ANCHOR: A Physically Grounded Closed-Loop Framework for Robust Home-Service Mobile Manipulation

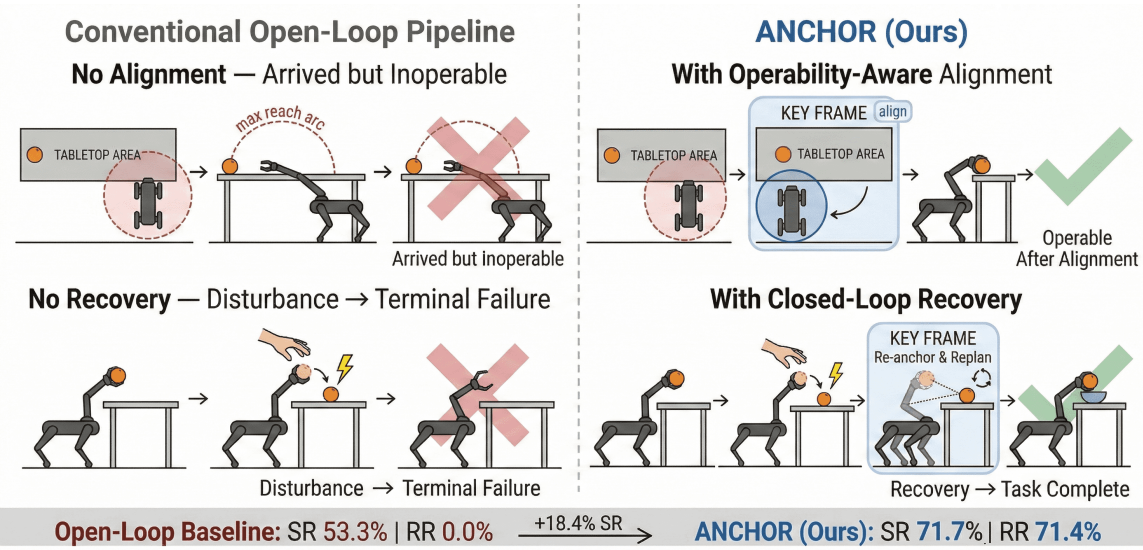

Fig. 1: Comparison between a conventional open-loop pipeline and ANCHOR on two representative failure modes. **Top row**: without operability-aware alignment, the robot navigates to a geometrically reachable but kinematically infeasible pose, leaving the target beyond the arm's workspace (left); ANCHOR actively repositions the base into the operable region before grasping (right). **Bottom row**: when an external disturbance dislodges the grasped object, the open-loop system terminates immediately (left); ANCHOR re-anchors the world state, detects the displaced object, and replans to complete the task (right). Overall, ANCHOR improves the task success rate from 53.3% to 71.7% and achieves a 71.4% recovery rate under disturbances.

***Abstract*— Open-vocabulary mobile manipulation in real domestic environments requires reliable long-horizon execution under open-set object references and frequent disturbances. In practice, many failures arise not from semantic misunderstanding, but from inconsistency between symbolic plans and the evolving physical world. Existing systems often rely on pre-scanned semantic maps that become stale after scene changes, select navigation endpoints without ensuring downstream manipulation feasibility ("arrived-but-inoperable"), and handle anomalies through undifferentiated global replanning. To address this execution inconsistency, we present ANCHOR, a physically grounded closed-loop framework that aligns symbolic reasoning with verifiable physical state during execution. ANCHOR integrates three mechanisms: (i) physically anchored task planning, which binds symbolic predicates to observable geometric anchors and re-validates them after each action; (ii) operability-aware base alignment, which ensures that navigation endpoints satisfy kinematic reachability and local collision feasibility; and (iii) minimum-responsible-layer hierarchical recovery, which localizes failures across perception, base–arm coordination, and execution layers to prevent cascading retries. Across 60 real-robot trials in previously unseen environments, ANCHOR improves task success from 53.3% to 71.7% and achieves a 71.4% recovery rate under perturbations, demonstrating that explicit physical grounding and structured failure containment are critical for robust long-horizon open-vocabulary mobile manipulation.**

I. INTRODUCTION

Autonomous home-service robots are increasingly expected to assist humans with everyday tasks in complex and diverse domestic environments [5]. Delivering on this promise requires open-vocabulary mobile manipulation (OVMM): the ability to navigate, localize objects specified by natural-language expressions, and interact with them physically, without relying on pre-built object catalogs or environment-specific training. Domestic settings further amplify the difficulty. Layouts evolve, objects are open-set, occlusions are frequent, and human activity continually perturbs the scene, making long-horizon execution a robustness problem rather than a one-shot planning problem.

To address these demands, recent OVMM systems integrate vision-language models (VLMs) [9], [12] with modular navigation and manipulation pipelines [1], [2], [4], moving beyond controlled benchmarks toward deployment. Robustness, however, remains the main bottleneck. Small perception errors and external disturbances can accumulate over time and eventually surface as *physical infeasibility* during execution. In particular, many systems rely on *pre-execution* semantic maps that become stale under scene changes [3], [26], choose navigation endpoints that ignore downstream

manipulation feasibility (the "arrived-but-inoperable" problem) [4], [5], and handle anomalies with undifferentiated global replanning or blind retries, which can amplify disturbances instead of containing them [16]. These failure modes call for a system-level design that keeps high-level decisions grounded in up-to-date physical state and provides structured recovery across modules.

