REST: Receding Horizon Explorative Steiner Tree for Zero-Shot Object-Goal Navigation

Shuqi Xiao[1], Maani Ghaffari[2], Chengzhong Xu[1], and Hui Kong[1]*
[1]State Key Laboratory of Internet of Things for Smart City, University of Macau
[2]University of Michigan

Abstract—Zero-shot object-goal navigation (ZSON) requires navigating unknown environments to find a target object without task-specific training. Prior hierarchical training-free solutions invest in scene understanding (*belief*) and high-level decision-making (*policy*), yet overlook the design of *option*, *i.e.*, a subgoal candidate proposed from evolving belief and presented to policy for selection. In practice, options are reduced to isolated waypoints scored independently: single destinations hide the value gathered along the journey; an unstructured collection obscures the relationships among candidates. Our insight is that the option space should be a *tree of paths*. Full paths expose en-route information gain that destination-only scoring systematically neglects; a tree of shared segments enables coarse-to-fine LLM reasoning that dismisses or pursues entire branches before examining individual leaves, compressing the combinatorial path space into an efficient hierarchy. We instantiate this insight in REST (Receding Horizon Explorative Steiner Tree), a training-free framework that (1) builds an explicit open-vocabulary 3D map from online RGB-D streams; (2) grows an agent-centric tree of safe and informative paths as the option space via sampling-based planning; and (3) textualizes each branch into a spatial narrative and selects the next-best path through chain-of-thought LLM reasoning. Across the Gibson, HM3D, and HSSD benchmarks, REST consistently ranks among the top methods in success rate while achieving the best or second-best path efficiency, demonstrating a favorable efficiency-success balance.

I. INTRODUCTION

Object-goal Navigation (ObjectNav) requires an embodied agent to navigate to an object of a specified category (*e.g.*, coffee cup) within a previously unseen environment [2]. Whereas a Vision-Language Navigation (VLN) agent is guided by step-by-step linguistic instructions, an ObjectNav agent must rely on itself to reason and plan strategies that balance exploring unknown spaces and exploiting observed cues toward the goal. It underpins real-world challenging tasks such as locating a first-aid kit in an unfamiliar building or fetching a condiment in a rearranged kitchen. To acquire such autonomy from experience, fueled by high-fidelity simulators and massive 3D datasets, deep reinforcement and imitation learning have driven significant progress [3]–[5]. However, these end-to-end ObjectNav policies remain brittle under distribution shifts (*e.g.*, closed vocabulary, sim-to-real gap) and demand large volumes of high-quality interaction data, limiting their generalizability and scalability. In recent years, the zero-shot capabilities of AI foundation models, *e.g.*, Vision-Language Models (VLM) and Large Language Models (LLM), have

catalyzed a paradigm shift toward zero-shot object-goal navigation (ZSON), in which no task-specific training is required. Since Majumdar *et al*. [6] first demonstrated this concept by transferring an image-goal navigation policy to object-goal tasks via CLIP [7], subsequent work is increasingly converging on fully training-free and hierarchical architectures [8]–[11] that marry the two paradigms: data-driven foundation models supply generalizable semantic understanding across novel objects and environments, while model-based navigation frameworks, *e.g.*, SLAM and path planning, provide safety and efficiency guarantees grounded in geometric principles.

These hierarchical ObjectNav agents can be generally characterized by three axes: (1) *belief update*: provides prior commonsense knowledge (*e.g.*, VLM) and aggregates posterior observations (*e.g.*, mapping); (2) *option space*: represents the set of candidate subgoals; (3) *hierarchical policy*: a global policy selects the next-best option and a local policy maps a high-level option to low-level actions. Prior work has invested heavily in enriching the belief, *e.g.*, semantic value map [1], [12] or open-vocabulary 3D scene graphs [10], and strengthening the global policy, *e.g.*, LLM-based ranking [10], [13] or heuristic-based scoring [8], [11]. However, the option space, which is the interface that mediates information flow from belief to policy, remains shaped by convenience rather than design, inheriting whatever representation adjacent modules happen to produce or consume. Prior hierarchical ObjectNav agents adopt diverse option representations: pixel coordinates predicted by VLMs [14], [15], metric map coordinates predicted by RL-based policies [16], [17], exploration frontiers at the boundary of known and unknown space [1], [8], [10], [13], [18], and topological graph nodes that simplify the navigable space [11]. Despite this variety, these representations share a fundamental trait: every option is reduced to a single waypoint and evaluated in isolation. We term this the *next-best-waypoint* paradigm: at each decision epoch, the agent scores candidates by estimating the arrival utility at a waypoint, blind to what lies en route and the structural relationships among candidates.

