2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Long-HOT: A Modular Hierarchical Approach for Long-Horizon Object Transport

Sriram Narayanan[1,3], Dinesh Jayaraman[2] and Manmohan Chandraker[1,4]

***Abstract*— We aim to address key challenges in long-horizon embodied exploration and navigation by proposing a long-horizon object transport task called Long-HOT and a novel modular framework for temporally extended navigation. Agents in Long-HOT need to efficiently find and pick up target objects that are scattered in the environment, carry them to a goal location with load constraints, and optionally have access to a container. We propose a modular topological graph-based transport policy (HTP) that explores efficiently with the help of weighted frontiers. Our hierarchical approach uses a combination of motion planning algorithms to reach point goals within explored locations and object navigation policies for moving towards semantic targets at unknown locations. Experiments on both our proposed Habitat transport task and on MultiOn benchmarks show that our method outperforms baselines and prior works. Further, we analyze the agent's behavior for the usage of the container and demonstrate meaningful generalization to harder transport scenes with training only on simpler versions of the task.**

I. Introduction

A robot tasked with finding an object in a large environment or executing a complex maneuver must reason over a long horizon. Existing end-to-end reinforcement learning (RL) approaches often suffer in long-horizon navigation tasks due to a combination of challenges: (a) inability to provide exploration guarantees when the object of interest is not visible, (b) difficulty in backtracking previously seen locations and (c) difficulty in planning over long horizons. In this work, we address these issues by proposing a novel long-horizon embodied transport task, as well as modular hierarchical methods for embodied transport and navigation.

Our proposed long-horizon object transport task, Long-HOT, is designed to study modular approaches in the Habitat environment [1]. It requires an embodied agent to pick up objects placed at unknown locations in a large environment and drop them at a goal location, while satisfying load constraints, which may be relaxed by picking up a special container object (Fig. 1). This can be considered an extension to tasks like MultiOn[2] and TDW-transport[3] that also benefit from long-range planning, where the proposed task includes additional constraints like pick-up order and container, which may require complex decision-making and consideration of exploration-exploitation trade-offs.

We argue that modularity is a crucial choice for tackling the above challenge, whereby navigation and interaction policies can be decoupled through temporal and state abstractions that significantly reduce training cost and enhance semantic interpretability compared to end-to-end approaches. This is distinct from existing hierarchical methods for subgoal generation [4], [5] in long horizon tasks, where the expressivity of subgoals is largely limited to goal reaching for embodied navigation and which still face scalability challenges when the task requires long trajectory demonstrations.

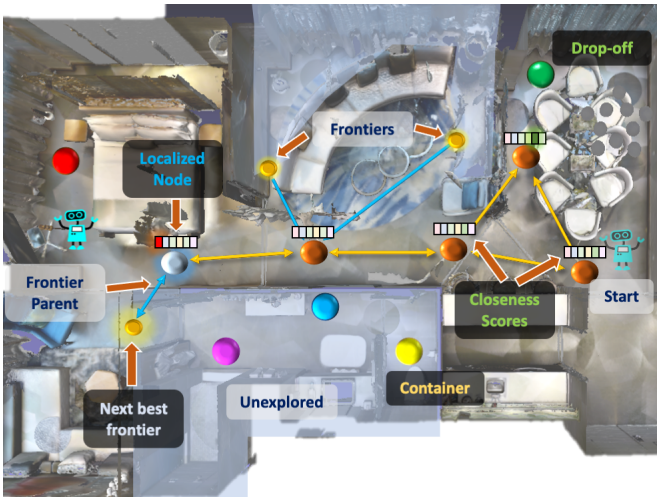

Fig. 1. An instance of object transport problem and our proposed modular approach that builds a topological map. If the agent decides to explore more it would then detect other objects and a container which it can pick up to transport more efficiently, rather than dropping just one object at the goal.

Our modular approach for long-horizon embodied tasks constitutes a topological graph based exploration framework and atomic policies to execute individual sub-tasks (Fig. 2). The higher level planner is a finite state machine that decides on the next sub-routine to execute from one of {`Explore`, `Pickup`, `Drop`} actions. The topological map representation consists of nodes connected in the form of a graph that serves to infuse geometric and odometry information to aid deeper exploration and facilitate backtracking. Unlike methods that utilize 360-degree panoramic images as input [6], [7], [4], we divide every node to aggregate representations from several directions in its vicinity. The representation within a specific node and direction consists of latent features $\mathscr{F}_A$ from a pre-trained encoder, an exploration score $\mathscr{F}_E$ that captures the likelihood of the agent finding an object if it explores a frontier in that direction and object closeness score $\mathscr{F}_O$ that indicates distance to the object within the agent's field of view. We also propose a novel weighted improvement of frontier exploration [8] using the predicted exploration scores.