We propose ANCHOR, an online closed-loop framework that emphasizes *deployability* and *robustness* via continual state re-anchoring. Rather than deferring feasibility checks to execution, ANCHOR couples perception, planning, and control throughout the task. Unlike black-box end-to-end approaches, ANCHOR's modular recovery mechanism enables predictable, interpretable interventions that are critical for safe interaction in human environments. Concretely, ANCHOR integrates three coupled mechanisms. *Physically anchored task planning* derives symbolic states from real-time sensor observations and re-evaluates them after each action. *Operability-aware base alignment* refines navigation endpoints using manipulator reachability and local collision constraints. *Minimum-responsible-layer recovery* attributes failures across perception, base–arm coordination, and execution, escalating only when necessary. The main contributions of this work are as follows:

- We formulate a state consistency framework grounded in physical anchors, tightly coupling symbolic predicates with continuously verified geometric representations and avoiding stale semantic observations.
- We introduce operability-aware base alignment as a navigation–manipulation consistency constraint, ensuring that navigation endpoints are feasible not only geometrically but also kinematically.
- We design a minimum-responsible-layer recovery hierarchy that localizes interventions, preventing cascading replanning and stabilizing long-horizon execution.

II. RELATED WORKS

A. Open-Vocabulary Mobile Manipulation Systems

Recent advances in vision-language models (VLMs) [9], [12] have significantly accelerated advanced open-vocabulary mobile manipulation. Systems such as OK-Robot [1], MoTo [4], and related embodied VLM pipelines [2], [6] combine open-set object grounding with modular navigation and manipulation stacks, enabling language-conditioned target retrieval and zero-shot interaction in indoor environments [5]. These approaches demonstrate strong semantic generalization and practical system integration under open-category settings. Many existing systems construct or rely on relatively stable scene representations [7], [26]–[28], often involving large-scale reconstruction or semantic aggregation prior to or during early task stages, and subsequent decision making is grounded in this global representation. While such designs provide coherent scene-level reasoning, they typically assume moderate environmental stability and rely on representation maintenance when scene changes occur [8]. In contrast, our work emphasizes continual online state updates during execution. ANCHOR incrementally maintains a semantic scene graph together with grid and octree occupancy maps [27], [28], forming a tightly coupled perception–decision loop in previously unseen and evolving environments.

B. Coupling Between Navigation and Manipulation

Mobile manipulation inherently requires tight coupling between base navigation and arm execution. Conventional navigation frameworks are primarily optimized for geometric distance or reachability objectives [30] and often treat manipulation feasibility as a subsequent stage. As a result, a base pose that is geometrically valid may not always satisfy downstream kinematic reachability, pose alignment, or local collision constraints required for manipulation, particularly in cluttered or spatially constrained environments. Recent work has begun exploring joint reasoning over base positioning and manipulation feasibility [4], [6]; however, many open-vocabulary mobile manipulation systems still adopt a staged pipeline in which navigation precedes arm execution [1], [5]. In this work, we introduce an operability-aware base alignment strategy that refines navigation endpoints by explicitly incorporating manipulator kinematics and local collision feasibility, aiming to improve consistency between navigation outcomes and manipulation requirements.

C. LLM-Based Task Planning in Embodied Systems

Large language models (LLMs) have been widely adopted for high-level task planning in embodied robotics [35]. Recent approaches leverage LLMs to decompose natural language instructions into structured subtask sequences [15], [18], symbolic plans [19], [20], [22], [23], or executable skill invocation logic [32], [33], enhancing flexibility and compositionality in long-horizon tasks. Representative paradigms include symbolic plan generation as well as code-generation approaches such as Code-as-Policies [14] and their embodied extensions [17], [34]. In practice, LLM-based planners are typically integrated with perception and control modules through iterative generate–execute loops [16], [21], enabling adaptation when observations change or execution fails. In many such systems, planning primarily operates at the symbolic or program level, while geometric reachability, pose constraints, and collision feasibility are verified during execution [29]. ANCHOR complements this line of work by introducing a physically anchored planning mechanism that explicitly associates high-level predicates with observable geometric entities and spatial relations. Combined with a hierarchical error attribution and recovery strategy across perception, base–arm coordination, and execution layers [2], [21], our framework aims to improve long-horizon stability while retaining semantic flexibility.

III. METHOD

We propose ANCHOR, an online closed-loop framework for open-vocabulary mobile manipulation in previously unseen environments. At each control cycle, ANCHOR (i) updates the robot state and incrementally maintains a 3D

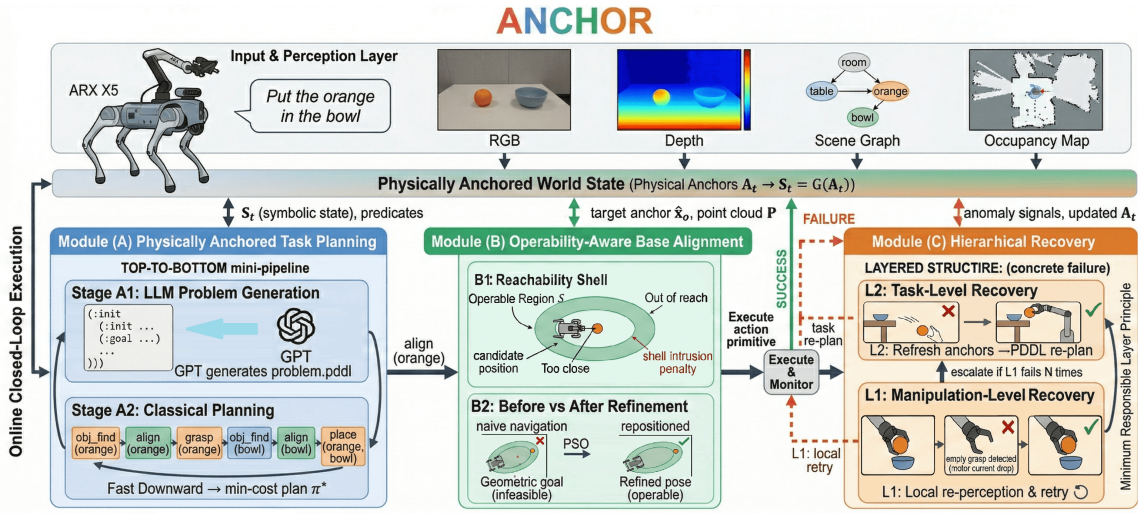

Fig. 2: Overview of the proposed ANCHOR framework. Given an open-vocabulary instruction (e.g., "Put the orange in the bowl"), the system first builds a *Physically Anchored World State* from RGB-D observations, including a 3D scene graph and occupancy maps. (A) *Physically Anchored Task Planning* generates a PDDL-based symbolic plan via an LLM and a classical planner (Fast Downward), executing in a receding-horizon fashion where only the first action is dispatched per cycle. (B) *Operability-Aware Base Alignment* bridges navigation and manipulation by refining the base pose within a dual-ellipsoid reachability shell, transforming a geometrically reachable but potentially infeasible endpoint into an operable one. (C) *Hierarchical Recovery* handles execution failures at the minimum responsible layer: L_1 performs local manipulation-level retries, escalating to L_2 task-level replanning only when necessary. The closed loop ensures that all symbolic states remain grounded in continuously updated physical observations throughout execution.