Consider the search for a target like a *toilet*, as illustrated in Fig. 1. A waypoint-based agent might deprioritize the end of a long hallway because the destination itself lacks semantic cues and incurs high travel cost. However, traversing that corridor offers the chance to "peek" into multiple passing doorways, one of which likely contains the goal. In such cases, point-based scoring systematically undervalues options whose merit lies not at the terminus but along the journey. An effective
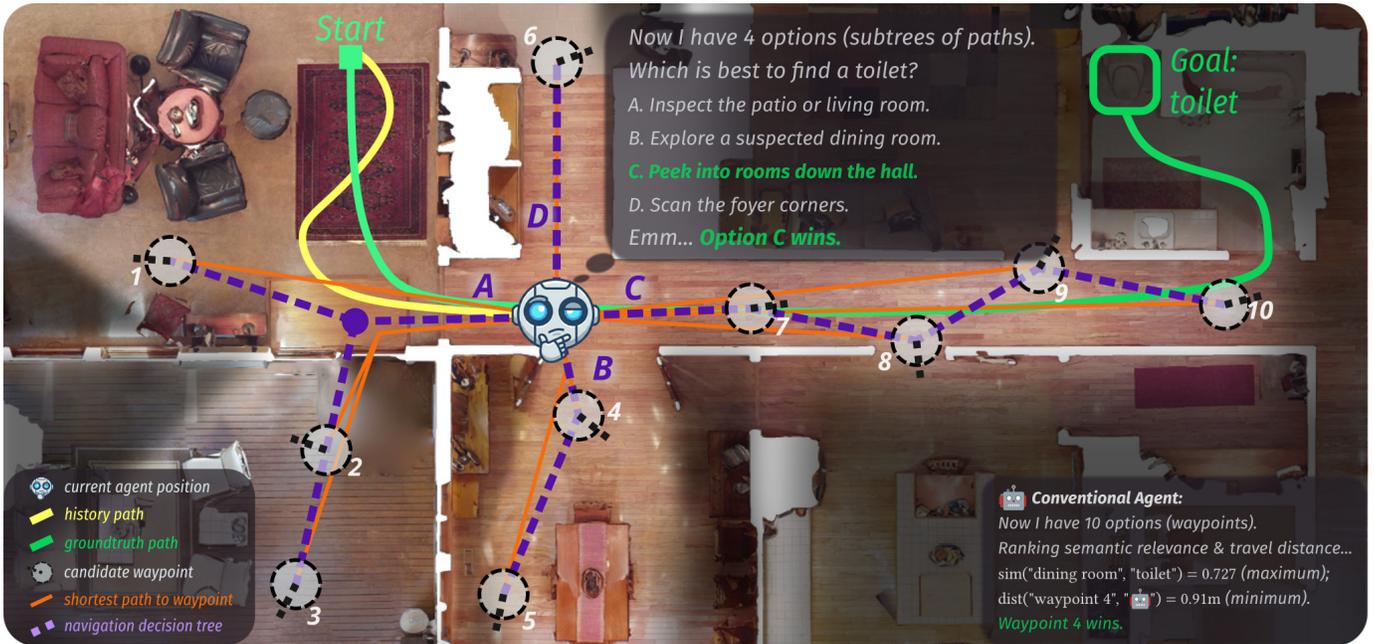
Figure 1. REST reasons over an agent-centric tree of safe and informative paths rather than evaluating isolated waypoints. Here, REST selects the next-best subtree among four options via spatial narratives; a conventional agent (*e.g.*, VLFM [1]) independently scores ten waypoints by semantic similarity and geometric proximity, discarding spatial-temporal context.

option space should therefore be *path-grounded*, making the cumulative gain and cost of the trajectory explicit to the policy. Transitioning from points to paths, however, introduces a combinatorial explosion: n reachable waypoints can yield an exponential number of possible routes, far exceeding the reasoning capacity of LLM-based policies. Furthermore, listing all candidates independently obscures their shared intent: in Fig. 1, the paths to waypoints 7–10 all traverse the corridor, yet appear as four separate options, forcing the policy to evaluate route-level variations when the underlying decision, which region to explore next, is the same. We propose that both challenges are addressed by a *navigation decision tree* that hierarchically clusters candidate paths by identifying their shared segments. Such a structure enables coarse-to-fine reasoning under partial observation, mirroring human navigation where one first selects a promising direction before refining the specific destination based on new observations. As shown in Fig. 1, dozens of potential paths are compressed into four compact subtrees, allowing the LLM to select the optimal "branch" through efficient linguistic reasoning. We therefore argue that the option space for zero-shot ObjectNav should be a *tree of paths*: path-grounded to expose en-route utility, and tree-structured to enable tractable, hierarchical reasoning.

We instantiate this insight in REST (Receding Horizon Explorative Steiner Tree), a training-free ObjectNav framework. REST completely replaces conventional isolated waypoint sets with a compact *navigation decision tree*, elegantly organizing path-grounded options into an efficient hierarchy. We evaluate REST on the Gibson, HM3D, and HSSD ObjectNav benchmarks. Across all three benchmarks, REST consistently ranks

among the top methods in success rate while achieving the best or second-best path efficiency, demonstrating a favorable efficiency-success balance that we attribute to tree-structured, path-grounded planning. We make three contributions:

- We introduce the *next-best-path* paradigm for zero-shot ObjectNav, which replaces isolated waypoint scoring with a path-grounded option space that exposes en-route information gain.
- We propose the *navigation decision tree*, a hierarchical option representation constructed via Euclidean Steiner Tree optimization and an open-vocabulary textualization pipeline.
- We present REST, a training-free ZSON framework, and demonstrate consistently competitive performance across three benchmarks with a favorable efficiency-success trade-off.

II. RELATED WORK

A. Foundation models for ObjectNav

Foundation models provide transferable priors that reduce or remove the need for task-specific training in embodied agents. A prominent line of work [1], [12], [18] uses CLIP-style VLMs [7] for both open-vocabulary perception and high-level decision-making. These methods project goal-conditioned cosine similarities onto a top-down grid map, producing an implicit "value map" that guides frontier selection. However, cosine similarities among CLIP image or text embeddings can only measure semantic relevance rather than performing complex linguistic reasoning. In the example in Fig. 1, $\langle \text{toilet}, \text{dining room} \rangle \approx 0.73 > 0.67 \approx \langle \text{toilet}, \text{corridor} \rangle$ leads

the agent to search the dining room, unaware of the benefits of traversing the corridor to locate a bathroom. Moreover, since every cell of the grid map encodes relevance to a single target category, the entire map must be discarded and recomputed whenever the goal switches, preventing reuse of previously gathered spatial knowledge. These two limitations motivate the separation in our framework between a goal-agnostic open-vocabulary semantic map, which can persist across goal changes, and a language-model reasoning layer that exploits commonsense priors over the map.

