MOPA: Modular Object Navigation with PointGoal Agents

Sonia Raychaudhuri[1], Tommaso Campari[2,3], Unnat Jain[4], Manolis Savva[1], Angel X. Chang[1,5]
[1]Simon Fraser University, [2]University of Padova, [3]FBK, [4]Meta AI, [5]Amii
https://3dlg-hcvc.github.io/mopa

Abstract

We propose a simple but effective modular approach MOPA (Modular ObjectNav with PointGoal agents) to systematically investigate the inherent modularity of the object navigation task in Embodied AI. MOPA consists of four modules: (a) an object detection module trained to identify objects from RGB images, (b) a map building module to build a semantic map of the observed objects, (c) an exploration module enabling the agent to explore the environment, and (d) a navigation module to move to identified target objects. We show that we can effectively reuse a pretrained PointGoal agent as the navigation model instead of learning to navigate from scratch, thus saving time and compute. We also compare various exploration strategies for MOPA and find that a simple uniform strategy significantly outperforms more advanced exploration methods.

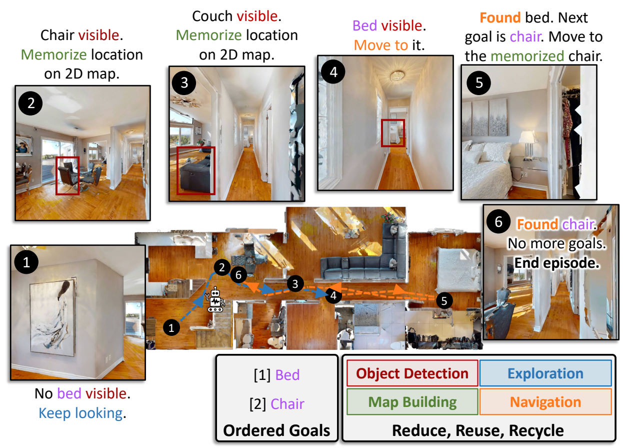

Figure 1. **Approach Overview.** We tackle long-horizon navigation tasks by leveraging their inherent modularity. The agent uses an exploration module to seek the goal in the environment. Once the goal is observed, a navigation module moves the agent towards the goal. While exploring, the agent memorizes objects it sees along the way so it can more efficiently navigate to them later.

1. Introduction

Intelligent agents that can help us in our homes need to tackle tasks such as navigating to objects given different forms of goal specification. Recently, the embodied AI community has studied various navigation approaches, including classical approaches without learning, end-to-end training with deep reinforcement learning (RL), and modular approaches with learned components. End-to-end deep RL agents achieved near-perfect performance on basic navigation tasks such as PointGoal where the agent navigates to a relative goal position [60]. However, navigation tasks where the agent needs to find objects or areas in the environment are far from solved [2, 5, 6, 15, 35, 40]. These tasks require capabilities such as visual understanding, mapping and exploration in addition to basic navigation (see Fig. 1).

In this work, we leverage agents trained on the simpler PointGoal task in the context of more complex longer-horizon navigation tasks. We propose a modular approach called Modular ObjectNav with PointGoal agents (*MOPA*), where each module is responsible for a specific task: (a) *object detection* – to detect objects using the sensory inputs to the agent; (b) *map building* – a semantic map storing observed objects for easy querying; (c) *exploration* – to efficiently search the environment when an object of interest is yet to be located; and (d) *navigation* – to reach a target object that has been located. The first two contribute to acquiring and storing semantic knowledge about the environment, while the latter two enable efficient navigation.

Combining these abilities in a monolithic approach is challenging. Thus, recent work has shifted to modular approaches for semantically-driven navigation [12, 13, 23]. The modular approach allows combinations of learned and traditional modules, reuse of pretrained components, and outperforms end-to-end trained methods when transferring agents developed in simulation to the real world [23].

Despite this interest in modularity, there are few studies of the design choices for different modules. Some work has focused on choices for mapping (or more broadly memory) modules [9, 16, 57], or the impact of better vision modules [32]. Other work has compared exploration modules, finding that a learned exploration policy works better than a frontier-based policy [23], and that heuristic policies can be effective [20, 36]. These works use an analytical path planner to find the best path from the current agent position to the target object. Thus, the design choices in the path planning module have not been studied in a focused manner.

We identify common modules for designing a modular navigation agent and analyze the performance impact of different design choices. Notably, we focus on the path planning (navigation) module and examine different strategies for navigation and their interaction with the exploration modules. Our analysis shows that we can leverage a learned PointNav agent for navigation along with a surprisingly simple random (uniform) exploration policy.

We perform our analysis on the ObjectGoal navigation task and the longer-horizon Multi-Object Navigation (MultiON) [57] task where navigation is to an episode-specific ordered list of objects. The latter enables studying the impact of exploration and navigation strategies for objects that the agent already saw vs. objects not yet seen. In the simple ObjectNav task, our MOPA approach achieves higher Success than the current state-of-the-art modular approaches [23].

In summary, we: (1) propose leveraging pre-trained PointNav agents for more complex ObjectGoal navigation tasks, (2) develop a modular approach MOPA to implement this proposal, (3) show that we achieve significant performance gain by using a simple uniform strategy as the exploration module and PointNav as the navigation module over other complex methods, (4) show that MOPA achieves better Success than previous modular approaches on the ObjectNav task without any training on the ObjectNav dataset.

2. Related Work

Embodied AI tasks. The availability of large-scale datasets such as Matterport3D [10], Gibson [61], and Habitat-Matterport3D (HM3D) [48] along with photo-realistic simulators such as Habitat [44, 52, 55], GibsonEnv [61], AI2-THOR [33, 59] etc. enable a diverse set of Embodied AI tasks [8, 54, 56]. These include *PointGoal navigation* [52, 60, 68] where the target is a single point, *ObjectGoal navigation* [5, 12, 32, 67] where the target is a semantic label of an object, and *instruction following* [2, 15, 35, 40] where the agent follows instructions in natural language. In this work we explore ObjectGoal navigation (ObjectNav) along with *multi-object navigation* (MultiON) [57], which is a generalization of ObjectNav where the agent must reach multiple objects in a sequence. Thus far, methods addressing the MultiON task have used end-to-end trained agents [38, 39, 57]. In contrast, we propose a modular architecture that requires no training yet performs competitively across a range of settings.

