

[13] 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, 2020. 1, 2, 3, 5, 6, 7, 8

[14] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural topological slam for visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12875–12884, 2020. 2, 3

[15] Changan Chen, Ziad Al-Halah, and Kristen Grauman. Semantic audio-visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15516–15525, 2021. 2

[16] Changan Chen, Unnat Jain, Carl Schissler, Sebastia Vicenc Amengual Gari, Ziad Al-Halah, Vamsi Krishna Ithapu, Philip Robinson, and Kristen Grauman. Soundspaces: Audio-visual navigation in 3d environments. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 17–36. Springer, 2020. 2

[17] Changan Chen, Sagnik Majumder, Ziad Al-Halah, Ruohan Gao, Santhosh Kumar Ramakrishnan, and Kristen Grauman. Learning to set waypoints for audio-visual navigation. In *International Conference on Learning Representations*, 2020. 3

[18] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning exploration policies for navigation. In *International Conference on Learning Representations*, 2018. 2

[19] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2018. 2

[20] 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. 2

[21] 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. 1, 2

[22] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 538–547, 2019. 2

[23] Chuang Gan, Yiwei Zhang, Jiajun Wu, Boqing Gong, and Joshua B Tenenbaum. Look, listen, and act: Towards audio-visual embodied navigation. In *ICRA*, 2020. 3

[24] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4089–4098, 2018. 2

[25] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Con-*

ference on Computer Vision and Pattern Recognition, pages 2616–2625, 2017. 2

[26] Meera Hahn, Devendra Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No rl, no simulation: Learning to navigate without navigating. *arXiv preprint arXiv:2110.09470*, 2021. 3

[27] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968. 4, 5

[28] 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. 5, 6

[29] Joao F Henriques and Andrea Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8476–8484, 2018. 2, 5

[30] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[31] Jindong Jiang, Lunan Zheng, Fei Luo, and Zhijun Zhang. Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation. *arXiv preprint arXiv:1806.01054*, 2018. 5, 6

[32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[33] 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. 1

[34] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020. 2

[35] Ashish Kumar, Saurabh Gupta, and Jitendra Malik. Learning navigation subroutines from egocentric videos. In *Conference on Robot Learning*, pages 617–626. PMLR, 2020. 3

[36] 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. 1, 2, 3

[37] 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. 1, 2, 3, 6, 7, 8

[38] 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. 2

[39] Mahdi Kazemi Moghaddam, Qi Wu, Ehsan Abbasnejad, and Javen Shi. Optimistic agent: Accurate graph-based value estimation for more successful visual navigation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3733–3742, 2021. 2

[40] 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. 2

[41] Suraj Nair and Chelsea Finn. Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation. *arXiv preprint arXiv:1909.05829*, 2019. 3

[42] Anwesan Pal, Yiding Qiu, and Henrik Christensen. Learning hierarchical relationships for object-goal navigation. In *Conference on Robot Learning*, pages 517–528. PMLR, 2021. 2

[43] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 6

[44] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy anticipation for efficient exploration and navigation. In *European Conference on Computer Vision*, pages 400–418. Springer, 2020. 3, 7

[45] Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, et al. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*, 2021. 1

[46] Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *International Journal of Computer Vision*, 2021. 2

[47] Santhosh K Ramakrishnan, Tushar Nagarajan, Ziad Al-Halah, and Kristen Grauman. Environment predictive coding for embodied agents. *arXiv preprint arXiv:2102.02337*, 2021. 3

[48] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 4, 6

[49] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. In *International Conference on Learning Representations*, 2018. 2

[50] Manolis Savva, Angel X. Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv:1712.03931*, 2017. 1, 2

[51] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia

Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347, 2019. 1, 2, 6

[52] Alexander Sax, Jeffrey O Zhang, Bradley Emi, Amir Zamir, Silvio Savarese, Leonidas Guibas, and Jitendra Malik. Learning to navigate using mid-level visual priors. In *Conference on Robot Learning*, pages 791–812. PMLR, 2020. 2

[53] James A Sethian. Fast marching methods. *SIAM review*, 41(2):199–235, 1999. 5

[54] Gregory J Stein, Christopher Bradley, and Nicholas Roy. Learning over subgoals for efficient navigation of structured, unknown environments. In *Conference on Robot Learning*, pages 213–222. PMLR, 2018. 3

[55] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. *arXiv preprint arXiv:2106.14405*, 2021. 1

[56] Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. Embodied question answering in photorealistic environments with point cloud perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6659–6668, 2019. 2

[57] Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *International Conference on Learning Representations*, 2019. 2, 6, 7, 8

[58] Wikipedia. Pixel connectivity — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Pixel%20connectivity&oldid=1071816307, 2022. [Online; accessed 22-March-2022]. 4

[59] Jimmy Wu, Xingyuan Sun, Andy Zeng, Shuran Song, Johnny Lee, Szymon Rusinkiewicz, and Thomas Funkhouser. Spatial action maps for mobile manipulation. *arXiv preprint arXiv:2004.09141*, 2020. 3

[60] Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018. Gibson license is available at http://svl.stanford.edu/gibson2/assets/GDS_agreement.pdf. 1, 2, 5, 6

[61] 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'*, pages 146–151. IEEE, 1997. 3, 6, 7

[62] Wei Yang, Xiaolong Wang, Ali Farhadi, Abhinav Gupta, and Roozbeh Mottaghi. Visual semantic navigation using scene priors. *arXiv preprint arXiv:1810.06543*, 2018. 1, 2

[63] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary tasks and exploration enable objectgoal navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16117–16126, 2021. 1, 2, 6, 7, 8

[64] 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. 2

[65] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017. 2