B. Autonomous exploration for ObjectNav

In ObjectNav, exploration is inseparable from goal seeking: the agent must decide where to look next so that each observation gathers semantic cues that guide it toward the target. Map-based ObjectNav systems therefore borrow epistemic heuristics from autonomous exploration to reduce uncertainty about the unknown environment efficiently. Frontier-based exploration [19] has been widely used [1], [8], [10], [13], [18], and VoroNav [11] builds a Generalized Voronoi Graph following [20] to provide a graph of collision-free paths for geometric exploration. Frontier-based methods follow an "explore-then-plan" paradigm [21]: they first select a next-best frontier, then invoke a path planner such as A* [22] to reach it. Because each option in our framework is a complete, executable path, exploration and path planning cannot be decoupled in this way. The Voronoi graph distills free space into a skeleton that maximizes clearance from obstacles, yielding a rigid topology dictated entirely by known geometry. Because the target location is unknown, the paths an agent needs are shaped by what it has yet to observe, not by where walls lie; a fixed geometric skeleton cannot adapt to an evolving belief state. For instance, an agent confined to maximal-clearance paths cannot approach a table closely enough to inspect whether a small target rests on it. Our method draws instead on sampling-based informative path planning methods [21], [23], which incrementally grows a tree of both feasible and informative paths in a more flexible manner. By shaping sampling strategies, our method solves the combined exploration-planning problem efficiently.

III. METHODOLOGY

An overview of REST is shown in Fig. 2. The system is organized around three components: *belief*, *option*, and *policy*. At each decision epoch, the agent first updates its belief through geometric, semantic, and road mapping (Sec. III-A). The option space is then constructed by the Receding Horizon Explorative Steiner Tree (Sec. III-B), which maintains a navigation decision tree of safe, informative, and hierarchical paths; each branch is textualized by projecting the semantic map along a virtual panoramic camera sweep. Finally, the policy (Sec. III-C) selects the next-best-path via LLM reasoning over the textualized options and executes it with an information-gain-driven orientation policy that determines camera headings along the selected path. The system replans in a receding-horizon manner.

A. Belief Update

1) Geometric mapping: We leverage UFOMap [24], an efficient 3D volumetric mapping system, to integrate a posed RGB-D stream into an octree of occupancy voxels online. Each octree node is categorized as unknown, free, or occupied based on its occupancy value. The geometric map mainly serves as an anchor for semantic mapping and road mapping, and also provides efficient geometry intersection computation for both collision-checking in safe path planning and ray-casting for informative viewpoint sampling.

2) Semantic mapping:

a) 2D perception: We capture open-vocabulary, explicit and fine-grained 2D semantics by a cascaded *recognize-detect-segment* workflow based on off-the-shelf edge-friendly foundation models. For each RGB image, first, we prompt a Small Vision-Language Model (SVLM) to tag the image by an array of salient, discrete, manipulable entities with clear affordances (*e.g.*, washing machine, door, plant) it contains. Then, we feed the entity labels, *e.g.*, wooden cabinet, into an Open-Vocabulary Object Detector (OVOD) to obtain 2D bounding boxes with confidence scores. Finally, we use an edge-optimized universal segmentation model based on SAM2 [25] to refine detection boxes to segmentation masks. This pipeline synergizes these foundation models to overcome their individual limitations on balancing semantic understanding capabilities and visual grounding granularity. While VLMs offer strong zero-shot understanding, they lack localization precision; conversely, OVOD and SAM excel at dense grounding and segmentation but require explicit text or spatial prompts. The final results are consumed by the 3D semantic mapping described next.

b) 3D mapping: First, following [26], we annotate the 3D volumetric map by fusing 2D perception results across multiple viewpoints. For each frame, we retrieve all visible occupied octree voxels inside the camera's view frustum and project them onto the image plane. Each projected pixel that falls inside a detected instance mask casts a vote, weighted by detection confidence, for the corresponding instance label. Each voxel maintains a per-label vote histogram; the label with the highest accumulated score serves as the current semantic estimate. By aggregating confidence-weighted votes across multiple viewpoints, this majority-voting scheme smooths out single-frame noise and yields a robust 3D semantic map. Second, we periodically apply per-label DBSCAN [27] to suppress floating artifacts and fit a 3D bounding box to each surviving cluster, producing a set of discrete entities from the annotated voxel map. To track entities over time, we match clusters between consecutive snapshots by computing 3D IoU of their bounding boxes; a match above a threshold carries forward the existing entity identity, while unmatched clusters are initialized as new entities. However, open-vocabulary tagging introduces cross-frame label inconsistency: the same real-world cup may be tagged as "blue cup", "ceramic cup", or "teacup" across different frames, producing redundant overlapping entities. We resolve this with a spatial-semantic joint clustering: we measure spatial overlap by intersection over
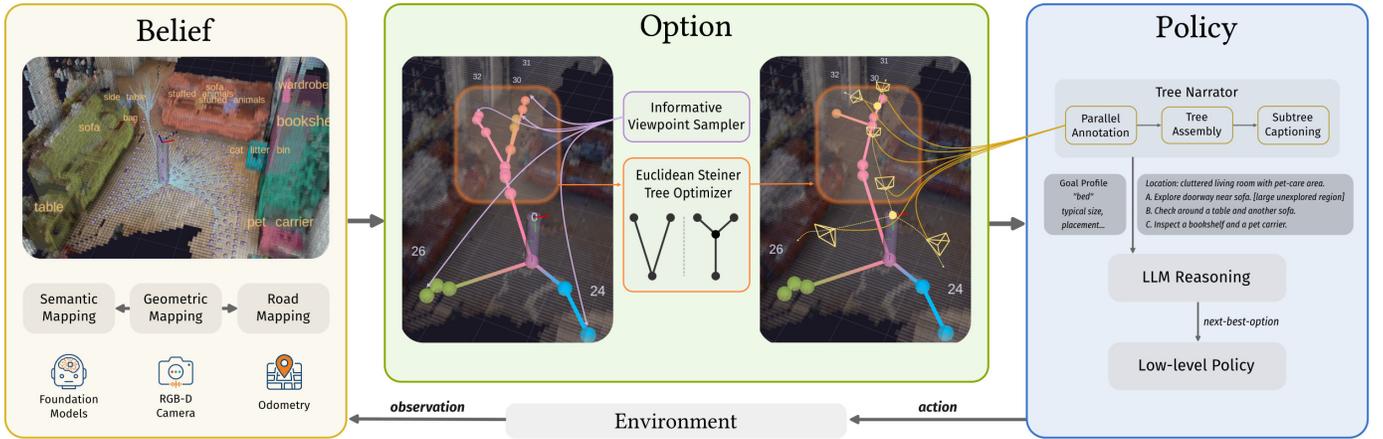
Figure 2. Overview of REST, a training-free ObjectNav framework that replans in a receding-horizon manner. At each decision cycle, the agent updates the from online RGB-D streams, grows an agent-centric Steiner tree of safe and informative paths as the option space, textualizes each branch into a spatial narrative, and selects the next-best path through chain-of-thought LLM reasoning.