Modular navigation in robotic vision. Classical robotic pipelines divide navigation into mapping [19] and path planning [30, 53]. Hybrid approaches using neural high-level policies with model predictive control emerged as more robust and sample-efficient alternatives for navigation [4, 29]. In embodied AI, an initial line of work used largely monolithic reactive or recurrent deep net policies [14, 24, 28, 31, 52]. Modular policies for navigation consisting of separately trained modules using structured neural modular networks have been shown to be more sample efficient [3, 34]. Modular approaches have been shown to be effective and easier to deploy on ObjectNav task as well [7, 12, 23, 70] and unsurprisingly in the MultiON 2022 competition [18] most entries are modular combining learned and rule-based modules. Our modular approach is most similar to Chaplot et al. [12], which extends Active Neural SLAM [11] to have three modules: semantic mapping, goal-oriented exploration and an analytical path planner. It outperformed previous methods in ObjectNav but is ineffective in the MultiON setting where objects are placed randomly, making semantic environment priors not helpful. In contrast, our approach decouples semantic map building from other modules, thus providing better generalization and adaptability to both ObjectNav and MultiON tasks.

Exploration in navigation. Exploration has been studied extensively in both visual navigation and robotics, and it is particularly critical for long-horizon semantic navigation tasks. A common approach is to estimate an exploratory waypoint and navigate towards it [4, 50]. Traditional methods explore the environment based on heuristics, such as selecting points on the frontier between explored and unexplored regions [66]. More recent work uses learning-based methods to generalize to unseen environments better. Notable works include learning end-to-end RL exploration policies from coverage rewards [17, 46, 47] and intrinsic rewards using inverse dynamics [42, 43]. Other approaches leverage first-person depth images [11], predicting semantic maps [12], and topologically-structured maps [13, 25, 58]. Recently, Gervet et al. [23] have shown that a semantically learned exploration policy outperforms a frontier-based policy in ObjectNav. Cartillier et al. [9] employ a pre-exploration setting to build a semantic map, which is later used to explore the free space and navigate to the goal using an analytical path planner. Luo et al. [36] proposes 'Stubborn': a rule-based exploration strategy which outperforms more complex strategies such as frontier-based and semantic exploration. This 'Stubborn' strategy selects and moves towards one of four cardinal directions centered on the agent until it encounters an obstacle. In this work, we focus on non-semantically based exploration methods and compare variants of Stubborn with other rule-based methods.

Reusing PointNav for semantic-based navigation. While using pretrained image encoders as a module is common, there is little work studying the use of pretrained PointNav agents as components in ObjectNav agents. Georgakis et al. [21] use a pre-trained PointGoal model as a local policy while predicting semantic maps outside the agent's field of

	Train		Validation		Test	
	#Scenes	#Ep	#Scenes	#Ep	#Scenes	#Ep
MultiON [57]	61	3.05M	11	1050	18	1050
MultiON 2.0 (Ours)	800	8.00M	30	1050	70	1050

Table 1. **Comparing dataset statistics.** MultiON 2.0 contains significantly larger number of episodes (#Ep) spanning over a diverse set of scenes (#Scenes) compared to MultiON 1.0.

view. We similarly use a PointNav policy as our navigation policy, but we carry out a detailed analysis on both ObjectNav and MultiON tasks to show that this outperforms analytical path planners by piggybacking on the near-perfect performance of PointGoal agents.

3. MultiON 2.0 Dataset

To study our approach in the context of a longer-horizon task planning, we create MultiON 2.0 – a large-scale dataset for the Multi-Object Navigation task. Compared to the original MultiON dataset [57], MultiON 2.0 contains 10x more scenes, uses an additional set of *Natural objects*[1], includes distractor objects, and has longer episodes.

Diversity of objects. The original MultiON dataset [57] contains only cylinder objects of equal size but different (single) color. In MultiON 2.0, we reproduce this Cylinder objects (*CYL*) setup and also include realistic objects that occur naturally in houses. We choose large and visually diverse objects so they are relatively easy to detect and identify. We call this set of objects *Natural objects (NAT)*. The same set of episodes is used to create both NAT and CYL variants by simply swapping cylinder goal objects with natural objects.

Episode generation. We select 800 training, 30 validation, and 70 test scenes from HM3D [48] for use in both tracks (CYL and NAT). Tab. 1 compares the statistics of MultiON and our MultiON 2.0. Episodes are generated by sampling random navigable points as start and goal locations, such that they are on the same floor and a navigable path exists between them. Next, *n* goal objects are inserted at random navigable positions. We have a single object instance for each category. Note that we do not place the objects based on the semantics of the environment, meaning any object can be placed in any room. This choice is deliberate as we want to decouple the need for common-sense priors of where things are located from our study of navigation and exploration policies. We also insert *m* distractor objects in each episode such that $m = (8 - n)$. The distractors come from the same set as the goal objects, and they require the agent to discriminate between goal and non-goal objects, making success by random stumbling onto objects more rare. The minimum geodesic distance between inserted objects is 0.6m in the training split and 1.3m in the validation and test sets to make the latter more challenging overall (details in supplement).

[1] 3D models from https://sketchfab.com/3d-models distributed under permissive licenses.

4. Approach

In the MultiON task, the agent is given the current goal g_i from a set of *n* goals $\{g_1, g_2, ..., g_n\}$. Once the agent has reached g_i and successfully generated the *Found* action, it is given the next goal g_{i+1}. This continues until the agent has found all the goals in the episode. In our MOPA (Fig. 2) approach we employ the following modules: (1) *Object detection* ($\mathcal{O}$), (2) *Map building* ($\mathcal{M}$), (3) *Exploration* ($\mathcal{E}$) and (4) *Navigation* ($\mathcal{N}$). These modules are intuitively woven together. The agent identifies objects ($\mathcal{O}$) by observing the environment and builds a semantic map ($\mathcal{M}$) by projecting the category labels of the observed objects. If the agent has not yet discovered the current goal, g_i, it continues to explore ($\mathcal{E}$) until the current goal has been discovered. The agent then plans a path from its current location to the goal and navigates ($\mathcal{N}$) towards it by generating actions. We experiment with different exploration and navigation strategies to systematically investigate their contribution to the agent performance. Next, we deep dive into each of these modules.

Object detection ($\mathcal{O}$). We consider several object detection approaches based on the type of object we are detecting. For MultiON, we use an object detector (FasterRCNN [51]) trained offline on frames collected from an oracle agent (see supplement). For *CYL-objects*, we fine-tune the FasterRCNN model to detect whether a cylinder is present in a frame and use a k-NN to classify the color. For *NAT-objects*, we fine-tune the FasterRCNN to detect the eight possible objects directly. For our experiments on ObjectNav [62], we use the zero-shot object detector Detic [69].

Map building ($\mathcal{M}$). A cumulative memory representation is key for long-horizon tasks like multi-object navigation. We use the depth channel to project semantic detections onto a 2D top-down grid map of the environment, assuming perfect depth observations and odometry similar to prior work [11, 12, 57]. Each cell in this map spans a 0.2m-by-0.2m square and contains the latest predicted semantic label of the object at that position. Objects once seen remain seen on the map for the length of the episode. This map is used by both the Exploration module to sample exploration goals and the Navigation module to navigate to the goal.