Unlike methods [5], [4], [7], [9] that completely rely on

[1] NEC Laboratories America, [2] University of Pennsylvania, [3] Carnegie Mellon University, [4] UC San Deigo

either motion planning algorithms [10], [11] or use pure RL for low-level actions [7], [2], our approach uses the best of both worlds with motion planning for point goals within explored regions and RL policies to travel the last-mile towards semantic targets at unknown locations. Indeed, on both Long-HOT and MultiON , we show that our proposed modular hierarchical approach performs significantly better, especially on longer horizons, compared to agents operating without map or other hierarchical approaches that sample navigation subgoals for task completion. Moreover, it realizes a key benefit of modularity, namely, adaptability to harder settings when trained on a simpler version of the task.

II. Related Work

a) Embodied intelligence: There are several simulation environments [1], [12], [13], [3], [14], [15], [16], [17], [18] and associated tasks to study embodied agents in tasks like object goal navigation [19], [20], [21], [22], [2], and several others in this regard [23], [1], [24], [25], [3], [26], [27], [9], [26], [28]. While there are handful of previous works designed for navigation [19], [20], [2] or rearrangement [3], [27], they do not extensively stress test methods with increasing task complexities. We find the typically used flat policy architectures [2] in embodied AI tasks fail completely when executing over longer horizons. Hence, we propose a new benchmark called Long-HOT that has potential to serve as a testbed for, and accelerate the development of novel architectures for planning, exploration, and reasoning over long spatial and temporal horizons.

Our task builds on previous transport tasks defined in embodied intelligence [27], [9], [3], [2] but differs in ways that it requires deeper exploration and long horizon planning. While previous work like [27] focus on identifying state changes using visual inputs to perform rearrangement or [9] use geometrically specified goals in single apartment environments these works operate in minimal exploration scenarios where the focus is shifted more towards perception or interaction with objects. Our task is closest to [3], while [3] focuses on performing transport including physics based simulations, we abstract our interactions and focus more on complex long-horizon planning. Our work extends [2] but rather than focusing on navigation in a predefined sequential fashion, our task requires more complex decision making to determine the order of picking and decide whether to perform a greedy transport if it sees the goal or to explore more in hopes of finding the container for efficient transport.

b) Modular-Hierarchical Frameworks: Solutions to long-horizon tasks typically involved hierarchical[29], [30], [31], [32] policies in reinforcement learning. [5], [4] present an approach where they sample navigation subgoals to be executed by the low-level controller. While these methods can temporally abstract navigation to an extent we find their performance to drop significantly in longer horizon settings. In HTP we show that modularity enables generalization while only training on the simpler versions of the task.

Closer to our work are modular approaches[6], [33], [4], [5] that provide an intuitive way to divide complex long horizon tasks as a combination of individual sub-tasks that can be solved using existing approaches. Chaplot et al. [6] propose a method for image goal navigation by generating a topological map of the scene using 360-degree panoramic images. We operate on perspective images and divide a node representation into segments across different directions.

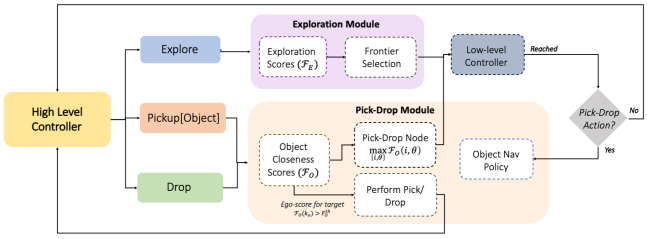

Fig. 2. Overall architecture of our proposed modular hierarchical approach. Our framework consists of three main modules a) a high level controller, b) exploration module and c) a pick-drop module. Each of these are made of components like exploration and closeness score predictor, object-nav policy and low-level controller. We provide details for each individual component in Sec. IV-A and explain the overall framework in Sec. IV-B.

III. Habitat Transport Task

We propose a novel transport task for embodied agents that simulates object search, interaction, and transport in large indoor environments. A robot assistant might be expected to perform such tasks in a warehouse or a hospital. In each episode, the agent must transport K target objects to a specified object goal location in a large partially observed Habitat [1] 3D indoor environment with many rooms and obstacles. The environment also contains a special "container" object (in yellow) that can be used to transport other objects, another special goal object (in green) whose position is the goal location. In our setting, all objects are cylinders of various colors, placed into the environment. Unlike previously studied tasks such as [9], the agent needs to explore the environment to find all objects, and does not have access to their geometric coordinates. At each step, the agent can turn by angle $\alpha = 30^\circ$ to the left or right, move forward by $x = 0.25m$, or execute object "Pickup" or "Drop" actions.