smaller (IoS) and semantic relevance by the cosine similarity between label embeddings (MobileCLIP2 [28]). When both exceed their thresholds (τ_{IoS}, τ_{sim}), the smaller entity is merged into the larger one, consolidating fragmented labels into a single block that represents one real-world entity.

3) Road Mapping: Our path-grounded option space (section III-B) requires a library of all feasible paths. Inspired by sampling-based path planning methods, we adopt Real-Time RRT* (RT-RRT*) [29] to continuously maintain a global tree of feasible paths rooted at the agent's pose.

We expand the RT-RRT* tree by sampling candidate waypoints with the hybrid local-global strategy of Schmid *et al*. [23]: when node density in the agent's vicinity falls below a threshold, samples are drawn locally to maintain connectivity; otherwise, samples are drawn globally to extend coverage. Each candidate must pass a traversability check. We attach a thin axis-aligned bounding box (AABB) beneath the agent's base and query the occupied voxels within this slab on the 3D occupancy grid; the waypoint is accepted only if the occupied-volume ratio exceeds a support threshold. This check rejects waypoints over open air, which in multi-storey scenes prevents a wheeled robot from falling downstairs. For each candidate edge, we approximate the robot body with a cylindrical bounding volume, compute the 3D swept volume of the straight-line motion via an oriented bounding box (OBB) estimated by PCA, and declare the edge feasible only if this volume has no intersection with occupied or unknown voxels. Together, these predicates ensure that every node is traversable and reachable by collision-free edges.

The tree is expanded following the RRT* rewiring procedure [29]: the new node is connected to its lowest-cost neighbor and nearby edges are rewired if a shorter path through the new node exists. This rewiring rule is the standard mechanism that gives RRT* its asymptotic path-length optimality. As the agent moves, the tree is rerooted to the agent's current pose, keeping path queries agent-centric without rebuilding

the tree from scratch. The tree persists across decision epochs and grows incrementally with each map update, eventually spanning all traversable and reachable space observed so far. The resulting roadmap records where the agent can stand and how it can get there at lowest cost. The option space (section III-B) queries this roadmap to extract informative subgoals and to compute the navigation decision tree.

B. Option Space

1) Informative viewpoint sampler: Our RT-RRT* roadmap provides probabilistically complete coverage of reachable states via an agent-centric tree of asymptotically shortest paths. However, the roadmap typically contains thousands of paths, which exceeds the reasoning capacity of an LLM-based decision-maker and introduces extreme redundancy. Many of these paths are semantically equivalent for high-level intent, differing only in low-level geometric details such as local motion smoothness. We therefore extract *informative viewpoints* by applying a two-pass online filter to every newly added RT-RRT* node.

a) Spatial thinning: Because information gain computation requires computational heavy ray-casting, we therefore apply online Poisson-disc sampling first: a candidate is accepted only if no existing viewpoint lies within radius r. This produces a well-distributed, low-redundancy candidate set at negligible cost.

b) Information-gain gating: For each spatially accepted candidate $\theta \in \mathrm{SE}(3)$, we define its information gain as the number of unknown, visible voxels within its view frustum:

$$r(\theta \mid \mathcal{M}) = |\{v \in \mathcal{M} \cap \mathcal{F}(\theta) : \mathrm{I}_{\text{visible}}(v, \theta) \cdot \mathrm{I}_{\text{unknown}}(v) = 1\}| \quad (1)$$

where $\mathcal{M}$ is the UFOMap, $\mathcal{F}(\theta)$ is the view frustum of camera pose θ, $\mathrm{I}_{\text{visible}}(v, \theta) = 1$ iff the ray from θ to the center of voxel v is unoccluded, and $\mathrm{I}_{\text{unknown}}(v) = 1$ iff v is classified as unknown. Only candidates whose information gain exceeds a threshold τ_{IG} are retained as informative

viewpoints. Our informative viewpoints serve a similar role to frontiers in classical frontier-based exploration [19]: both identify candidate poses from which the robot can observe currently unknown space. Our sampling-based informative viewpoints integrate naturally with our RT-RRT* roadmap, as viewpoints are generated as a byproduct of tree expansion rather than requiring a separate grid search like classical exploration frontiers.

Given N retained viewpoints, we can query the RT-RRT* for the current shortest path to each by tracing parent nodes in the tree structure. These N paths share significant segments thanks to the global RRT*, paving the way for more aggressive tree abstraction described next.

2) Euclidean Steiner Tree Optimizer: Though RT-RRT* provides a tree of collision-free paths to informative viewpoints, because each path is optimized independently, unnecessary redundant edges exist for navigation decision-making (Fig. 3, left). Our key insight is that sacrificing per-path optimality for a globally shorter tree also reveals a hierarchical decision structure: paths toward nearby terminals save total cost by sharing a common segment and splitting only where the detour penalty exceeds the sharing benefit, producing a trunk-to-branch-to-leaf topology, visualized in "Euclidean Steiner Tree Optimizer" in Fig. 2.