Exploration ($\mathcal{E}$). For any policy to train well, a tradeoff of exploration-exploitation is imperative. This is particularly crucial for long-horizon tasks, where the agent has to tackle ambiguity for large intervals and the current goal is yet to be discovered. Since the exploration policy may select targets that are not reachable, we sample a different exploration goal if the agent does not reach it in α_{exp} steps. We investigate several simple-yet-effective exploration strategies based on success in prior works.

- *Uniform Top-down Sampling.* The agent samples an

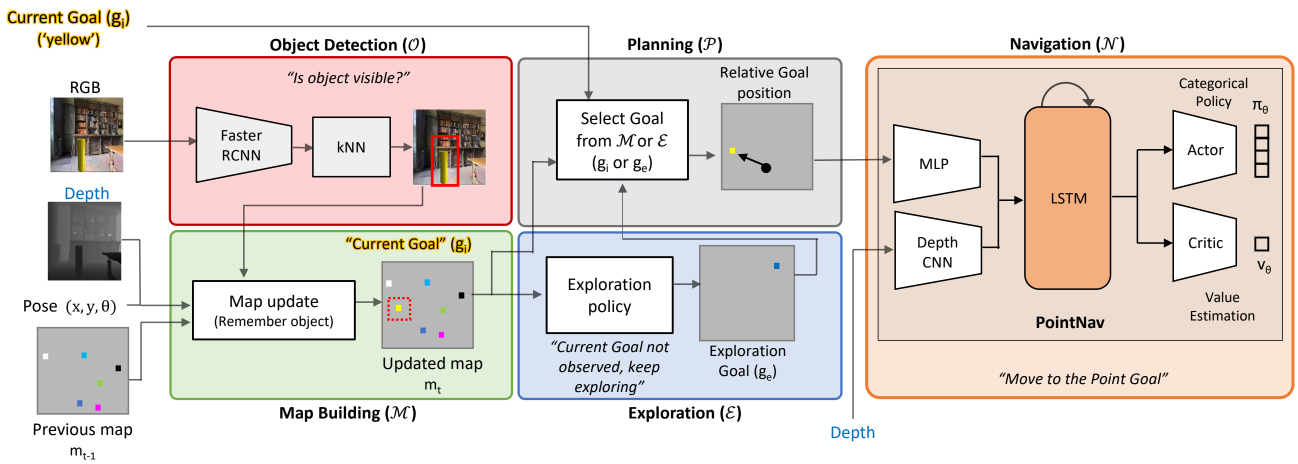

Figure 2. **Architecture.** We adopt a modular approach with PointNav agents (MOPA) to tackle object navigation tasks. The *Object detection* module transforms raw RGB to semantic labels. These are projected onto a top-down semantic map using depth observations by the *Map building* module. The map is passed as input for the *Exploration* module to uncover unseen areas of the environment. A Planning module then selects a relative goal (from either the task goal if on map or an exploratory goal). Finally, a low-level *Navigation* policy predicts the action for the agent to execute.

exploration goal uniformly on a top-down 2D map.

- *Stubborn* [36]. the agent uses the local grid around itself and selects a corner as an exploration goal.
- *Frontier* [66]. the agent navigates to the frontier of the previously explored regions
- *ANS* [11]. Active Neural SLAM (ANS) is a learned method to predict an exploration goal to maximize coverage based on the agent pose on the free-space map. We use the official pretrained global policy.

Planning ($\mathcal{P}$). This module is responsible for selecting either the task goal or the sampled exploration goal ($\mathcal{E}$) to navigate to. We notice label splattering on the semantic map that the agent builds in $\mathcal{M}$. The Planning module selects the centroid of each label cluster as the goal position, which we found to be more effective than selecting a random cell from the cluster. While there are more sophisticated goal selection strategies, such as those based on uncertainty [22], we found this centroid strategy to be sufficient for our MultiON setting.

Navigation ($\mathcal{N}$). Given a relative location from the Planning module, the Navigation module determines the steps to take to reach the location by generating a sequence of actions. This can be achieved by using a trained neural agent or a path planner to determine the path to the relative location. We advocate formulating this problem as a PointNav task and using a pretrained neural policy model from Wijmans et al. [60]. Concretely, our PointNav agent includes a visual encoder with a ResNet50 backbone [26] (for depth observations), a multi-layer perceptron to transform the GPS+Compass coordinates to latent representations and an LSTM [27] to capture state features from previous time steps. All these latent representations are concatenated and transformed using two fully-connected layers *i.e*. the actor and critic heads which give the predicted action logits and state's estimated value, respectively. The low-level actions for interaction with the environment are sampled from the predicted policy logits.

Path planner details. We investigate three analytical path planners: *Shortest Path Follower (SPF)* with access to ground-truth collision map, *breadth first search (BFS)* and *Fast Marching Method (FMM)* on predicted maps. Note that exploration module goals may not be navigable (as that region may not be explored yet). To compensate, we limit the number of steps the navigation module can take to α_{exp}. We stop navigation and resample an exploration goal if the target is not reached within α_{exp} steps. In the case of SPF, we have access to the ground-truth navigation mesh, so we plan a path to the closest navigable point. *Shortest Path Follower* uses a greedy shortest path algorithm on ground-truth navigation meshes from Habitat [52]. It plans a path to the goal location by greedily selecting the best action based on the shortest geodesic distance. *BFS* and *FMM* plan a path to the goal on a 2D occupancy map. The occupancy map uses a similar mapping method as $\mathcal{M}$. The agent builds a collision map by marking the grid cells where it collides. *BFS* searches grid cells adjacent to the agent using breadth-first search until it finds a path to the goal. In contrast, *FMM* finds the shortest path greedily using the 2D occupancy grid. In BFS and FMM, it is possible for the agent to get stuck in corners or crevices so we dilate obstacles to prevent the agent from getting stuck at corners and crevices. This is analogous to the pessimistic collision map from Luo et al. [36]. However, this pessimistic collision map may result in failure to plan a path, in which case we adopt a brute-force 'Untrap' strategy (similar to Stubborn), which keeps generating subsequent Left and Forward (LFLFLF) actions or Right and Forward (RFRFRF) actions until the agent gets unstuck.

5. Experiments

We conduct experiments on MultiON and compare different exploration and navigation strategies in our MOPA framework. We also evaluate MOPA on the single-target ObjectNav task and show that our modular approach with pretrained object-detector can outperform other zero-shot methods that require training a navigation policy.

5.1. Task

We conduct experiments on both MultiON and ObjectNav in Habitat [52]. We focus the bulk of our analysis on MultiON, as the simplified objects allow us to more easily disentangle the effect of the object detector, and the multi-object nature of the task allows us to analyze the performance of the different agents after the first object is found and the map partially constructed.