The agent has access to standard egocentric perspective RGB and depth views. Figure 3 shows an example of the agent's view of a scene with a prominent red object. Aside from this, the agent has access to odometry ($P_{\phi xy}$), as well as the hand state O_h and goal state O_g, which indicate if a target/container object is either held by the agent or already at the goal respectively. Following [2], [27], if the agent is within $R = 1.5m$ of any pick-able object and a Pickup action is called, the closest object is removed from the scene and the hand state O_h is updated to include it. For the Drop action, any objects in the agent's hand are dropped near the agent's location. If the goal is within distance R, the goal state is updated to include the object. The agent can hold limited items in its hands at once, and is therefore constrained to carry at most two objects at a time unless it picks up the container, in which case any number of objects may be

carried. Picking up the container requires the agent's hands to be empty. Each episode runs for a maximum of $T = 2500$ timesteps.

This transport task naturally entails additional complexity compared to previously proposed navigation settings, and has properties that are not emphasized in previous benchmark tasks for embodied agents [2], [9], [1], [15]. It includes multiple task phases (searching for, navigating to, and interacting with objects), reasoning about the environment at various scales (such as coarse room connectivity charts over the explored map for planning long trajectories, and fine-grained local occupancy maps for object interaction), accounting for carrying capacity constraints. It also involves dynamically selecting among various sub-task sequences under uncertainty: for example, having found an object far from the goal, should an agent immediately return to drop it off at the goal, or should it look for another object before returning for efficiency?

IV. Hierarchical Transport Policy (HTP)

We now describe our modular policy (Fig. 2) that builds a topological map of the environment and operates at different levels of temporal abstraction. Our framework consists of three modules: a high-level controller, an exploration module, and a pick-drop module. The high-level controller decides on the next high-level action to execute from a set of $\mathscr{A}_H$ ={`Explore`, `Pickup[Object]`, `Drop`} actions. The appropriate sub-task module then takes over to perform the given high-level action. The modules are made of several functions which we describe briefly in Sec. IV-A and then provide details on how those components are connected to our overall framework.

A. HTP Model Components

An overview of our model components along with their connections in the HTP framework is shown in Fig. 3. Our sub-task modules consist of the following components: a) a topological graph builder, b) exploration score predictor c) object-closeness predictor, d) object navigation policy and e) a low-level controller. We now describe all of them in detail.

1) Topological Graph Builder: This function is responsible for creating a topological map of the environment as a graph $G = (V, E)$, where $V \in \mathbb{R}^{N_t \times f_d}$ and E represents spatial nodes and connecting edges respectively. Here, N_t represents number of nodes at timestep t and f_d represents the length of node features. Node features for a node $V_i \in V$ consist of concatenated "node-direction" features $V_{i,\theta}$ corresponding to $D = 12$ directions $\theta \in \{1, ..., D\}$ spanning 360 degrees, centered on the node. These node-direction features $V_{i,\theta}$ in turn are computed by encoding perspective RGB images through an encoder $\mathscr{F}_A$, pretrained in an autoencoder.

At every timestep t, the graph builder updates the map as follows. It takes the pose information ($P_{\phi xy}$) and encoded egocentric image features from $\mathscr{F}_A$ as input. The agent's location P_{xy} is mapped to the nearest node V_i and heading angle P_ϕ. The corresponding features $V_{i,\theta}$ is then updated to the image feature vector. We add a new node V_{i+1} if the agent is not within a distance threshold $l_{th} = 2m$ of existing nodes, and store the corresponding node coordinate P_{xy}. When the agent transitions between two nodes in the graph, the graph is updated to add an edge between them. Similar to [7], we also keep track of the last visited time-step for every node to provide additional context for downstream processing.

2) Exploration and Object Closenes Scores: At every node-direction, indexed by (i, θ), aside from the feature vector above, we store two score predictions: (a) an *exploration score* for frontiers[8] that predicts the likelihood of finding an object by exploring that frontier and (b) an *object closeness score* that indicates distance to various objects in the current field of view.

a) Exploration Score Predictor ($\mathscr{F}_E$):: We explore two variants of this function: (1) a Q learning based graph convolutional network (GCN)[34] that reasons for frontiers over the entire topological graph and (2) a convolutional network (CNN) predicts frontier scores on a per-frame basis.

GCN Exploration Score: This function operates on (1) the current topological map $G = (V, E)$ with associated features as computed above, and (2) a binary mask $M^{N_t \times \theta_d}$ indicating the availability of a frontier at every node-direction, computed using the method described in Sec. IV-B.2. Provided these inputs, we train a graph convolutional network (GCN) [34] to produce reinforcement learning Q-values for each node-direction, representing future rewards for finding objects after visiting each frontier associated with that node-direction. The object-finding reward function r^e_t at every timestep t is:

$$r^e_t = \mathbb{I}_{success} \cdot r^e_{success} + r^e_{slack} + \sum_o \mathbb{I}^o_{found} \cdot r^e_{found}, \qquad (1)$$

where $\mathbb{I}^o_{found}$ is the indicator if object o was found at timestep t, r^e_{found} is the reward for finding a new object, r^e_{slack} is the time penalty for every step that encourages finding the objects faster, $\mathbb{I}_{success}$ is an indicator if *all* objects were found and $r^e_{success}$ is the associated success bonus. We consider the object to be found if the object is in the agent's field of view with distance less than maximum pre-defined distance. Our GCN architecture involves three layers of graph convolution layers and a fully connected layer.