Formally, let the agent pose $r \in \mathcal{F}$ be the root and the informative viewpoints $V_T = \{v_1, \ldots, v_N\} \subset \mathcal{F}$ be terminals, where $\mathcal{F} \subseteq \mathbb{R}^2$ is obstacle-free space. We seek a tree $T = (V, E)$ embedded in $\mathcal{F}$ with $\{r\} \cup V_T \subseteq V$ that minimizes total edge cost $C(T) = \sum_{e \in E} \|e\|_2$. This is an instance of the *obstacle-avoiding Euclidean Steiner Minimum Tree* (OAESMT [30]) problem, which is NP-hard even without obstacles: the agent pose and informative viewpoints are the *terminals*, the nodes any feasible tree must connect, while the solver may introduce auxiliary *Steiner points* to reduce total edge cost by merging nearby path segments.

We approximate the solution in algorithm 1. A critical observation is that the RT-RRT* subtree connecting root to all terminals is already a feasible Steiner tree, just not at minimum cost. We can therefore perform iterative local improvement on this warm start. In each iteration, we generate candidate Steiner points by computing the geometric median (Weiszfeld's algorithm [31]) of each branching node and its children via post-order traversal, add collision-free candidates to the node set, compute a Minimum Spanning Tree over all nodes using Kruskal's algorithm [32] with edges restricted to collision-free line-of-sight segments, and prune non-terminal leaves. The output tree is fed back as input until total cost no longer decreases, converging to a local minimum. Each iteration yields a valid tree, so the agent can act on the current best result while refinement continues. In practice, we refines the tree in the background and serves the best-so-far result on demand; when the underlying planning problem changes (*e.g.*, new viewpoints appear), the solver restarts from the updated warm start.

Viewpoints may become intermediate nodes when doing so reduces total cost; the only invariant is that every view-point remains connected. The resulting navigation decision tree surfaces decision points at topological junctions such as corridor forks and room entrances (Fig. 3, right). The LLM planner selects the next-best subtree at topological junctions and commits to it in a receding-horizon fashion (Sec. III-C).

Figure 3. The RT-RRT* subtree connecting current agent (indexed by 0) to all informative viewpoints indexed from 1 to 15 (left) versus the optimized Steiner tree (right). Independent per-path optimization produces redundant edges, while the OAESMT formulation merges shared segments and surfaces decision junctions, reducing total path length from 85m to 47m.

Algorithm 1: Navigation Decision Tree Generation

Input: RT-RRT* tree $\mathcal{T}_{\mathrm{rrt}}$, agent pose r, informative viewpoints V_T, collision checker $\mathcal{CC}$

Output: Navigation decision tree T^*

1   $T \leftarrow \textsc{ExtractSubtree}(\mathcal{T}_{\mathrm{rrt}}, r, V_T);$ $C^* \leftarrow C(T);$

2   **repeat**

3     $V \leftarrow \mathrm{Nodes}(T);$
       $V_T \leftarrow \{v \in V : v \text{ is root or goal}\};$

4   **foreach** *branching node* $v \in T$ (post-order) **do**

5       $\mathbf{s} \leftarrow \textsc{WeiszfeldMedian}(\{v\} \cup \mathrm{Children}(v));$

6   **if** $\exists \mathbf{s}^* \in \mathcal{T}_{\mathrm{rrt}}$ *s.t.* $\|\mathbf{s} - \mathbf{s}^*\| < \delta$ **then** $\mathbf{s} \leftarrow \mathbf{s}^*;$

7   **if** $\textsc{CollisionFree}(\mathbf{s})$ **then** $V \leftarrow V \cup \{\mathbf{s}\};$

8     $E \leftarrow \{(u, v) : u, v \in V, \textsc{CollisionFree}(u, v)\};$

9     $T' \leftarrow \textsc{PruneLeaves}(\textsc{Kruskal}(V, E), V_T);$

10   **if** $C(T') < C^*$ **then** $T^* \leftarrow T';;$

11    $C^* \leftarrow C(T');;$

12    $T \leftarrow T';$

13 **until** $C(T') \geq C^*;$

14 **return** T^*

C. Hierarchical Policy

1) Tree Narration: Given the geometric map, the semantic entity map (Sec. III-A), and a constructed agent-centric navigation decision tree, our goal is to convert this tree structure into a linguistic representation that an LLM can reason about and plan. We achieve this through a three-stage pipeline: (1) per-edge annotation, (2) tree-level assembly, and (3) subtree captioning.

a) Per-Edge Annotation: We decompose the navigation decision tree into individual edges and annotate each edge in parallel. Along each edge, we simulate a forward-facing virtual camera traversing from the parent node to the child node,

querying the geometric and semantic maps for visible content. At designated informative viewpoints (identified during tree construction), the virtual camera switches to a panoramic field of view, allowing a full scan of the surroundings at key decision points. Each edge produces a structured description with two categories:

- **Known:** For each visible semantic entity, we log its unique identifier, label, caption, and the sighting distance along the edge from the parent node (*e.g.*, "pet carrier @ 0.5 m, sofa @ 2.0 m").
- **Unknown:** We cluster visible unknown voxels via DB-SCAN, computing each cluster's volume and its centroid distance to the nearest known entity (*e.g.*, "3.2 m^3 unknown region near the sofa").

b) Tree-Level Assembly: We assemble the per-edge annotations in a top-down traversal from root to leaves. Since each entity carries a unique identifier, we track sighting distances across parent-child edges to detect spatial trends. For instance, if an entity's sighting distance decreases along a root-to-leaf path, the assembly marks this subtree as *approaching* that entity.

c) Subtree Captioning: A summarizer LLM converts the assembled structured annotations into concise natural language options, one per root-level subtree. The summarizer is encouraged to produce condensed, semantically rich descriptions: individual object lists are abstracted into region-level or room-level characterizations. For example, a subtree containing {chairs, dining table} may be captioned as "a dining area," while a subtree observing {coffee table, sofa, pet carrier, cat litter bin, stuffed animals} becomes "a cluttered cozy living room with a pet-care area." This summarizer is separate from the downstream decision-making LLM: the summarizer distills geometric and semantic detail into compact option descriptions, while the decision-maker reasons over these options to select an action.