MultiON. In MultiON [57] the agent needs to find and navigate to a fixed sequence of *n* objects in an unexplored environment. Specifically, the agent has access to (256×256) egocentric RGB image and depth map of the current view, current agent coordinates relative to its starting point in the episode through a noiseless GPS+Compass sensor, and the current goal category at a given time step of the episode. The agent can take one of four actions: *move forward* by 25 cm, *turn left* by 30°, *turn right* by 30°, and *found*. Following Wani et al. [57], the agent has a maximum time horizon of 2500 steps to complete the task. Note that this is longer than standard navigation tasks due to the long-horizon nature of multi-object navigation. The agent must execute the *found* action within 1 meter of each goal for each of the *n* goals, in the right order, to be successful on an episode. A single incorrect *found* action terminates the episode – making the task challenging. We use the widely adopted Habitat platform for our experiments. The agent embodiment is a cylindrical body of 0.1 meter radius and 1.5 meter height.

ObjectNav. In the ObjectNav task, the agent is required to navigate to a single object of a given category. The setup is similar to MultiON, except that the *found* action is replaced by a *stop* action (that concludes the episode), the goal category can have multiple instances in the environment, and the maximum number of actions is set to 500. The agent has a cylindrical body of 0.18 meter radius and 0.88 meter height.

5.2. Metrics

In addition to the standard visual navigation metrics such as *success* (whether the agent can reach all the targets successfully in the given sequence) and *SPL* [1] (Success weighted by inverse Path Length) we use *progress* (proportion of objects correctly found in the episode) and *PPL* (Progress weighted by Path Length) introduced for MultiON by Wani et al. [57]. The SPL and PPL metrics quantify the navigation efficiency in the context of success/progress and increase if the agent trajectory better matches the optimal trajectory. For ObjectNav we use *success* and *SPL*.

5.3. Agents

We use a neural PointNav policy trained using the distributed PPO [60] framework for efficient parallelization on HM3D scenes from Ramakrishnan et al. [48]. We consider three variants of map building agents, ranging from having access to an oracle map to using oracle semantic sensors with ground-truth object detections for map building, to using predicted semantic sensors for map building. The use of oracle sensors and maps allows us to investigate the performance of the exploration and navigation modules without confounding errors from the object detectors.

OracleMap. The *OracleMap* agent has access to the top-down oracle map of the environment directly obtained from the Habitat simulator marked with objects (targets and distractors) observed by the agent during exploration. The ground-truth object locations are directly transformed into grid coordinates to build this map.

OracleSem. Using egocentric depth observations, the *OracleSem* agent builds a semantic map of the environment. We get the semantic labels of the objects (targets and distractors) directly from the Habitat simulator. This agent does not have access to the ground-truth locations of the objects.

PredSem. The *PredSem* agent also builds a top-down semantic map following the same mapping method in OracleSem, but the egocentric semantic labels are predicted using a pre-trained object detection model.

5.4. Implementation Details

We set the confidence threshold of the object detection models as 0.95. We find that a step threshold α_{exp} of 50 works well for all exploration methods. We found that a grid size (l_{r}) corresponding to 10m works best for the uniform sampling-based exploration methods, whereas a grid size (l_{s}) corresponding to 3m works best for the stubborn-based methods. We assume noiseless sensor and actuation similar to prior works [11, 57] in order to decouple the challenges of dealing with noise from the focus of this paper. That said, it should be straightforward to plug in a PointNav agent trained under noisy settings [41] into our method.

5.5. MultiON results

We present results on the test set here (see supplement for validation results). For all experiments, we report mean and standard deviation over 5 runs with random seeds.

Overall performance. Tab. 2 shows the overall comparison of MOPA performance for various agents. We first compare the performance of the PredSem agent and observe that it performs better on the cylinder objects than the natural objects. This is expected since the cylinder objects are easier

	Object Types	Modules		Test			
		$\mathcal{O}$	$\mathcal{M}$	Success	Progress	SPL	PPL
PredSem	CYL	FRCNN	[12]	**52** (± 2)	**66** (± 2)	**21** (± 1)	**27** (± 2)
	NAT	FRCNN	[12]	29 (± 2)	45 (± 2)	11 (± 1)	17 (± 1)
OracleSem	CYL	GT	[12]	81 (± 2)	87 (± 2)	37 (± 1)	39 (± 1)
	NAT	GT	[12]	81 (± 2)	87 (± 2)	37 (± 1)	39 (± 1)
OracleMap	CYL	GT	GT	81 (± 2)	85 (± 2)	36 (± 1)	39 (± 1)
	NAT	GT	GT	81 (± 2)	85 (± 2)	36 (± 1)	39 (± 1)

Table 2. **Performance on MultiON 2.0.** We find that our PredSem agent performs better on cylinder ('CYL') objects than natural ('NAT') objects on the test split of MultiON 2.0.

Method	Navigation ($\mathcal{N}$)	Success	Progress	SPL	PPL
OracleSem	FMM [11]	18 (± 2)	36 (± 2)	11 (± 1)	21 (± 1)
	BFS [18]	21 (± 2)	44 (± 2)	12 (± 1)	22 (± 1)
	SPF *	76 (± 2)	83 (± 2)	**39** (± 1)	**42** (± 1)
	PointNav [48]	**81** (± 2)	**87** (± 2)	37 (± 1)	39 (± 1)

Table 3. **Navigation module comparisons.** We find that a learned PointNav navigation module outperforms other path planners in Success and Progress on the MultiON 2.0 test split. Note that the Shortest Path Follower (SPF) module has access to ground truth navigation meshes and unsurprisingly has highest SPL and PPL.

Method	Exploration ($\mathcal{E}$)	Success	Progress	SPL	PPL
OracleSem	Stubborn	72 (± 2)	80 (± 2)	33 (± 1)	36 (± 1)
	Frontier [66]	72 (± 2)	80 (± 2)	33 (± 1)	35 (± 1)
	ANS [11]	76 (± 2)	83 (± 2)	35 (± 1)	38 (± 1)
	Uniform	**81** (± 2)	**87** (± 2)	**37** (± 1)	**39** (± 1)

Table 4. **Exploration module comparisons.** We find that the Uniform strategy is surprisingly effective, outperforming other complex exploration methods on the MultiON 2.0 test split.

to detect than natural objects with varying shapes, colors and sizes. We then compare the performance with two oracle agents, the OracleSem, which uses ground-truth information in the Object detection module, and OracleMap, which uses ground-truth information in both the Object detection and the Map building modules. All the experiments use Uniform Top-down Sampling ('Uniform') as the Exploration module and PointNav [48] ('PN') as the Navigation module. We find that OracleMap and OracleSem have similar performance. Moreover, these oracle methods have the same performance across CYL and Nat datasets since the object placements are the same in both with only the object labels varying.