CNN Exploration Score: This variant of the exploration score is computed directly from the agent's current RGBD view. Given this view, a CNN predicts three exploration scores: each score represents the chances of finding an object if the agent explores the farthest frontier available within a corresponding range of angles centered on the current agent heading: -45° to -15°, -15° to $+15^\circ$, and $+15^\circ$ to $+45^\circ$ respectively for the three scores. This CNN is trained with labels set to $\max(\max_o(d_{a,o} - d_{f,o})/5, 0)$ where $d_{a,o}, d_{f,o}$ represent geodesic distance to object o from the agent and frontier respectively. If a frontier is not available then the score is set to 0. These three scores are then stored respectively to three consecutive node-directions $\theta - 1, \theta, \theta + 1$, centered on the current direction θ, and the current node i.

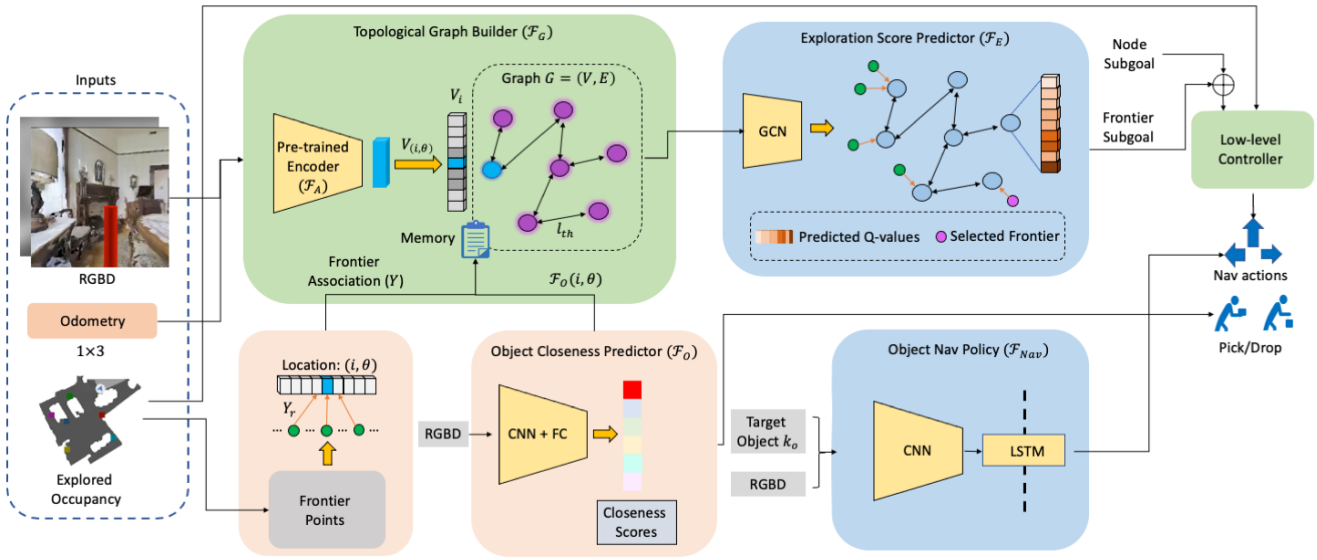

Fig. 3. Shows an overview of the model components used in our overall framework along with their connectivity. It takes egocentric RGBD information along with odometry and explored occupancy map as input and provides low-level actions executed by the agent. At every timestep the framework builds a topological map (Sec. IV-A.1), predicts object closeness scores (Sec. IV-A.2), sample frontiers using exploration scores (Sec. IV-A.2) and executes navigation actions through an object-nav policy (Sec. IV-A.3) or low-level controller (Sec. IV-A.4).

b) Object Closeness Predictor ($\mathcal{F}_O$):: This CNN maps the current RGBD observation I to a "closeness score" for every object. It is trained with supervised learning to predict target closeness labels for each object, which are set to $\max(1-d/5,0)$ where d is the true distance to the object in m. So, objects farther than $5m$ away (or invisible) have labels 0, and very close objects have labels ≈ 1. Each node-direction has an associated closeness score for each object.

3) Object Navigation Policy: Next, we discuss our object navigation policy. Given the RGBD observation I and a one-hot encoding k_o of a target object, the policy must select navigation actions from {`FORWARD`, `TURN-LEFT`, `TURN-RIGHT`} that take it closer towards the target. This policy is trained with the following reward r_t^n [2] at each timestep t:

$$r_t^n = \mathbb{I}_{[reached-obj]} \cdot r_{obj}^n + r_{slack}^n + r_{d2o}^n + r_{collision}^n, \qquad (2)$$

where r_{obj}^n is the success reward if it reaches closer than a threshold distance d_{th} with the target object, r_{slack} is a constant time penalty for every step, $r_{d2o}^n = (d_{t-1} - d_t)$ is the decrease in geodesic distance with the target object and $r_{collision}^n$ is the penalty for collision with the environment.