scene graph together with occupancy representations (grid and octree maps) from RGB-D observations, (ii) plans the next symbolic subgoal grounded in physical anchors, and (iii) executes only the first low-level action with continuous monitoring and recovery. The method is built upon three tightly coupled modules: (A) physically anchored task planning (Sec. III-A), (B) operability-aware base alignment (Sec. III-B), and (C) hierarchical error recovery (Sec. III-C). An overview of the framework is illustrated in Fig. 2.

A. Physically Anchored Task Planning

Physically Anchored Task Planning (PATP) enforces a state-consistency contract: every symbolic predicate must be backed by directly observable, verifiable geometric evidence at execution time. The planner reasons only over the *physically anchored* current world state; after each executed step, perception writes back the updated state and the system replans in a rolling fashion. This closed-loop design mitigates state drift and real-world perturbations in long-horizon execution.

Our pipeline follows the compositional pattern *natural language → PDDL → classical planner → plan* [19]. In contrast to approaches that employ large multimodal models for grounding or constraint synthesis [15], [29], we restrict the LLM to generating a well-formed `problem.pddl` skeleton (GPT-5.1 in our experiments). All geometry-related constraints and predicate evaluations are computed by lightweight, calibrated sensing modules, keeping symbolic generation decoupled from physical anchoring.

1) Physical Anchors and World-State Representation: At each control cycle t, we maintain a set of task-relevant physical anchors A_t and deterministically derive the symbolic state set S_t:

$$S_t = \mathcal{G}(A_t), \tag{1}$$

where $\mathcal{G}(\cdot)$ denotes calibrated, thresholded perception and geometric evaluation rules. By design, we do not use generative models to "fill in" predicate truth values via commonsense completion.

TABLE I: Physical anchors A_t used for predicate evidence.

Category	Examples (non-exhaustive)
Robot anchors	Chassis pose; gripper state & joint currents; onboard sensors; coordinate transforms.
Object anchors	Semantic IDs; 2D detection & segmentation; point clouds from depth or multi-sensor fusion.
Relation anchors	Reachability, proximity, containment/overlap, etc. to support predicate evaluation.

Anchors A_t serve as evidence structures interpreted by predicates and include the following categories (examples shown in Table I).

2) Perception-Driven Predicate Anchor and Update Rules: A continuously running *perception handler* evaluates all domain predicates in each cycle and outputs S_t. Let $\mathbf{x}_r$ denote the robot chassis pose, $\hat{\mathbf{x}}_o$ the expected object anchor returned by `obj_find`, and B_o^{xy}, B_c^{xy} the xy-projected bounding boxes of the object and container point clouds, respectively. The key predicates are defined as follows.
near(r, o) holds when the planar distance $\|\pi_{xy}(\mathbf{x}_r - \hat{\mathbf{x}}_o)\| \leq \epsilon_{\text{near}}$. **aligned**$(r, o)$ is strictly stronger: it additionally requires that the target is stably segmented from the current viewpoint *and* lies within the feasible region of the reachability shell

(Sec. III-B). **holding**(r, o) requires both gripper current confirming a closed, loaded grasp and visual confirmation of the object in the gripper ROI. **in**(o, c) is anchored by the planar overlap ratio

$$\phi_{xy}(o, c) = \frac{\text{area}(B_o^{xy} \cap B_c^{xy})}{\text{area}(B_o^{xy})}, \tag{2}$$

and holds when $\phi_{xy}(o, c) \geq \epsilon_{\text{in}}$. We enforce the domain axiom $\texttt{aligned}(r, o) \Rightarrow \texttt{near}(r, o)$; if `aligned` already holds after `obj_find`, the `align` action is skipped to avoid redundant motion.

3) PDDL Interface and LLM-Based Problem Generation: We adopt a fixed `domain.pddl` defining types, predicates, and four action primitives: `obj_find` (semantic search), `align` (operability-aware base refinement), `grasp` (6-DoF grasp execution), and `place` (container-targeted placement); their implementations are detailed in Sec. III-D. At each planning cycle, the initial state `:init` is populated directly from $S_t = \mathcal{G}(A_t)$, ensuring that all asserted predicates are grounded in calibrated sensor evidence rather than inferred by the LLM via commonsense reasoning. Likewise, PDDL action effects serve only as planner-side *predictions*; actual symbolic updates are committed by the perception handler after physical re-anchoring in the next cycle, so the system always trusts sensor observations over model assumptions.

An LLM generates the `problem.pddl` skeleton from the natural-language instruction, together with open-vocabulary target identifiers (`task_obj`, `task_container`). The LLM is restricted to predefined predicates and actions declared in the fixed domain.

4) Planning and Receding-Horizon Execution: We use a classical planner Fast Downward [24] with the `seq-sat-lama-2011` configuration [25] to quickly obtain feasible plans and to improve plan quality within a fixed time budget. At each planning round, we collect a set of candidate plans Π_t, compare their costs, and select the minimum-cost plan $\pi_t^\star$.

To preserve physical consistency over long horizons, we adopt receding-horizon execution: we execute only the first action a_t of $\pi_t^\star$, then refresh anchors and regenerate `:init` for re-planning, repeating until the goal predicate set is satisfied. The closed-loop procedure is summarized in Algorithm 1.