Navigation: pretrained PointNav outperforms analytical path planners. We compare different navigation choices for our *OracleSem* agent (see Tab. 3), by fixing the other three modules to use ground truth semantic labels in the Object detection module, Map building from Chaplot et al. [12] and Uniform as the Exploration module. We observe that the pretrained PointNav policy performs significantly better than the analytical path planners in both validation and test sets. We find that the Shortest Path Follower (SPF) planner achieves the closest performance to PointNav which is expected since it has access to the ground-truth navigation meshes. The other two analytical path planners, BFS [18] and FMM [11], perform worse than SPF since they do not have access to the ground-truth obstacle map of the environment and can only plan a path by using a progressively built occupancy map.

Navigation: Analytical path planners are expensive and hard to configure. Analytical path planners are computationally expensive and need handcrafted rules, in contrast to PointNav policy. While PointNav internally learns a representation of obstacles from depth observations, the analytical path planners (BFS and FMM) need to build and update an obstacle map (in addition to the semantic map) at every step. All these handcrafted rules result in longer running times for analytical path planners. We found that a full evaluation run took 12 hours for PointNav but 48 hours for the BFS Path Planner and FMM. This makes them a less desirable choice in navigation tasks compared to neural policies.

Exploration: Uniform outperforms complex Exploration strategies. Tab. 4 compares different Exploration strategies by using ground truth semantic labels in the Object detection module. We find that a simple uniform sampling-based strategy with a fail-safe outperforms the other heuristic-based modules (Frontier and Stubborn) and learned methods (ANS). We observe that since the exploration policy may propose a goal that is not reachable, it is important to have a fail-safe limit (α_{exp}) on the number of steps (see supplement for details). This is especially important for simpler methods such as Uniform and Stubborn as they are more likely to select unreachable goals.

Frontier selects an unexplored location at the frontier in a direction closest to the agent. It tends to maximize coverage in one direction before it starts exploring other directions. We find that when the task goal lies in the opposite direction, this strategy often exhausts the time budget before it can discover the goal. On the other hand, the Uniform strategy enables the agent to frequently pick a new random direction and thus performs better in such cases. In addition, we find that the performance of the frontier exploration method is sensitive to the distance out from the frontier at which the goal is sampled. On the validation set using 2m gave success of 75% vs 41% for 1m and 73% for 3m (see supplement).

Stubborn systematically covers local areas of the environment. We find that it often gets stuck in local pockets in scenes with multiple navigable areas connected by narrow corridors. However, we notice that Stubborn performs better in episodes where the goals are located in a nook or cranny which is often missed by the Uniform sampling method. But the number of such episodes is relatively low in our dataset which explains its overall performance.

We note that it is sufficient for our agent to 'see' the objects from a distance without having to navigate to them in order to be successful. Hence, uniformly sampling locations and moving towards them for a certain number of steps

Method	Trained module	Training scenes	Evaluated on (3ON test set)											
			MultiON (MP3D)				MultiON 2.0 (HM3D)				MultiON 2.0 (HM3D w/o distractors)			
			Success	Progress	SPL	PPL	Success	Progress	SPL	PPL	Success	Progress	SPL	PPL
PredSem	PointNav ($\mathcal{N}$)	MP3D	**38** (± 2)	**53** (± 2)	15 (± 1)	21 (± 1)	**38** (± 2)	**54** (± 2)	**17** (± 1)	**25** (± 1)	**39** (± 2)	**54** (± 2)	**18** (± 1)	**25** (± 1)
No-Map [57]	end-to-end	MP3D	10 (± 2)	24 (± 2)	4 (± 1)	14 (± 1)	0.4 (± 2)	6 (± 2)	0.2 (± 1)	3 (± 1)	1 (± 2)	13 (± 2)	0.5 (± 1)	6 (± 1)
ObjRecogMap [57]	end-to-end	MP3D	22 (± 2)	40 (± 2)	17 (± 1)	30 (± 1)	0.3 (± 2)	10 (± 2)	0.1 (± 1)	0.3 (± 1)	3 (± 2)	18 (± 2)	0.8 (± 1)	6 (± 1)
ProjNeuralMap [57]	end-to-end	MP3D	27 (± 2)	46 (± 2)	**18** (± 1)	**31** (± 1)	0.5 (± 2)	9 (± 2)	0.2 (± 1)	4 (± 1)	4 (± 2)	19 (± 2)	1 (± 1)	8 (± 1)

Table 5. **Transferability.** We show that our MOPA framework transfers better to unseen environments than the end-to-end models from prior work [57]. Our PredSem achieves similar performance on both MP3D and HM3D scenes (with and without distractors) across all metrics, even when we use the PointNav trained on MP3D scenes, outperforming the end-to-end models.

allows for more efficient object discovery than systematically visiting every location. This is especially true for HM3D scenes which are relatively small ($< 100m^2$ for most scenes). This enables our Uniform method to perform better than the others.

The learned Global Policy from Active Neural SLAM performs the closest to our Uniform, and outperforms Stubborn and Frontier. This is aligned with the observation from Chaplot et al. [11] that the trained Global policy learns to predict distant exploration goals leading to higher coverage than Frontier within a time budget.

Better transferability with MOPA. To investigate how our MOPA transfers to unseen environments (scenes different than the ones in training), we evaluate it on MultiON 2.0 (based on HM3D scenes) by using the PointNav agent pre-trained on MP3D scenes. We compared this to three end-to-end models from Wani et al. [57] which were also trained on MP3D scenes. We observe in Tab. 5 that our PredSem outperforms the other methods in both MultiON (MP3D) and MultiON 2.0 (HM3D) episodes, with and without distractors. Moreover, our agent performs consistently across all environments, indicating invariance to environment priors.

Generalization of MOPA on n-ON. We additionally study the ability of MOPA to generalize to n-ON (1ON, 3ON, 5ON) episodes without retraining. Although the performance decreases as we introduce more target objects, with 1ON being the best and 5ON being the worst, the agent still performs considerably well across all n-ONs. The agent achieves a progress of 95% on 1ON, 87% on 3ON, and 76% on 5ON on the test set (see supplement for details).

Effect of spatial map on exploration and navigation. We perform an analysis on MultiON (3ON) to understand the effect of spatial maps for exploration and navigation when the agent needs to backtrack. We find that when the future goals have been already observed and stored in the map the agent can efficiently navigate back to them without having to explore. For these 'seen' goals, we further find the path length to be much shorter in Shortest Path Follower compared to PointNav and FMM, since it has access to the ground-truth navigation meshes and plans the shortest path based on the geodesic distance to the goal. We also find that the Uniform exploration covers the most area before the first goal is reached, thus leading to the discovery of more future goals (see supplement).