We train this policy using the proximal policy optimization (PPO) [35] reinforcement learning algorithm, for approximately 40M iterations using 24 simulator instances. We use mini-batch size of 4 and perform 2 epochs in each PPO update. We use other hyper-parameters similar to [2].

4) Low-Level Navigation Controller: Our final module is a low-level controller that takes a goal location (from within the explored regions) to be reached as input. It then plans a path towards the specified goal location using the classical A*[36] planning algorithm using a pre-built occupancy map. Here, for simplicity and faster training speeds, instead of creating a depth projected occupancy every timestep, we use a pre-built occupancy map in all our experiments that is only revealed to a maximum of 5m (no object information) and use ray tracing to avoid revealing more when occlusions are involved.

B. HTP Control Flow

We are now ready to describe how HTP manages the flow of control between these components to perform long-horizon transport tasks. Note that while we describe the HTP algorithm for object transport, we show in Sec. V-A that HTP also works for other embodied navigation tasks.

1) High-Level Controller: The high-level controller (π^H) is a finite state machine. Based on object closeness scores $\mathcal{F}_O$ (Sec. IV-A.2), hand state O_h, and goal state O_g, it selects one subtask from among A_H ={`Explore`, `Pickup[Object]`, `Drop`}. At timestep t, if the next high level action predicted by the controller is different from the current sub-task that is being executed, the controller interrupts the execution, and agent performs the updated high-level action. For example, during exploration if the agent finds an object with closeness score higher than a some threshold it then switches control from exploration to picking the object if the hand is not full or if it holds a container.

2) Weighted Frontier Exploration: If π^H selects the `<Explore>` sub-task, the exploration module is executed. We use a weighted frontier technique based on the predicted exploration score function $\mathcal{F}_E$ (Sec IV-A.2). At each timestep, we calculate the set of frontiers $\mathbf{S}$ using the occupancy information [8]. For each frontier, we assign a parent node-direction $Y_r = (i, \theta^n)_r$, where (i, θ) is the current

localized node-direction and θ^n is calculated based on the angle made by the frontier with the agent. We then calculate a representative frontier $S^{(i,\theta)}$ for node-direction (i,θ), as: $S^{(i,\theta)} = \{s_k : \text{argmin}_k \|s_k - X_c\| \ \forall \ Y_k = (i,\theta)\}$ where $s_k \in \mathbf{S}$ and X_c is the center of frontiers associated with $Y_k = (i,\theta)$.

At each timestep, the exploration module selects a node-direction (i,θ) from the topological graph G that has the highest exploration score $\mathscr{F}_E$. Its corresponding frontier $S^{(i,\theta)}$ is then set as the goal location for the agent's low-level controller to achieve. The highest-score frontier as goal is recomputed at every timestep, and may switch as new views are observed during exploration.

3) Pick-Drop Module: This module performs the pick or drop actions in the object transport task when the controller π^H selects an action $a_H \in \{\texttt{Pickup[Object]}, \texttt{DropAtGoal}\}$. It selects a node (i,θ) from graph G with the highest object closeness score $\mathscr{F}_O$ for the target object. If the agent is not already in the selected i^{th} node, then its location $P_{xy}(i)$ is set as the goal for the low-level controller. Once the agent is localized to the i^{th} node, it orients in the direction of θ. The module then selects the pickup or drop action whenever the object closeness score $\mathscr{F}_O$ for the target object, based on the current view, exceeds a threshold. The sub-task is successful when the hand state or goal state is changed accordingly and the controller π^H predicts the next high level action to execute. The module executes until it performs the pick/drop or for a maximum of T_p steps, after which control is given back to the high-level controller π^H.

V. Experiments

We evaluate all approaches on two tasks set in photo-realistic Matterport3D scenes (MP3d)[37] in Habitat [1]: Long-HOT transport, and Multi-ON [2] object navigation.

TABLE I
SPECIFICATION PARAMETERS FOR OUR TASK LEVELS

Level	goal-range(m)	obj-dist-min/max(m)
default	(2, 15)	2/10
hard	(5, 20)	5/20
harder	(5, 30)	5/30

Long-HOT: We split MP3D[37] scenes into train, validation, and test scenes. Each task configuration consists of a specific configuration of objects, container, goal location, and agent starting location and pose. We generate 10k training task configurations and 3k each of validation and test configurations. Tab. I shows settings for different task levels used in our experiments. We train all methods on *default*-level tasks on 61 scenes. After training, we evaluate them on 15 disjoint test scenes in the *default* setting and perform evaluation on *large* scenes that have at least one dimension $> 40m$ and sample *default*, *hard* and *harder* level tasks from Tab. I.