Algorithm 1 PATP closed-loop planning and receding-horizon execution.

Input: task description in natural language; fixed `domain.pddl`.

Output: executed action sequence until goal predicates are satisfied.

begin
repeat
  Construct physical anchors A_t and derive symbolic state $S_t = \mathcal{G}(A_t)$.
  Populate `:init` in `problem.pddl` using S_t.
  Run Fast Downward (`seq-sat-lama-2011`) to obtain candidate plans Π_t.
  Select $\pi_t^\star = \arg\min_{\pi \in \Pi_t} \text{cost}(\pi)$.
  Execute the first action a_t of $\pi_t^\star$ and monitor the outcome.
  Re-anchor at cycle $t{+}1$ and re-plan.
until goal predicates are satisfied
end

B. Operability-Aware Base Alignment

To eliminate the structural inconsistency between geometric reachability and downstream kinematic feasibility, we reformulate base pose selection as an operability-constrained optimization problem. The module employs an offline-learned reachability surrogate and a runtime safety-aware refinement.

1) Offline Reachability Shell Modeling: We characterize the arm workspace by sampling end-effector poses $\mathbf{p}$ and computing a manipulability score based on the ratio of collision-free IK solutions to total trials:

$$\mu(\mathbf{p}) = \frac{\text{number of collision-free IK solutions}}{\text{number of IK trials}}. \tag{3}$$

Given a threshold μ_{th}, the high-manipulability region is

$$\mathcal{S} = \{\mathbf{p} \mid \mu(\mathbf{p}) \geq \mu_{\text{th}}\}. \tag{4}$$

To enable efficient online optimization, we approximate $\mathcal{S}$ with a dual-ellipsoid shell surrogate:

$$d_{\text{out}}(\mathbf{p}) = (\mathbf{p} - \mathbf{c}_{\text{out}})^\top \mathbf{E}_{\text{out}} (\mathbf{p} - \mathbf{c}_{\text{out}}), \tag{5}$$

and an inner ellipsoid

$$d_{\text{in}}(\mathbf{p}) = (\mathbf{p} - \mathbf{c}_{\text{in}})^\top \mathbf{E}_{\text{in}} (\mathbf{p} - \mathbf{c}_{\text{in}}), \tag{6}$$

where $d(\mathbf{p}) = (\mathbf{p} - \mathbf{c})^\top \mathbf{E} (\mathbf{p} - \mathbf{c})$. The shell is defined by $d_{\text{out}}(\mathbf{p}) \leq 1$ and $d_{\text{in}}(\mathbf{p}) \geq 1$. Unlike concentric approximations, we allow a center offset $\mathbf{o} = \mathbf{c}_{\text{in}} - \mathbf{c}_{\text{out}}$ to capture kinematic anisotropy. This differentiable surrogate replaces dense reachability grids, reducing memory footprint while enabling closed-form feasibility checks during runtime optimization.

2) Online Base Pose Refinement: Given a desired end-effector pose $\mathbf{T}_{\text{ee}}$, we optimize the base pose $\mathbf{x} = [x, y, \theta]^\top$ using a safety-aware objective:

$$J(\mathbf{x}) = w_a J_{\text{align}} + w_s J_{\text{shell}} + w_c J_{\text{chassis}}. \tag{7}$$

where the alignment term encourages a favorable approach direction,

$$J_{\text{align}} = 1 - \hat{\mathbf{d}}_{\text{base}}^\top \hat{\mathbf{d}}_{\text{ee}}, \tag{8}$$

with $\hat{\mathbf{d}}_{\text{base}}$ as the base heading and $\hat{\mathbf{d}}_{\text{ee}}$ as the projected end-effector approach direction.

Let $\mathcal{P}$ denote the observed point cloud. We introduce the shell risk term because target reachability alone is insufficient; obstacles inside the operable shell can still cause failures during approach and local adjustment. We therefore penalize shell intrusion by

$$J_{\text{shell}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \sigma\big(\alpha(1 - d_{\text{out}}(\mathbf{p}))\big) \cdot \sigma\big(\alpha(d_{\text{in}}(\mathbf{p}) - 1)\big), \tag{9}$$