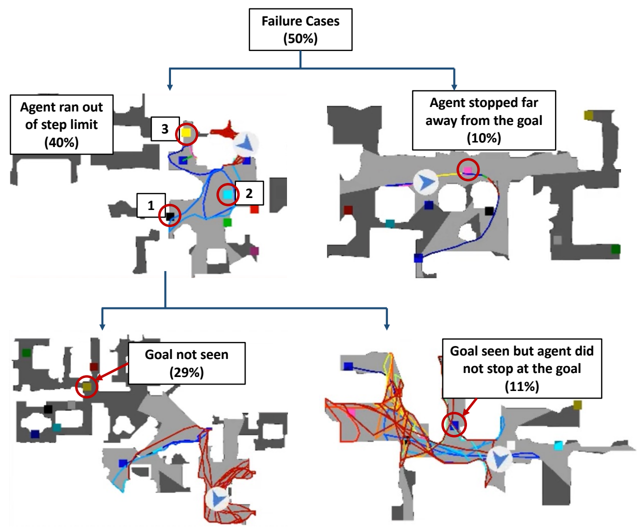

Figure 3. **MultiON performance analysis.** Error modes include the agent running out of steps or stopping far away from the goal. For the former, it either has not yet discovered the goal or has discovered the goal but failed to stop near it.

Failure analysis. In Fig. 3, we analyze the performance of our PredSem agent which achieves 50% success on the 3ON CYL dataset. We find that the agent runs out of the maximum steps quota (2500 steps) in most of the failure cases (40% of episodes). For the remaining 10% of failed episodes, the agent fails to stop (*i.e.* generate the *found* action) within 1m of the goal. This is a limitation of the learned PointNav module. For most episodes where the agent reaches the step quota, it did not yet discover the goal (29% of episodes), which is a limitation of the exploration module. For the other episodes (11% of episodes), the agent discovered the goal but failed to generate the *found* action, which again is a limitation of the PointNav module.

5.6. ObjectNav results

We evaluate MOPA on the single-object navigation task, ObjectNav, where the agent needs to navigate to an instance of a given object category. Experiments are on the validation split of the 2022 ObjectNav challenge [62] (the test split is not publicly available). The dataset is based on HM3D

	Method	Object Detection	Exp ($\mathcal{E}$)	Nav ($\mathcal{N}$)	Validation Success	Validation SPL
1)	OracleSem (Ours)	GT	U	PointNav	64	32
2)	ModLearn[23]	GT	SemExp[12]	FMM	62	32
3)	PredSem (Ours)	Detic[69]	U	PointNav	**30**	14
4)	ModLearn[23]	Mask-RCNN[12]	SemExp[12]	FMM	29	**17**
5)	ModLearn[23]	Detic[69]	SemExp[12]	FMM	27	16
6)	ZSON[37]	CLIP[45]	end-to-end w/ DD-PPO		25	13
7)	OVRL[64]*	Self-supervised pretraining + ObjectNav finetuning			33	12
8)	PIRLNav[49]	end-to-end w/ Imitation Learning+RL finetuning			62	28
9)	OVRL2[63]	end-to-end w/ Imitation Learning using ViT			65	28

Table 6. **ObjectNav performance.** Our PredSem outperforms the modular method ModLearn in Success without additional training on the ObjectNav dataset. It also outperforms end-to-end trained OVRL in SPL demonstrating the effectiveness of our approach. (*OVRL numbers from Majumdar et al. [37])

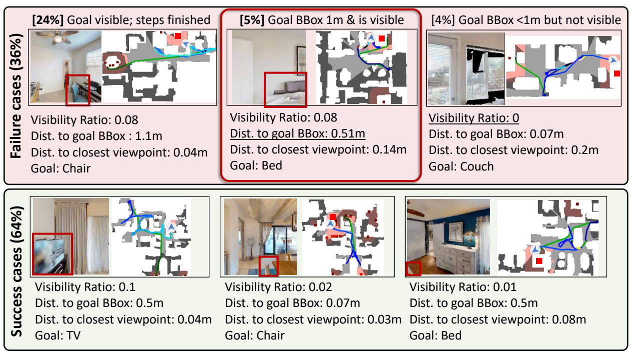

Figure 4. **ObjectNav performance analysis.** Examples of successful (64%) and failed episodes (36%) with OracleSem. Some episodes fail even when the agent is within 1m of the goal bounding box with the goal in sight (top middle), indicating that the viewpoints sampled for determining success in ObjectNav are sparse.

Semantics [65] scenes with 6 object categories and 2000 validation episodes total. As one key advantage of our modular approach is ability to transfer to new domains with no training, we adapt our method to ObjectNav by using a frozen pretrained Detic [69] Object detection module in our PredSem agent. Tab. 6 shows that our OracleSem and PredSem agents outperform ModLearn [23], an approach using learned semantic exploration (SemExp) with FMM as the low-level navigation module, on Success (rows 1 vs 2 and 3 vs 4,5). We also compare our PredSem with ZSON [37], which is trained using DD-PPO [60] on the ImageNav task and evaluated on the ObjectNav task. We find that PredSem outperforms ZSON on both Success and SPL (row 3 vs 6). Next, we compare with fully-supervised SOTA methods in ObjectNav (rows 7-9). Note that, unlike these methods, we do not train any of our components on the ObjectNav dataset. Interestingly we find that PredSem achieves better SPL and similar Success to OVRL [64], signifying the effectiveness of our approach without additional training. However, both PIRLNav and OVRL2 outperform PredSem by a significant margin since they use advanced training strategies and powerful vision transformers respectively. PIRLNav [49] uses the pretrained RESNET encoder from OVRL and trains a policy using Imitation Learning (IL) followed by a second stage of RL finetuning and hence achieves high performance. Similarly, the high performance of the current SOTA method OVRL2 [63] can be attributed to the use of vision transformers (ViT). We observe similar results on ObjectNav [5] with MP3D [10] scenes as well (see supplement).

Failure analysis. We analyze failure cases on ObjectNav similarly to our analysis for MultiON. The failure cases are largely similar (see Fig. 4), with episodes not succeeding primarily due to: i) exceeding the maximum step limit (including cases where the agent did not observe the goal, and cases where it did but failed to navigate close to it); and ii) stopping at a position away from the goal. We found some cases where the definition of success threshold distance to the goal is overly strict. In the ObjectNav evaluation protocol [5], success is defined as the agent stopping close (within 0.1m) to a set of sampled viewpoints each 1m away from the goal object bounding box, an approximation of stopping within 1m of the object. We found episodes where the agent stopped within 1m of the object and with the object in view, but the episode was deemed to have failed due to sparse sampling of the viewpoints, suggesting the ObjectNav evaluation protocol should be improved.