Baselines: We compare our proposed transport policy with several baseline methods and ablations. These include NoMap, which maps RGBD image I, hand state O_h and goal state O_g directly to low-level robot actions, and OracleMap, which improves NoMap by assuming additional access to a ground truth 2D occupancy map of the $10m \times 10m$ area centered on the agent in the overhead view. We also evaluate two versions of OracleMap: with occupancy alone ("Occ"), and with extra annotated true locations of the task-relevant objects and container ("Occ+Obj"). Additionally, we consider OracleMap-Waypoints, which represents a popular hierarchical approach in embodied navigation, and MultiOn Baselines, which adapt the ProjNeuralMap baseline from [2] that projects perspective features in top view to our task. Finally, we study three variants of our method with different exploration strategies: NearestFrontier, CNN, GCN.

TABLE II
STANDARD LONG-HOT TASK: COMPARISON OF THE PROPOSED HTP ALONG WITH THE BASELINES IN STANDARD TRANSPORT SETTINGS.

Model	%Success↑	%Progress↑	SPL↑	PPL↑	Episode energy↓
OracleMap (Occ)	56	68	34	37	0.34
OracleMap (Occ+Obj)	92	95	74	75	0.05
OracleMap-Waypoints	85	90	51	52	0.10
NoMap	43	63	26	32	0.41
ProjNeuralMap[2]	43	60	24	29	0.42
HTP - NearestFrontier	52	66	24	29	0.38
HTP - CNN	56	**72**	25	31	**0.32**
HTP - GCN	**59**	70	**28**	**33**	0.33

Metrics: We use standard evaluation metrics following previous works [27], [2], [38], [39], [14], [23], [1] such as Success Rate, Progress Rate, Success/Progress weighted by Path Length (SPL/PPL), and Episode Energy.

A. Results

a) Standard Long-HOT Task:: Table II shows the results of evaluations on Standard Long-HOT task for 1000 test episodes generated using the *default* task level. All variants of HTP clearly outperform NoMap and ProjNeuralMap on all five metrics.

Are hierarchies good? HTP-NearestFrontier outperforms non-hierarchical baselines, demonstrating the importance of our modular hierarchical approach with separate policies for different task phases and a topological map. OracleMap-Waypoints performs worse than flat OracleMap ("Occ+Obj"), indicating that combining learning with planning like HTP is more robust and fares better than alternative hierarchical approaches. Note that OracleMap methods have access to ground truth map information and are not directly comparable with HTP, but can be compared among themselves.

Does weighted frontier exploration work? Among HTP variants, both HTP-GCN and HTP-CNN, which use predicted scores for weighted frontier exploration, clearly outperform HTP-NearestFrontier. Between them, GCN and CNN are roughly equivalent in this setting.

b) Large Long-HOT Task:: We evaluate the trained policies on more challenging settings in large scenes to test generalization and the benefits of hierarchy and modularity. Scenes used in Large Long-HOT have some overlap with training scenes, but not with the same episodes due to limited availability of large scenes in the disjoint test set. Our observation of better generalization stands since all methods

TABLE III

LARGE LONG-HOT TASK: SHOWS GENERALIZABILITY OF METHODS TO MORE DIFFICULT SETTINGS WITHIN THE SAME TASK. THE METHODS WERE TESTED ON 26 HABITAT SCENES WITH AT LEAST ONE DIMENSION $> 40m$. ALL METHODS WERE TRAINED ON DATASET GENERATED WITH *default* TASK CONFIGURATION (MAX. RADIUS 15M) AND TESTED ON *hard* (MAX. RADIUS 20M) AND *harder* (MAX. RADIUS 30M) SETTINGS.

	Model	%Success↑	%Progress↑	SPL↑	PPL↑
Default	OracleMap (Occ)	26	45	18	27
	OracleMap (Occ+Obj)	85	91	65	67
	OracleMap-Waypoints	80	88	38	39
	NoMap	39	59	25	31
	HTP-NearestFrontier	41	54	17	20
	HTP-CNN	**55**	**68**	**26**	**30**
	HTP-GCN	46	58	19	22
Hard	OracleMap (Occ)	3.6	17	2.3	10
	OracleMap (Occ+Obj)	46	69	29	38
	OracleMap-Waypoints	48	72	18	24
	NoMap	5.2	24	2.6	10
	HTP-NearestFrontier	22	40	8.1	14
	HTP-CNN	**33**	**53**	**13**	**20**
	HTP-GCN	27	47	9.4	16
Harder	OracleMap (Occ)	2	12	1.3	7.7
	OracleMap (Occ+Obj)	26	48	16	26
	OracleMap-Waypoints	31	59	12	19
	NoMap	2.8	16	1.3	8.0
	HTP-NearestFrontier	15	32	5.5	12
	HTP-CNN	21	**42**	**8.6**	**16**
	HTP-GCN	**22**	39	8.0	14