2) Receding-Horizon LLM Planning: The option-space pipeline (Sec. III-B) runs in a model-predictive-control (MPC) style loop: the RT-RRT* roadmap re-roots to the agent's current pose, and the informative-viewpoint sampler together with the Steiner tree optimizer refresh the navigation decision tree at sensor rate. The LLM deliberation, by contrast, requires on the order of seconds per invocation. We therefore decouple the system into two layers: a *fast reactive layer* that maintains the option space continuously, and an *event-triggered deliberative layer* that invokes the LLM only at decision-relevant moments. Because the reactive layer updates the tree regardless of the agent's current commitment, new obstacles and map changes are immediately reflected in the option space; only when such changes are decision-relevant does the deliberative layer re-engage.

a) Commitment Model: Upon selection, the agent follows the chosen path through non-branching nodes without requerying the LLM. The execution granularity of the receding-horizon loop is therefore not a fixed metric segment but an adaptive path between consecutive decision events, shaped by the topological structure of the environment. This avoids redundant LLM invocations along corridors and other topologically simple segments where no meaningful alternative exists.

b) Re-Invocation Triggers: The LLM is re-invoked when the agent's current position is a branching node in the live decision tree, i.e., the root of the continuously updated tree has multiple child subtrees. Because the tree updates in real time, this condition captures both physical arrival at a pre-existing junction and the dynamic emergence of a new branch at the agent's current position. Re-invocation also occurs when a decision-relevant structural change is detected within the committed subtree: a committed node is removed (its viewpoint is no longer informative) or the subtree's parent-child topology is rewired by the Steiner tree optimizer. Changes outside the committed subtree are deferred until the agent reaches a branching node, avoiding unnecessary re-invocations from distant map updates.

c) Chain-of-Thought Scoring: The LLM receives the target object category and the numbered subtree captions, along with an explicit "none of the above" option that permits abstention. We enforce a Chain-of-Thought (CoT) reasoning process: the LLM first analyzes each caption by reasoning about the likelihood of finding the target based on semantic co-occurrence and spatial layout priors, then assigns a numeric score (0–100) to each option. For instance, when searching for a bed, the model might reason: *"Option A describes a living room, which rarely contains beds. Option B leads to a hallway near a large unexplored region, which could plausibly lead to a bedroom."* The highest-scored option determines the next subgoal.

d) Geometric Fallback: When the LLM selects "none of the above," the semantic signal is insufficient for informed decision-making. This occurs in two common scenarios: early in exploration when the map contains few recognized entities, and in monolithic scenes where visually similar paths (*e.g.*, multiple featureless hallways at a junction) offer no discriminative semantic cues. In such cases, the agent falls back to a purely geometric strategy, navigating to the nearest informative viewpoint. Upon arrival, the full pipeline immediately reruns with the enriched belief state.

IV. EXPERIMENTS

A. Experimental Setup

a) Datasets and configurations: We evaluate our methodology in the Habitat simulator using three photorealistic indoor datasets: Gibson [16], [33], HM3D [34], and HSSD [35]. We follow the conventional setup of the Habitat Navigation Challenge 2023 [36] benchmark, where the embodied agent is modeled as a grounded robot, equipped with a head-mounted, forward-facing RGB-D camera that provides 640×480 resolution observations at a 79° horizontal field-of-view (HFOV) with a depth sensing range of $[0.5\mathrm{m}, 5.0\mathrm{m}]$. The agent's action space consists of six discrete commands: `MOVE_FORWARD` (0.25m), `TURN_LEFT` (30°), `TURN_RIGHT` (30°), `LOOK_UP` (30°), `LOOK_DOWN` (30°), and `STOP`.

b) Evaluation metrics: We report Success Rate (SR) and Success weighted by Path Length (SPL) to evaluate ObjectNav performance. SR measures the proportion of episodes successfully completed within a strict budget of 500 action steps. SPL further assesses navigation efficiency by weighting successful episodes against the optimal path length.

c) Implementation details: For off-the-shelf foundation models, we use a 2B-parameter, 8-bit quantized variant of the instruction-tuned Qwen3-VL [37] as the SVLM for both semantic perception (section III-A2) and LLM-based summarization and reasoning (section III-C). We use YOLO-World [38] for open-vocabulary object detection and EdgeTAM [39] for instance segmentation with box prompts. All experiments are executed on a desktop computer with an NVIDIA GeForce RTX 4080 GPU, an Intel Core i7-14700K CPU, and 32 GB RAM. We set the UFOMap voxel size to 5 cm, the information-gain threshold to $\tau_{IG} = 10$, and the Poisson-disc radius for informative viewpoint spacing to $r = 1.25$ m. We run DBSCAN at 1 Hz and use an IoU threshold of $\tau_{IoU} = 0.5$ to track entities across snapshots. For cross-label entity merging, we use an IoS threshold of $\tau_{IoS} = 0.3$ and a cosine similarity threshold of $\tau_{sim} = 0.8$.

B. Baseline Methods

We compare REST against recent zero-shot ObjectNav methods.

- VLFM [1] scores frontiers via VLM cosine similarity with the goal description
- ApexNAV [18] adaptively switches between exploration and exploitation frontiers with target-centric semantic fusion to denoise 2D semantics.
- GAMap [12] augments a 2D semantic map with goal-oriented affordance and geometry embeddings.
- SG-Nav [10] constructs an online 3D scene graph for frontier selection to unleash LLM reasoning capabilities.
- VoroNav [11] simplifies free space into a Voronoi graph for topological planning
- TriHelper [40] decomposes navigation into three specialized sub-policies for exploration, approach, and identification.
- ImagineNav [41] prompts a VLM to imagine unobserved regions in a mapless framework.
- UniGoal [42] extends SG-Nav to unify goal representations across navigation tasks.
- PanoNav [15] performs mapless navigation from panoramic VLM reasoning.