6. Conclusion

We carried out a systematic analysis of our modular approach MOPA to demonstrate that we can effectively leverage pretrained models from other tasks without having to retrain end-to-end models for complex longer-horizon object navigation task. We created a new large-scale dataset for MultiON task and compared various strategies for navigation and exploration. Our experiments show that deploying a PointGoal navigation agent in the MultiON task significantly outperforms analytical path planning. Moreover, a simple exploration strategy in MOPA based on uniform sampling outperforms more complex methods. We believe our work offers insight for more efficient, modular approaches towards solving long-horizon navigation tasks and encourages the community to explore a hybrid combination of transfer learning and simple heuristic-based methods.

Acknowledgements. The authors at SFU were supported by Canada CIFAR AI Chair grant, Canada Research Chair grant, NSERC Discovery Grant and a research grant by Facebook AI Research. Experiments at SFU were enabled by support from WestGrid and Compute Canada. TC was supported by the PNRR project Future AI Research (FAIR - PE00000013), under the NRRP MUR program funded by the NextGenerationEU. We also thank Angelica, Jiayi, Shawn, Bita, Yongsen, Arjun, Justin, Matthew, and Shivansh for comments on early drafts of this paper.

References

[1] 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. 5

[2] 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 *CVPR*, pages 3674–3683, 2018. 1, 2

[3] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural Module Networks. In *CVPR*, pages 39–48, 2016. 2

[4] Somil Bansal, Varun Tolani, Saurabh Gupta, Jitendra Malik, and Claire Tomlin. Combining Optimal Control and Learning for Visual Navigation in Novel Environments. In *CoRL*, pages 420–429. PMLR, 2020. 2

[5] 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, 8

[6] Tommaso Campari, Paolo Eccher, Luciano Serafini, and Lamberto Ballan. Exploiting Scene-Specific Features for Object Goal Navigation. In *ECCV*, pages 406–421. Springer, 2020. 1

[7] Tommaso Campari, Leonardo Lamanna, Paolo Traverso, Luciano Serafini, and Lamberto Ballan. Online Learning of Reusable Abstract Models for Object Goal Navigation. In *CVPR*, pages 14870–14879, 2022. 2

[8] Enrico Cancelli, Tommaso Campari, Luciano Serafini, Angel X. Chang, and Lamberto Ballan. Exploiting Proximity-Aware Tasks for Embodied Social Navigation. In *ICCV*, 2023. 2

[9] Vincent Cartillier, Zhile Ren, Neha Jain, Stefan Lee, Irfan Essa, and Dhruv Batra. Semantic MapNet: Building Allocentric SemanticMaps and Representations from Egocentric Views. In *AAAI*, 2021. 1, 2

[10] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. In *Intl. Conf. on 3D Comput. Vis.*, 2017. 2, 8

[11] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using Active Neural SLAM. In *ICLR*, 2019. 2, 3, 4, 5, 6, 7

[12] Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. Object Goal Navigation using Goal-Oriented Semantic Exploration. In *NeurIPS*, volume 33, pages 4247–4258, 2020. 1, 2, 3, 6, 8

[13] Devendra Singh Chaplot, Ruslan Salakhutdinov, Abhinav Gupta, and Saurabh Gupta. Neural Topological SLAM for Visual Navigation. In *CVPR*, 2020. 1, 2

[14] 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 *ECCV*, pages 17–36. Springer, 2020. 2

[15] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. TOUCHDOWN: Natural Language Navigation and Spatial Reasoning in Visual Street Environments. In *CVPR*, pages 12538–12547, 2019. 1, 2

[16] Shizhe Chen, Thomas Chabal, Ivan Laptev, and Cordelia Schmid. Object Goal Navigation with Recursive Implicit Maps. *arXiv preprint arXiv:2308.05602*, 2023. 1

[17] Tao Chen, Saurabh Gupta, and Abhinav Gupta. Learning Exploration Policies for Navigation. In *ICLR*, 2019. 2

[18] Matt Deitke, Dhruv Batra, Yonatan Bisk, Tommaso Campari, Angel X Chang, Devendra Singh Chaplot, Changan Chen, Claudia Pérez D'Arpino, Kiana Ehsani, Ali Farhadi, et al. Retrospectives on the Embodied AI Workshop. *arXiv preprint arXiv:2210.06849*, 2022. 2, 6

[19] Jorge Fuentes-Pacheco, José Ruiz-Ascencio, and Juan Rendón-Mancha. Visual Simultaneous Localization and Mapping: A Survey. *Artificial Intelligence Review*, 43(1), 2015. 2

[20] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. CoWs on Pasture: Baselines and Benchmarks for Language-Driven Zero-Shot Object Navigation. In *CVPR*, pages 23171–23181, 2023. 2

[21] Georgios Georgakis, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, and Kostas Daniilidis. Learning to Map for Active Semantic Goal Navigation. In *ICLR*, 2021. 2

[22] Georgios Georgakis, Bernadette Bucher, Anton Arapin, Karl Schmeckpeper, Nikolai Matni, and Kostas Daniilidis. Uncertainty-driven Planner for Exploration and Navigation. In *International Conference on Robotics and Automation (ICRA)*, pages 11295–11302. IEEE, 2022. 4

[23] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to Objects in the Real World. *Science Robotics*, 8(79):eadf6991, 2023. 1, 2, 8

[24] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive Mapping and Planning for Visual Navigation. In *CVPR*, pages 2616–2625, 2017. 2

[25] Meera Hahn, Devendra Singh Chaplot, Shubham Tulsiani, Mustafa Mukadam, James M Rehg, and Abhinav Gupta. No RL, No Simulation: Learning to Navigate without Navigating. *NeurIPS*, 2021. 2

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, pages 770–778, 2016. 4

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural computation*, 9(8):1735–1780, 1997. 4

[28] Unnat Jain, Luca Weihs, Eric Kolve, Mohammad Rastegari, Svetlana Lazebnik, Ali Farhadi, Alexander G Schwing, and Aniruddha Kembhavi. Two Body Problem: Collaborative Visual Task Completion. In *CVPR*, 2019. 2

[29] Elia Kaufmann, Mathias Gehrig, Philipp Foehn, René Ranftl, Alexey Dosovitskiy, Vladlen Koltun, and Davide Scaramuzza. Beauty and the Beast: Optimal Methods Meet Learning for Drone Racing. In *ICRA*, pages 690–696, 2019. 2

[30] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic Roadmaps for Path Planning in High-

Dimensional Configuration Spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, 1996. 2

[31] Arbaaz Khan, Clark Zhang, Nikolay Atanasov, Konstantinos Karydis, Daniel D Lee, and Vijay Kumar. End-to-End Navigation in Unknown Environments using Neural Networks. *arXiv preprint arXiv:1707.07385*, 2017. 2