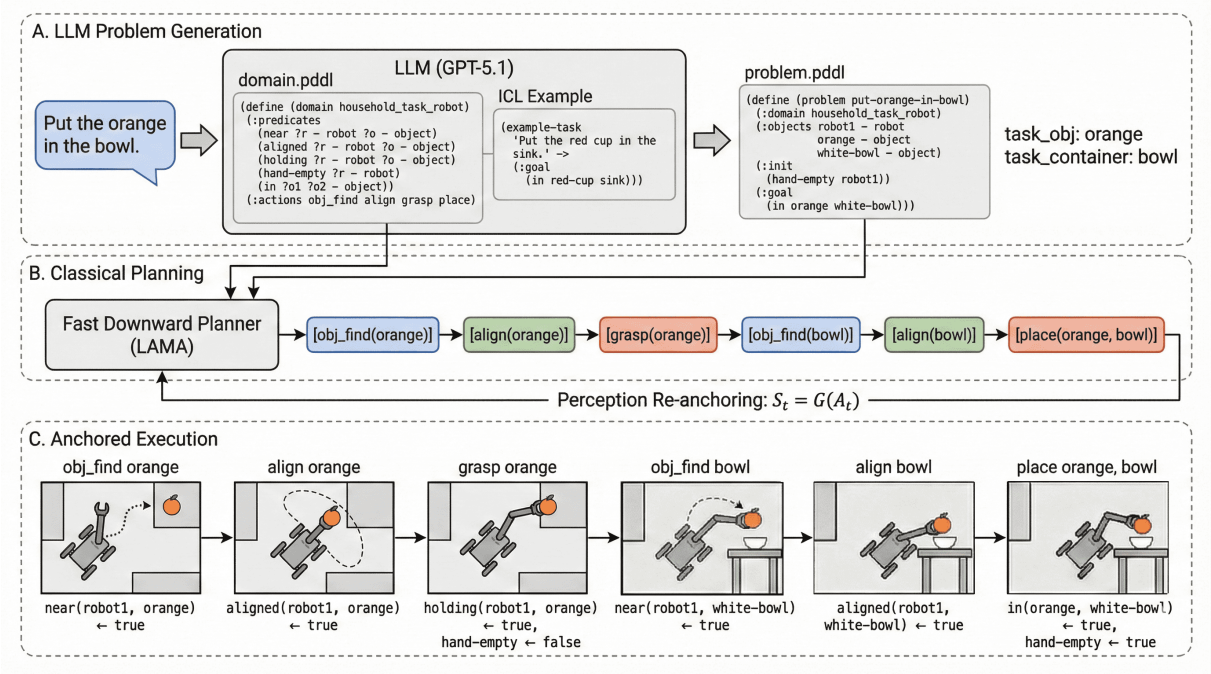

Fig. 3: Physically Anchored Task Planning pipeline. (A) The LLM receives the domain PDDL and generates a `problem.pddl` with open-vocabulary targets based on the in-context example (e.g., "Put the orange in the bowl"). (B) A classical planner produces a candidate action sequence over the fixed domain. (C) Execution proceeds in a receding-horizon fashion: the perception handler re-anchors the world state ($S_{t+1} = \mathcal{G}(A_{t+1})$) after each action, ensuring high-level plans stay grounded in evolving physical observations.

where $\sigma(\cdot)$ is a smooth sigmoid approximation. This is a continuous relaxation of binary shell membership $\mathbb{I}(d_{\text{out}}(\mathbf{p}) \leq 1,\ d_{\text{in}}(\mathbf{p}) \geq 1)$. The smooth relaxation improves optimization stability while preserving the operability semantics.

We additionally include a chassis footprint risk term to penalize collisions between the quadruped base and $\mathcal{P}$:

$$J_{\text{chassis}} = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \phi((r_b + \delta) - \|\mathbf{p}_{xy} - \mathbf{c}_b\|)\,, \tag{10}$$

where $\phi(\cdot)$ is a softplus function. The resulting non-convex problem is solved with particle swarm optimization [31] in a two-stage coarse-to-fine schedule, yielding an $SE(2)$ base pose for the navigation stack. In practice, the shell surrogate and smooth risk terms make the optimization both tractable and robust for online mobile manipulation. A schematic visualization of the fitted dual-ellipsoid shell and its role in online refinement is shown in Fig. 4.

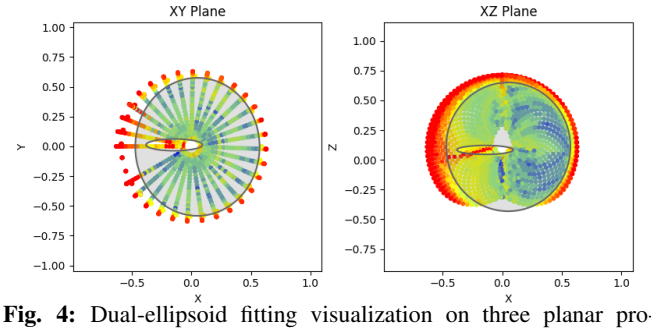

Fig. 4: Dual-ellipsoid fitting visualization on three planar projections of sampled end-effector poses. The color encodes the manipulability score $\mu(\mathbf{p})$ (Eq. 3): **blue** indicates a high ratio of collision-free IK solutions (high dexterity), while **red** indicates few valid IK solutions (low dexterity). The fitted dual-ellipsoid shell (overlaid curves) provides a compact surrogate boundary of the high-manipulability workspace.

C. Hierarchical Closed-loop Recovery

To ensure robust execution in disturbance-prone and previously unseen environments, ANCHOR employs a hierarchical recovery mechanism guided by a "minimum responsible layer" principle. Unlike traditional open-loop systems [1] that terminate upon the first execution anomaly, our framework categorizes failures into two levels based on their impact on the task state, enabling targeted interventions that minimize redundant movements. We distinguish routine receding-horizon replanning, which updates actions based on refreshed physical state during nominal execution, from recovery-triggered replanning, which is invoked only when execution anomalies invalidate the current task state.

1) L_1 Manipulation-level Recovery: This layer addresses local execution slips during grasping or placing. Anomalies such as an empty grasp, a mid-motion gripper slip, or a placement miss are detected directly from low-level sensor signals (e.g., a sudden drop in gripper motor current or a mismatch between expected and observed gripper state from the wrist camera), requiring no task-level reasoning. Upon detection, ANCHOR performs local re-perception and re-alignment without re-navigating. If the failure persists beyond N retry attempts, the system escalates to L_2.

2) L_2 Task-level Recovery: This layer handles larger environmental shifts, such as the target being displaced out of the workspace. Upon escalation from L_1 or detection of a scene-graph change by the VLM, the system refreshes A_t, updates the PDDL state, and invokes the planner to generate a new action sequence (e.g., reverting from `grasp` to `obj_find`). The task is terminated as a failure only when the planner determines the goal is unreachable or the target is confirmed missing from the scene.

D. Action-Primitives Execution

We instantiate each symbolic action as an executable action primitive that operates on the shared anchor set A_t and is validated through re-anchoring.

1) `find_obj`: The `obj_find` primitive performs open-vocabulary target search and returns an expected arrival anchor $\hat{\mathbf{x}}_o$ for `task_obj`. It jointly maintains a grid exploration map and a hierarchical scene graph over room–region–object entities, incrementally updated from RGB-D observations via open-vocabulary detection and segmentation [10], [11]. Frontier candidates are scored by combining per-region target-existence belief with travel cost, so already-exhausted regions are automatically down-weighted in favor of under-explored areas [30]. The primitive repeats a closed-loop *perceive–update–navigate* cycle until the target anchor stabilizes, at which point $\hat{\mathbf{x}}_o$ is passed to downstream primitives and predicate checks.