TABLE IV

MULTION BENCHMARK RESULTS FOR 3-OBJECT NAVIGATION

Model	%Success↑	%Progress↑	SPL↑	PPL↑
OracleMap (Occ)	16	36	12	27
OracleMap (Occ+Obj)	48	62	38	49
NoMap	10	24	4	14
FRMQN[40]	13	29	24	24
SMT[41]	9	22	7	18
ProjNeuralMap[2]	27	46	18	31
ObjRecogMap[2]	22	40	17	30
Lyon[42]	**57**	**70**	**36**	**45**
HTP-CNN (Ours)	56	69	30	36
HTP-GCN (Ours)	**57**	**70**	27	33

have the same advantage, but others suffer severe drops compared to ours.

Table III shows that NoMap and OracleMap perform poorly on *hard* and *harder* tasks, with NoMap's %Success dropping to 5.2% and 2.8%. HTP methods perform better, achieving up to 33% and 22% on these tasks. Despite having access to ground truth map information, OracleMap-Waypoints is outperformed by HTP, which benefits from modular design and weighted frontier exploration. OracleMap-Waypoints performs better than OracleMap (Occ+Obj), indicating the benefits of subgoals in long-horizon settings. Long-HOT is a challenging task that stress tests planning, exploration, and reasoning over long spatial and temporal horizons. Our HTP approach, which uses hierarchical policies and topological maps, is a first step towards addressing these challenges. Our atomic policies make HTP transferable to other tasks and it would be interesting to see how a learned controller optimizes between efficiency and task execution in the future.

TABLE V

ANALYSIS ON CONTAINER USAGE WITH AND WITHOUT THE INCENTIVE FOR PICKING UP THE CONTAINER. HERE, S-SUCCESS, P-PROGRESS AND C-CONTAINER PICKUP %.

Model	%S	%P	SPL	PPL	%C
NoMap(w/o c-reward)	26	47	15	21	55
NoMap(w c-reward)	28	48	16	23	70
OracleMap(w/o c-reward)	77	84	62	64	48
OracleMap(w c-reward)	81	88	67	70	50

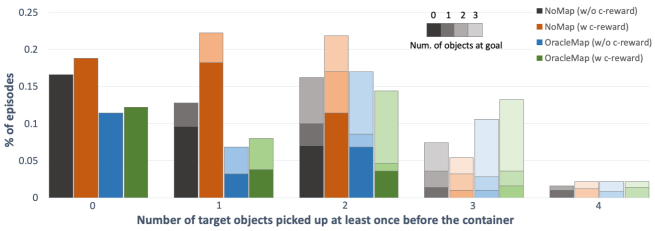

Fig. 4. Breakdown of container usage from Tab. V based on number of objects picked before container. In every column more shaded regions represent larger number of objects already at goal.

c) Analysis on Container Usage:: We analyse container usage and policies in Tab. V and Fig. 4. We see that adding an incentive for picking up a container improves SPL & PPL metrics for both methods, indicating efficiency in task execution. Further, NoMap's container usage is higher than OracleMap methods indicating that NoMap policies prefer picking up a container due to the uncertainties involved in the location of target objects, while OracleMap methods are aware of target object locations and container usage becomes more selective. Having zero objects at the goal while the container was picked up shows that agent executes behaviors like dropping already picked up objects for efficiency which exhibits complexity of behaviors executed by these policies.

d) Results on MultiOn:: Finally, we also evaluate our proposed HTP framework on the MultiOn[2] challenge. Table IV shows that our method is on par with [2]. Note that HTP uses weighted frontier exploration technique that comes with theoretical exploration guarantees hence, its significance is a lot higher in a long-horizon transport like Long-HOT compared to MultiOn. Furthermore, techniques proposed in [42] are complementary to ours.

VI. CONCLUSION

We designed a novel Long-HOT task, focused on deep exploration and long-horizon planning. Further, we proposed a modular hierarchical transport policy (HTP) that builds a topological graph of the scene to perform exploration with the help of weighted frontiers. We build several subtask policies that are connected in novel ways to perform object transport. We show how our approach leads to large improvements in performance on the transport task, its ability to generalize to harder settings and also achieve state-of-the-art numbers on MultiON.

REFERENCES

[1] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A platform for embodied ai research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[2] S. Wani, S. Patel, U. Jain, A. X. Chang, and M. Savva, "Multion: Benchmarking semantic map memory using multi-object navigation," in *NeurIPS*, 2020.

[3] C. Gan, S. Zhou, J. Schwartz, S. Alter, A. Bhandwaldar, D. Gutfreund, D. L. Yamins, J. J. DiCarlo, J. McDermott, A. Torralba, and J. B. Tenenbaum, "The threedworld transport challenge: A visually guided task-and-motion planning benchmark towards physically realistic embodied ai," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 8847–8854, 2022.