[32] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but Effective: CLIP Embeddings for Embodied AI. *CVPR*, 2022. 1, 2

[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. 2

[34] Satwik Kottur, Jose M. F. Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. Visual Coreference Resolution in Visual Dialog using Neural Module Networks. In *ECCV*, 2018. 2

[35] Jacob Krantz, Erik Wijmans, Arjun Majundar, Dhruv Batra, and Stefan Lee. Beyond the Nav-Graph: Vision and Language Navigation in Continuous Environments. In *ECCV*, 2020. 1, 2

[36] Haokuan Luo, Albert Yue, Zhang-Wei Hong, and Pulkit Agrawal. Stubborn: A Strong Baseline for Indoor Object Navigation. *arXiv preprint arXiv:2203.07359*, 2022. 2, 4

[37] Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. ZSON: Zero-Shot Object-Goal Navigation using Multimodal Goal Embeddings. *NeurIPS*, 35:32340–32352, 2022. 8

[38] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Teaching Agents how to Map: Spatial Reasoning for Multi-Object Navigation. *arXiv preprint arXiv:2107.06011*, 2021. 2

[39] Pierre Marza, Laetitia Matignon, Olivier Simonin, and Christian Wolf. Multi-Object Navigation with dynamically learned neural implicit representations. *arXiv preprint arXiv:2210.05129*, 2022. 2

[40] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping Instructions to Actions in 3D Environments with Visual Goal Prediction. In *EMNLP*, 2018. 1, 2

[41] Ruslan Partsey, Erik Wijmans, Naoki Yokoyama, Oles Dobosevych, Dhruv Batra, and Oleksandr Maksymets. Is Mapping Necessary for Realistic PointGoal Navigation? In *CVPR*, pages 17232–17241, 2022. 5

[42] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, pages 2778–2787, 2017. 2

[43] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-Supervised Exploration via Disagreement. In *ICML*, 2019. 2

[44] Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, et al. Habitat 3.0: A Co-Habitat for Humans, Avatars and Robots. *arXiv preprint arXiv:2310.13724*, 2023. 2

[45] 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 *ICML*, pages 8748–8763. PMLR, 2021. 8

[46] Santhosh K Ramakrishnan, Ziad Al-Halah, and Kristen Grauman. Occupancy Anticipation for Efficient Exploration and Navigation. In *ECCV*, pages 400–418. Springer, 2020. 2

[47] Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An Exploration of Embodied Visual Exploration. *IJCV*, 2021. 2

[48] Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3D dataset (HM3d): 1000 large-scale 3D environments for embodied AI. In *NeurIPS Datasets and Benchmarks Track (Round 2)*, 2021. 2, 3, 5, 6

[49] Ram Ramrakhya, Dhruv Batra, Erik Wijmans, and Abhishek Das. PIRLNav: Pretraining with Imitation and RL Finetuning for ObjectNav. In *CVPR*, pages 17896–17906, 2023. 8

[50] Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel Chang. Language-Aligned Waypoint (LAW) Supervision for Vision-and-Language Navigation in Continuous Environments. In *EMNLP*, pages 4018–4028, 2021. 2

[51] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *NeurIPS*, 28, 2015. 3

[52] 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 *ICCV*, pages 9339–9347, 2019. 2, 4, 5

[53] James A Sethian. Fast-marching level-set methods for three-dimensional photolithography development. In *Optical Microlithography IX*, volume 2726, pages 262–272. International Society for Optics and Photonics, 1996. 2

[54] Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A Foundation Model for Visual Navigation. In *CoRL*, 2023. 2

[55] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *NeurIPS*, 2021. 2

[56] Andrew Szot, Unnat Jain, Dhruv Batra, Zsolt Kira, Ruta Desai, and Akshara Rai. Adaptive Coordination for Social Embodied Rearrangement. In *ICML*, 2023. 2

[57] Saim Wani, Shivansh Patel, Unnat Jain, Angel Chang, and Manolis Savva. MultiON: Benchmarking Semantic Map Memory using Multi-Object Navigation. *NeurIPS*, 33:9700–9712, 2020. 1, 2, 3, 5, 7

[58] Justin Wasserman, Karmesh Yadav, Girish Chowdhary, Abhinav Gupta, and Unnat Jain. Last-Mile Embodied Visual Navigation. In *CoRL*, 2022. 2

[59] Luca Weihs, Jordi Salvador, Klemen Kotar, Unnat Jain, Kuo-Hao Zeng, Roozbeh Mottaghi, and Aniruddha Kembhavi. AllenAct: A Framework for Embodied AI Research. *arXiv preprint arXiv:2008.12760*, 2020. 2

[60] 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 *ICLR*, 2019. 1, 2, 4, 5, 8

[61] Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson Env: real-world perception for embodied agents. In *CVPR*, 2018. 2

[62] Karmesh Yadav, Santhosh Kumar Ramakrishnan, John Turner, Aaron Gokaslan, Oleksandr Maksymets, Rishabh Jain, Ram Ramrakhya, Angel X Chang, Alexander Clegg, Manolis Savva, Eric Undersander, Devendra Singh Chaplot, and Dhruv Batra. Habitat Challenge 2022. https://aihabitat.org/challenge/2022/, 2022. 3, 7

[63] Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. OVRL-V2: A simple state-of-art baseline for Imagenav and Objectnav. *arXiv preprint arXiv:2303.07798*, 2023. 8

[64] Karmesh Yadav, Ram Ramrakhya, Arjun Majumdar, Vincent-Pierre Berges, Sachit Kuhar, Dhruv Batra, Alexei Baevski, and Oleksandr Maksymets. Offline Visual Representation Learning for Embodied Navigation. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023. 8

[65] Karmesh Yadav, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Theo Gervet, John Turner, Aaron Gokaslan, Noah Maestre, Angel Xuan Chang, Dhruv Batra, Manolis Savva, et al. Habitat-Matterport 3D Semantics dataset. In *CVPR*, pages 4927–4936, 2023. 8

[66] Brian Yamauchi. A Frontier-Based Approach for Autonomous Exploration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA). 'Towards New Computational Principles for Robotics and Automation'*, pages 146–151, 1997. 2, 4, 6

[67] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary Tasks and Exploration Enable ObjectGoal Navigation. In *CVPR*, 2021. 2

[68] Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. Auxiliary Tasks Speed Up Learning PointGoal Navigation. In *CoRL*, 2021. 2

[69] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting Twenty-thousand Classes using Image-level Supervision. In *ECCV*, 2022. 3, 8

[70] Minzhao Zhu, Binglei Zhao, and Tao Kong. Navigating to Objects in Unseen Environments by Distance Prediction. In *IROS*, pages 10571–10578. IEEE, 2022. 2