2) `align`: The `align` primitive invokes the operability-aware base-pose refinement (Sec. III-B) using the latest target anchor in A_t (or $\hat{\mathbf{x}}_o$ if the object is momentarily undetectable), executes a short-range $SE(2)$ motion, and returns `success` only when re-anchoring confirms the `aligned` predicate; otherwise the hierarchical recovery policy is triggered.

3) `grasp`: The `grasp` primitive generates a 6-DoF end-effector command using AnyGrasp [13] on open-vocabulary segmented point clouds [10], [11]. To suppress single-frame jitter, candidates from two consecutive RGB-D frames are matched by nearest-neighbor pose distance; only temporally consistent pairs (within translation and rotation tolerances) are retained and ranked by

$$\text{Score}(g^{(t)}, g^{(t-1)}) = F(g^{(t)}) \cdot R\big(g^{(t)}, g^{(t-1)}\big), \qquad (11)$$

where $F(g)$ applies a thresholded tilt penalty on the end-effector roll–pitch–yaw relative to the gravity-aligned approach direction (small deviations tolerated, large ones sharply penalized), and $R(\cdot)$ combines temporal consistency with the raw AnyGrasp confidence score.

4) `place`: Given `task_container`, the `place` primitive segments the target container, computes a drop pose from its point-cloud centroid and surface height (plus a safety clearance), and releases the object.

IV. EXPERIMENTS

A. Experimental Setup

1) Robot Hardware: We evaluate ANCHOR on a mobile manipulation platform comprising a Unitree Go2 quadruped robot and an ARX X5 6-DOF manipulator. The robot is equipped with a Livox Mid-360 LiDAR for SLAM and navigation, and an Intel RealSense D435i depth camera mounted on the wrist for active perception and manipulation. The system is powered by an onboard NVIDIA GeForce RTX 3090 GPU, which enables real-time foundation model inference and high-frequency execution of the sense–plan–act loop.

2) Environments and Tasks: We conduct experiments in two previously unseen indoor environments (a laboratory and a home office) featuring five categories of daily objects with varying geometries. We perform 20 trials per difficulty level (60 in total). Following the long-horizon OVMM protocol [5], each trial requires the robot to: (i) explore the scene to locate a target, (ii) perform operability-aware base alignment, (iii) grasp the object, (iv) navigate to a receptacle, and (v) place it. To rigorously evaluate the framework's robustness, we categorize tasks into three levels: **Level 1 (Direct)**: the target is within the initial field of view (FoV) and reachable without navigation; **Level 2 (Navigation)**: the target is out of sight, requiring long-range navigation and global planning; **Level 3 (Disturbed)**: building on Level 2, we introduce unannounced perturbations during execution, such as displacing the object after the robot arrives or temporarily occluding the target during the approach.

B. Baselines

Directly comparing against existing OVMM systems is challenging because many approaches are not publicly available and faithful reproduction on different hardware is impractical. We therefore adapt OK-Robot [1] as a controlled baseline.

OK-Robot* (Adapted Baseline): OK-Robot* shares the same real-time VLM-based perception and navigation stack as ANCHOR, but retains the original open-loop, decoupled pipeline: upon reaching the VLM-suggested endpoint, it attempts a single-shot grasp without operability-aware refinement or closed-loop recovery. We compare this baseline against the full ANCHOR framework and two ablated variants (Sec. IV-D).

C. Metrics

Success Rate (SR): The percentage of trials in which the robot completes the full pick-and-place sequence without human intervention.

Step-wise Success Rate (SSR): Per-stage success for Find, Align, Grasp, and Place, pinpointing which module contributes most to overall failures.

Recovery Rate (RR): The fraction of detected execution or perception anomalies that are successfully resolved by L_1 or L_2 recovery without terminating the task.

D. Main Results

Table II reports per-level success rates. At Level 1 (target in FoV, no navigation), both methods perform similarly because the short interaction distance leaves little room for alignment or recovery to contribute. At Level 2, ANCHOR