C. Benchmark Comparison

Table I reports SR and SPL across three datasets. On Gibson, REST achieves competitive SR and SPL matching the top-performing method. The compact layouts of Gibson scenes leave limited room for differentiation, while strong methods already approach saturation on this dataset. This result establishes REST as competitive with top-performing methods on the standard benchmark. On HM3D, the larger and noisier real-world scans present a more discriminative

Table I
COMPARISON OF ZERO-SHOT METHODS ON GIBSON, HM3Dv1, AND HSSD OBJECTNAV VALIDATION SPLITS. FIRST , SECOND , AND THIRD RESULTS ARE HIGHLIGHTED.

Method	Gibson		HM3Dv1		HSSD	
	SR	SPL	SR	SPL	SR	SPL
SG-Nav	–	–	54.0	24.9	–	–
GAMap	85.7	55.5	53.1	26.0	–	–
VoroNav	–	–	42.0	26.0	41.0	23.2
VLFM	84.0	52.2	52.5	30.4	–	–
TriHelper	85.2	43.1	56.5	25.3	–	–
ApexNAV	–	–	59.6	33.0	–	–
ImagineNav	–	–	53.0	23.8	51.0	24.9
UniGoal	–	–	54.5	25.1	–	–
PanoNav	–	–	43.5	23.7	–	–
REST (Ours)	85.1	53.5	57.3	33.4	56.7	29.1

benchmark. ApexNAV achieves the highest SR, while REST attains competitive SR with the highest SPL across all methods. We attribute the SR gap primarily to scanning artifacts in HM3D; for example, reflective surfaces (mirrors, glass) are captured as geometric holes that introduce phantom openings and inflate the 3D information gain calculation, repeatedly drawing the agent to poses where no real progress can be made and exhausting the 500-step budget. By contrast, 2D frontier methods such as ApexNAV are largely immune. This sensitivity to scan noise reduces SR, yet episodes that REST does complete benefit from our tree-structured and path-grounded planning, yielding the highest SPL. On HSSD, among the methods that report results, REST leads in both SR and SPL. HSSD scenes feature diverse layouts (garages, outdoor lounges, extended corridors) with sparser semantic cues than Gibson or HM3D, which degrades point-wise waypoint ranking: the highest-similarity score can attach to an irrelevant nearby object, trapping the agent in a greedy local minimum. REST's option space structures waypoints into a tree of paths, letting the LLM evaluate directions rather than individual destinations, a coarser granularity that remains discriminative even when semantics are sparse in the environment. Compared with VoroNav [11], the closest graph-based baseline, REST improves SR by over 15 points on both HM3Dv1 and HSSD while also raising SPL by 7.4 and 5.9 points, respectively. Since both methods reason over a navigational graph, this gap isolates the value of our design choices: an epistemic, tree-structured graph that exposes en-route information to the policy outperforms a rigid geometric skeleton derived from known free space.

D. Ablation Study

We ablate REST's two core components on HSSD to isolate their individual contributions. Without LLM reasoning, the agent defaults to the nearest informative viewpoint and

Table II
ABLATION STUDY ON HSSD OBJECTNAV VALIDATION SPLIT.

Variant	SR	SPL
REST (Full)	56.7	29.1
w/o LLM reasoning	29.1	18.7
w/o Steiner tree	53.1	25.3

exhausts the 500-step budget more frequently, yielding lower SR and substantially lower SPL. More notably, removing the Steiner tree optimizer while retaining LLM reasoning reveals a compounding degradation across the pipeline. Without compaction, the raw RT-RRT* subtree branches early and shares shorter trunks, so edges that would have been merged into a single segment remain separate. This inflation multiplies the tree narrator's workload: more edges must be ray-cast in parallel, increasing wall-clock latency per decision cycle. Worse, the redundant edges produce repetitive tokens in the subtree captioning stage, degrading the performance of our deployed SVLM summarizer. The resulting option descriptions presented to the decision-making LLM are longer and more redundant, obscuring the topological decision points (corridor forks, room entrances) that the Steiner tree explicitly surfaces. Both SR and SPL decrease relative to the full model, confirming that compact decision trees are a more effective substrate for LLM spatial reasoning than individually optimized paths.

V. CONCLUSION

We argued that the option space of a zero-shot ObjectNav agent should be a tree of paths rather than independently scored waypoints. REST instantiates this idea with a Receding Horizon Explorative Steiner Tree whose branches are textualized via an open-vocabulary 3D semantic map and selected through hierarchical LLM reasoning. Experiments on Gibson, HM3D, and HSSD show that REST consistently ranks among the top methods in success rate while achieving the best or second-best path efficiency across all three benchmarks, confirming a favorable efficiency-success balance. These results suggest that rethinking the option space, the interface between perception and decision-making, is a complementary and underexplored axis for improving future hierarchical embodied AI agents.

REFERENCES

[1] N. Yokoyama *et al.*, "VLFM: Vision-Language Frontier Maps for Zero-Shot Semantic Navigation," in *ICRA*. IEEE, May 2024, pp. 42–48.

[2] D. Batra *et al.*, "ObjectNav Revisited: On Evaluation of Embodied Agents Navigating to Objects," Aug. 2020. [Online]. Available: http://arxiv.org/abs/2006.13171

[3] H. Du *et al.*, "VTNet: Visual transformer network for object goal navigation," in *ICLR*, 2021.

[4] R. Ramrakhya *et al.*, "Pirlnav: Pretraining with imitation and rl finetuning for objectnav," in *CVPR*, 2023, pp. 17 896–17 906.

[5] S. Chen *et al.*, "Object Goal Navigation with Recursive Implicit Maps," in *IROS*. IEEE, Oct. 2023, pp. 7089–7096.

[6] A. Majumdar *et al.*, "ZSON: Zero-shot object-goal navigation using multimodal goal embeddings," in *NeurIPS*), S. Koyejo *et al.*, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 32 340–32 352.

[7] A. Radford *et al.*, "Learning Transferable Visual Models From Natural Language Supervision," in *ICML*. PMLR, Jul. 2021, pp. 8748–8763.

[8] K. Zhou *et al.*, "ESC: Exploration with soft commonsense constraints for zero-shot object navigation," in *ICML*, ser. ICML '23. JMLR.org, 2023.