[4] J. Krantz, A. Gokaslan, D. Batra, S. Lee, and O. Maksymets, "Waypoint models for instruction-guided navigation in continuous environments," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[5] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese, "Relmogen: Integrating motion generation in reinforcement learning for mobile manipulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4583–4590, 2021.

[6] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[7] O. Kwon, N. Kim, Y. Choi, H. Yoo, J. Park, and S. Oh, "Visual graph memory with unsupervised representation for visual navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15890–15899, October 2021.

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

[9] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," 2021.

[10] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006. Available at http://planning.cs.uiuc.edu/.

[11] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Berlin, Heidelberg: Springer-Verlag, 2007.

[12] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An Interactive 3D Environment for Visual AI," *arXiv*, 2017.

[13] B. Shen, F. Xia, C. Li, R. Martín-Martín, L. Fan, G. Wang, C. Pérez-D'Arpino, S. Buch, S. Srivastava, L. Tchapmi, M. Tchapmi, K. Vainio, J. Wong, L. Fei-Fei, and S. Savarese, "igibson 1.0: A simulation environment for interactive tasks in large realistic scenes," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7520–7527, 2021.

[14] C. Chen, U. Jain, C. Schissler, S. V. A. Gari, Z. Al-Halah, V. K. Ithapu, P. Robinson, and K. Grauman, "Soundspaces: Audio-visual navigation in 3d environments," 2020.

[15] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: real-world perception for embodied agents," in *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*, IEEE, 2018.

[16] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford, L. Weihs, M. Yatskar, and A. Farhadi, "Robothor: An open simulation-to-real embodied ai platform," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba, "Virtualhome: Simulating household activities via programs," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8494–8502, 2018.

[18] X. Puig, T. Shu, S. Li, Z. Wang, Y.-H. Liao, J. B. Tenenbaum, S. Fidler, and A. Torralba, "Watch-and-help: A challenge for social perception and human-{ai} collaboration," in *International Conference on Learning Representations*, 2021.

[19] D. Batra, A. Gokaslan, A. Kembhavi, O. Maksymets, R. Mottaghi, M. Savva, A. Toshev, and E. Wijmans, "ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects," in *arXiv:2006.13171*, 2020.

[20] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, (Red Hook, NY, USA), Curran Associates Inc., 2020.

[21] M. Wortsman, K. Ehsani, M. Rastegari, A. Farhadi, and R. Mottaghi, "Learning to learn how to learn: Self-adaptive visual navigation using meta-learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[22] W. Yang, X. Wang, A. Farhadi, A. Gupta, and R. Mottaghi, "Visual semantic navigation using scene priors," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, 2019.

[23] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, "On evaluation of embodied navigation agents," 2018.

[24] E. Wijmans, A. Kadian, A. S. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra, "Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames," in *ICLR*, 2020.

[25] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman, "Occupancy anticipation for efficient exploration and navigation," in *Computer Vision – ECCV 2020* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 400–418, Springer International Publishing, 2020.

[26] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "Alfred: A benchmark for interpreting grounded instructions for everyday tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[27] L. Weihs, M. Deitke, A. Kembhavi, and R. Mottaghi, "Visual room rearrangement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5922–5931, June 2021.

[28] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," 2018.

[29] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," in *Advances in Neural Information Processing Systems* (M. Jordan, M. Kearns, and S. Solla, eds.), vol. 10, MIT Press, 1998.

[30] A. McGovern and A. G. Barto, "Automatic discovery of subgoals in reinforcement learning using diverse density," in *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, (San Francisco, CA, USA), p. 361–368, Morgan Kaufmann Publishers Inc., 2001.

[31] R. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

[32] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, Feb. 2017.

[33] A. Das, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Neural Modular Control for Embodied Question Answering," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2018.

[34] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[35] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[36] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[37] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017.

[38] S. Gupta, V. Tolani, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," *Int. J. Comput. Vision*, vol. 128, p. 1311–1330, may 2020.

[39] U. Jain, L. Weihs, E. Kolve, A. Farhadi, S. Lazebnik, A. Kembhavi, and A. Schwing, "A cordial sync: Going beyond marginal policies for multi-agent embodied tasks," in *Computer Vision – ECCV 2020* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 471–490, Springer International Publishing, 2020.

[40] J. Oh, V. Chockalingam, S. Singh, and H. Lee, "Control of memory, active perception, and action in minecraft," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, p. 2790–2799, JMLR.org, 2016.

[41] K. Fang, A. Toshev, L. Fei-Fei, and S. Savarese, "Scene memory transformer for embodied agents in long-horizon tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[42] P. Marza, L. Matignon, O. Simonin, and C. Wolf, "Teaching agents how to map: Spatial reasoning for multi-object navigation (winning entry of the multion challenge at cvpr 2021)," *International Conference on Intelligent Robots and Systems (IROS)*, 2022.