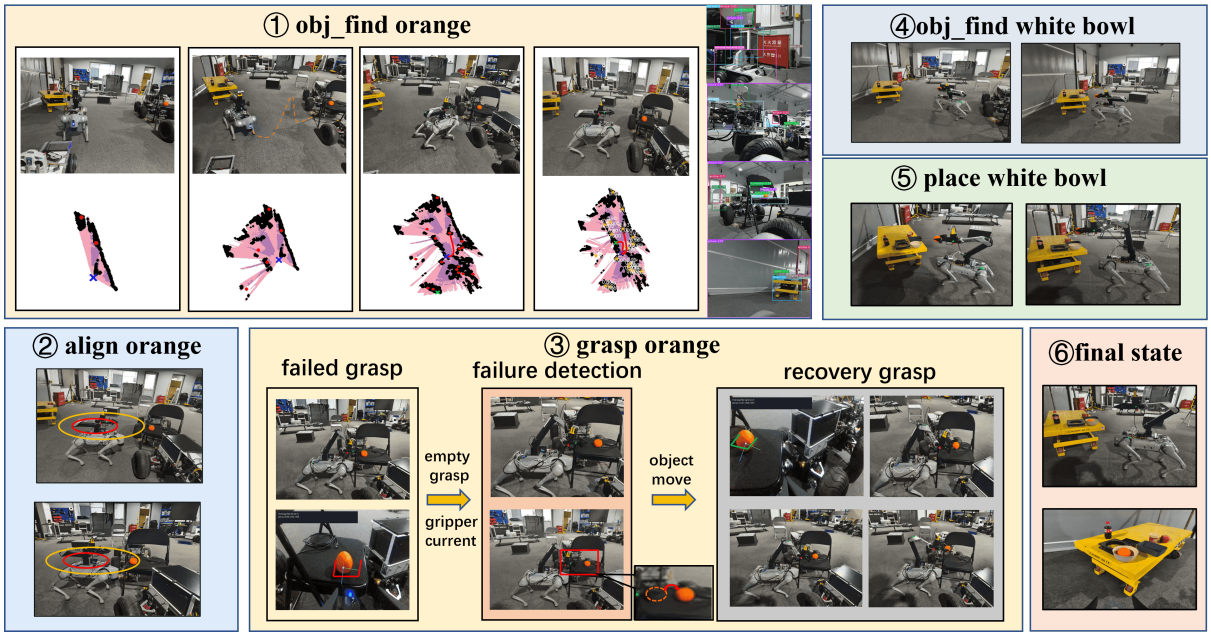

Fig. 5: Qualitative real-robot trial illustrating closed-loop recovery and operability-aware alignment. ① Open-vocabulary search locates the target orange. ② The base alignment module refines a geometrically reachable but kinematically infeasible pose to ensure manipulation feasibility. ③ An initial grasp fails (empty grasp detected via gripper current); L_1 recovery triggers local re-perception and re-grasp without re-navigation. ④–⑤ The robot locates the container and completes placement. The sequence demonstrates that operability-aware alignment prevents "arrived-but-inoperable" failures, while hierarchical recovery handles execution-level anomalies with minimal cost.

TABLE II: Success rate breakdown by difficulty level (20 trials each). Level 1–3 are defined in Sec. IV-A.

Method	Level 1	Level 2	Level 3	Overall
OK-Robot*	80.0%	55.0%	25.0%	53.3%
ANCHOR (Ours)	**85.0%**	**75.0%**	**55.0%**	**71.7%**

TABLE III: Overall task performance and Ablation Study.

Method	SR ↑	RR ↑	Steps ↓	Time ↓
OK-Robot [1] (Orig.)	58.5%	–	–	–
OK-Robot* (Adapted)	53.3%	–	**6.2**	**2.4m**
ANCHOR (Ours full)	**71.7%**	**71.4%**	8.5	3.1m
w/o Base Alignment	60.0%	60.0%	9.2	3.5m
w/o Hier. Recovery	55.0%	0.0%	7.1	2.8m

gains 20 pp: the 20 pp improvement at Level 2 indicates that navigation–manipulation inconsistency, rather than perception failure, is a dominant bottleneck in open-vocabulary mobile manipulation. At Level 3, the gap widens to 30 pp because the baseline has no recovery mechanism—any mid-task perturbation (e.g., object displaced during approach) leads to immediate failure, while ANCHOR's L_1/L_2 recovery salvages the majority of such anomalies. Table III further summarizes the overall SR, RR, and efficiency.

Step-wise Breakdown: As shown in Table IV, both methods achieve the same finding rate (70.0%), but ANCHOR's grasping rate (80.0% vs. 69.2%) confirms that operability-aware base alignment effectively bridges the navigation–manipulation gap. A representative trial is shown in Fig. 5.

TABLE IV: Step-wise Success Rate (SSR) Breakdown.

Method	Find	Align	Grasp	Place
OK-Robot*	70.0%	–	69.2%	88.9%
Ours	70.0%	92.8%	**80.0%**	**100.0%**

TABLE V: Analysis of Hierarchical Recovery Efficiency.

Method	L1 Rec.	L2 Rec.	Total Recov.	RR ↑
OK-Robot*	0/17	0/11	0/28	0.0%
Ours	13/17	7/11	20/28	**71.4%**

E. Ablations

We evaluate the necessity of ANCHOR's core components by comparing the full framework against two stripped-down variants (Table III).

1) Synergy of Alignment and Recovery: Removing the base alignment module leads to an 11.7 pp drop in SR. Without alignment, the robot often attempts grasps from suboptimal poses, increasing the likelihood of execution failures. While hierarchical recovery can mitigate some of these errors, the lack of physical grounding at the start makes the task inherently more difficult.

2) Effectiveness of Hierarchical Recovery: Table V provides a detailed analysis of the recovery mechanism. Our full framework achieves a Recovery Rate (RR) of 71.4%, successfully salvaging 13 out of 17 L_1 anomalies (e.g., re-grasping after a slip, as illustrated in Fig. 5) and 7 out of 11 L_2 anomalies (e.g., re-planning after object displacement). In contrast, removing this layer (w/o Hier. Recovery) or using the baseline OK-Robot* results in immediate terminal failure upon any disturbance. This demonstrates that robust long-

horizon stability is a direct result of our tiered intervention strategy.

F. Failure Analysis

Despite the improvements, we identified two primary failure modes: **(i) Perceptual Ambiguity**: the VLM occasionally misidentifies objects with similar visual textures in cluttered scenes, leading to task-level errors. **(ii) Sensing Noise**: depth inaccuracies from the D435i camera, particularly on reflective surfaces, can provide erroneous inputs to the base alignment module. While ANCHOR attempts to re-align, extreme initial estimation errors can still lead to reachability failures.

V. CONCLUSIONS

We presented ANCHOR, an online closed-loop framework for robust open-vocabulary mobile manipulation in unseen environments. By combining operability-aware base alignment with physically anchored task planning and hierarchical recovery, ANCHOR effectively mitigates common failure modes—particularly the "arrived-but-inoperable" issue. Across 60 real-robot trials spanning three difficulty levels, ANCHOR achieves a 71.7% task success rate (vs. 53.3% for the baseline) and recovers from 71.4% of detected execution anomalies, confirming that explicit physical grounding and tight navigation–manipulation coupling are more effective than undifferentiated global replanning. Future work will extend the framework to articulated object manipulation (e.g., opening doors) and interactive navigation in cluttered real-world settings.

REFERENCES

[1] P. Liu, Y. Orru, J. Paxton, N. M. M. Shafiullah, and L. Pinto, "OK-Robot: What really matters in integrating open-knowledge models for robotics," in *Proc. Robotics: Science and Systems (RSS)*, 2024.