[9] S. Y. Gadre *et al.*, "CoWs on Pasture: Baselines and Benchmarks for Language-Driven Zero-Shot Object Navigation," in *CVPR*. IEEE, Jun. 2023, pp. 23 171–23 181.

[10] H. Yin *et al.*, "SG-Nav: Online 3D scene graph prompting for LLM-based zero-shot object navigation," in *NeurIPS*), ser. NeurIPS '24. Curran Associates Inc., 2024.

[11] P. Wu *et al.*, "VoroNav: Voronoi-based zero-shot object navigation with large language model," in *ICML*, 2024.

[12] Y. Fang *et al.*, "GAMap: Zero-Shot Object Goal Navigation with Multi-Scale Geometric-Affordance Guidance," in *NeurIPS*), vol. 37. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024, pp. 39 386–39 408.

[13] B. Yu *et al.*, "L3MVN: Leveraging Large Language Models for Visual Target Navigation," in *IROS*. IEEE, Oct. 2023, pp. 3554–3560.

[14] W. Cai *et al.*, "Bridging Zero-shot Object Navigation and Foundation Models through Pixel-Guided Navigation Skill," in *ICRA*. IEEE, May 2024, pp. 5228–5234.

[15] Q. Jin *et al.*, "PanoNav: Mapless Zero-Shot Object Navigation with Panoramic Scene Parsing and Dynamic Memory," Nov. 2025. [Online]. Available: http://arxiv.org/abs/2511.06840

[16] D. S. Chaplot *et al.*, "Object goal navigation using goal-oriented semantic exploration," *NeurIPS*), vol. 33, pp. 4247–4258, 2020.

[17] G. Georgakis *et al.*, "Learning to map for active semantic goal navigation," in *ICLR*, 2022.

[18] M. Zhang *et al.*, "ApexNAV: An adaptive exploration strategy for zero-shot object navigation with target-centric semantic fusion," *IEEE Robot. Autom. Lett.*, vol. 10, no. 11, pp. 11 530–11 537, 2025.

[19] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. IEEE Int. Symp. Comput. Intell. Robot. Autom.* IEEE Comput. Soc. Press, 1997, pp. 146–151.

[20] Ji Yeong Lee *et al.*, "Sensor-based exploration for convex bodies: A new roadmap for a convex-shaped robot," *IEEE Trans. Robot.*, vol. 21, no. 2, pp. 240–247, Apr. 2005.

[21] B. Lindqvist *et al.*, "A Tree-Based Next-Best-Trajectory Method for 3-D UAV Exploration," *IEEE Trans. Robot.*, vol. 40, pp. 3496–3513, 2024.

[22] P. Hart *et al.*, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, 1968.

[23] L. Schmid *et al.*, "An Efficient Sampling-Based Method for Online Informative Path Planning in Unknown Environments," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1500–1507, Apr. 2020.

[24] D. Duberg *et al.*, "UFOMap: An efficient probabilistic 3D mapping framework that embraces the unknown," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6411–6418, 2020.

[25] N. Ravi *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024. [Online]. Available: https://arxiv.org/abs/2408.00714

[26] J. McCormac *et al.*, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *ICRA*, May 2017, pp. 4628–4635.

[27] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, vol. 96, no. 34, 1996, pp. 226–231.

[28] F. Faghri *et al.*, "MobileCLIP2: Improving multi-modal reinforced training," *TMLR*, 2025.

[29] K. Naderi *et al.*, "RT-RRT*: A real-time path planning algorithm based on RRT*," in *Proc. ACM SIGGRAPH Conf. Motion in Games*. ACM, Nov. 2015, pp. 113–118.

[30] M. Zachariasen *et al.*, *Obstacle-Avoiding Euclidean Steiner Trees in the Plane: An Exact Algorithm*, 1999, pp. 286–299.

[31] E. Weiszfeld, "Sur le point pour lequel la somme des distances de n points donnés est minimum," *Tohoku Math. J., First Ser.*, vol. 43, pp. 355–386, 1937.

[32] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, no. 1, pp. 48–50, 1956.

[33] F. Xia *et al.*, "Gibson Env: Real-World Perception for Embodied Agents," in *CVPR*. IEEE, Jun. 2018, pp. 9068–9079.

[34] S. K. Ramakrishnan *et al.*, "Habitat-matterport 3D dataset (HM3D): 1000 large-scale 3D environments for embodied AI," 2021. [Online]. Available: https://arxiv.org/abs/2109.08238

[35] M. Khanna *et al.*, "Habitat Synthetic Scenes Dataset (HSSD-200): An Analysis of 3D Scene Scale and Realism Tradeoffs for ObjectGoal Navigation," in *CVPR*.   IEEE, Jun. 2024, pp. 16 384–16 393.

[36] K. Yadav *et al.*, "Habitat challenge 2023," https://aihabitat.org/challenge/2023/, 2023.

[37] S. Bai *et al.*, "Qwen3-VL Technical Report," Nov. 2025. [Online]. Available: http://arxiv.org/abs/2511.21631

[38] T. Cheng *et al.*, "YOLO-World: Real-Time Open-Vocabulary Object Detection," in *CVPR*.   IEEE, Jun. 2024, pp. 16 901–16 911.

[39] C. Zhou *et al.*, "EdgeTAM: On-device track anything model," in *CVPR*, 2025, pp. 13 832–13 842.

[40] L. Zhang *et al.*, "TriHelper: Zero-Shot Object Navigation with Dynamic Assistance," in *IROS*.   IEEE, Oct. 2024, pp. 10 035–10 042.

[41] X. Zhao *et al.*, "ImagineNav: Prompting vision-language models as embodied navigator through scene imagination," in *ICLR*, 2025.

[42] H. Yin *et al.*, "UniGoal: Towards Universal Zero-shot Goal-oriented Navigation," in *CVPR*.   IEEE, Jun. 2025, pp. 19 057–19 066.