[2] P. Zhi, Z. Zhang, Y. Zhao, M. Han, Z. Zhang, Z. Li, Z. Jiao, B. Jia, and S. Huang, "Closed-loop open-vocabulary mobile manipulation with GPT-4V," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2025.

[3] Z. Yan, S. Li, Z. Wang, L. Wu, H. Wang, J. Zhu, L. Chen, and J. Liu, "Dynamic open-vocabulary 3D scene graphs for long-term language-guided mobile manipulation," *IEEE Robot. Autom. Lett.*, 2025.

[4] Z. Wu, A. Ma, X. Xu, H. Yin, Y. Liang, Z. Wang, and J. Lu, "MoTo: A zero-shot plug-in interaction-aware navigation for general mobile manipulation," in *Proc. Conf. Robot Learning (CoRL)*, PMLR, vol. 305, pp. 2933–2948, 2025.

[5] S. Yenamandra *et al.*, "HomeRobot: Open-vocabulary mobile manipulation," in *Proc. Conf. Robot Learning (CoRL)*, 2023.

[6] D. Honerkamp, M. Buchner, F. Despinoy, T. Welschehold, and A. Valada, "Open-vocabulary mobile manipulation in unseen dynamic environments with 3D semantic maps," *arXiv preprint arXiv:2406.18115*, 2024.

[7] D. Honerkamp, M. Buchner, F. Despinoy, T. Welschehold, and A. Valada, "Language-grounded dynamic scene graphs for interactive object search with mobile manipulation," *arXiv preprint arXiv:2403.08605*, 2024.

[8] B. Memmel, J. Paxton, and D. Fox, "Online dynamic spatio-semantic memory for open world mobile manipulation," *arXiv preprint arXiv:2411.04999*, 2024.

[9] A. Radford *et al.*, "Learning transferable visual models from natural language supervision," in *Proc. Int. Conf. Mach. Learning (ICML)*, 2021.

[10] S. Liu *et al.*, "Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2024.

[11] A. Kirillov *et al.*, "Segment anything," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023.

[12] OpenAI, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[13] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu, "AnyGrasp: Robust and efficient grasp perception in spatial and temporal domains," *IEEE Trans. Robot.*, vol. 39, no. 5, pp. 3929–3945, 2023.

[14] J. Liang *et al.*, "Code as policies: Language model programs for embodied control," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023.

[15] M. Ahn *et al.*, "Do as I can, not as I say: Grounding language in robotic affordances," in *Proc. Conf. Robot Learning (CoRL)*, 2022.

[16] W. Du, Y. Yang, Y. Du, J. Tenenbaum, and D. Schuurmans, "Closed-loop long-horizon robotic planning via equilibrium sequence modeling," *arXiv preprint arXiv:2410.01440*, 2024.

[17] I. Singh *et al.*, "ProgPrompt: Generating situated robot task plans using large language models," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023.

[18] Y. Zhi, Z. Jiang, Y. Gao, and J. Suh, "Grounding LLMs for robot task planning using closed-loop state feedback," *arXiv preprint arXiv:2402.08546*, 2024.

[19] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone, "LLM+P: Empowering large language models with optimal planning proficiency," *arXiv preprint arXiv:2304.11477*, 2023.

[20] K. Rana *et al.*, "SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning," in *Proc. Conf. Robot Learning (CoRL)*, 2023.

[21] M. Skreta, Z. Zhou, J. L. Yuan, K. Darvish, A. Aspuru-Guzik, and A. Garg, "RePLan: Robotic replanning with perception and language models," *arXiv preprint arXiv:2401.04157*, 2024.

[22] F. Wang, S. Lyu, P. Zhou, A. Duan, G. Guo, and D. Navarro-Alarcon, "Instruction-augmented long-horizon planning: Embedding grounding mechanisms in embodied mobile manipulation," *arXiv preprint arXiv:2503.08084*, 2025.

[23] H. Ye, Y. Xiao, C. Lu, and P. Cai, "UniPlan: Vision-language task planning for mobile manipulation with unified PDDL formulation," *arXiv preprint arXiv:2602.08537*, 2025.

[24] M. Helmert, "The Fast Downward planning system," *J. Artif. Intell. Res.*, vol. 26, pp. 191–246, 2006.

[25] S. Richter and M. Westphal, "The LAMA planner: Guiding cost-based anytime planning with landmarks," *J. Artif. Intell. Res.*, vol. 39, pp. 127–177, 2010.

[26] Q. Gu *et al.*, "ConceptGraphs: Open-vocabulary 3D scene graphs for perception and planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2024.

[27] N. Hughes, Y. Chang, S. Hu, R. Talak, R. Abdulhai, J. Strader, and L. Carlone, "Foundations of spatial perception for robotics: Hierarchical representations and real-time systems," *Int. J. Robot. Res.*, vol. 43, no. 6, pp. 765–816, 2024.

[28] L. Werby, K. Schmid, O. Bennewitz, and W. Burgard, "HOV-SG: Hierarchical open-vocabulary 3D scene graphs for language-grounded robot navigation," in *Proc. Robot.: Sci. Syst. (RSS)*, 2024.

[29] W. Huang *et al.*, "VoxPoser: Composable 3D value maps for robotic manipulation with language models," in *Proc. Conf. Robot Learning (CoRL)*, 2023.

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

[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw. (ICNN)*, vol. 4, pp. 1942–1948, 1995.

[32] A. Brohan *et al.*, "RT-2: Vision-language-action models transfer web knowledge to robotic control," in *Proc. Conf. Robot Learning (CoRL)*, 2023.

[33] M. J. Kim *et al.*, "OpenVLA: An open-source vision-language-action model," in *Proc. Conf. Robot Learning (CoRL)*, 2024.

[34] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile ALOHA: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," *arXiv preprint arXiv:2401.02117*, 2024.

[35] R. Firoozi *et al.*, "A survey on robotics with foundation models: Toward embodied AI," *arXiv preprint arXiv:2402.02385*, 2024.