Visibility-Aware Mobile Grasping in Dynamic Environments

Author Names Omitted for Anonymous Review

***Abstract*—This paper addresses the problem of mobile grasping in dynamic, unknown environments where a robot must operate under a limited field-of-view. The fundamental challenge is the inherent trade-off between "seeing" around to reduce environmental uncertainty and "moving" the body to achieve task progress in a high-dimensional configuration space, subject to visibility constraints. Previous approaches often assume known or static environments and decouple these objectives, failing to guarantee safety when unobserved dynamic obstacles intersect the robot's path during manipulation. In this paper, we propose a unified mobile grasping system comprising two core components: (1) an iterative low-level whole-body planner coupled with velocity-aware active perception to navigate dynamic environments safely; and (2) a hierarchical high-level planner based on behavior trees that adaptively generates subgoals to guide the robot through exploration and runtime failures. We provide experimental results across 400 randomized simulation scenarios and real-world deployment on a Fetch mobile manipulator. Results show that our system achieves a success rate of 68.8% and 58.0% in unknown static and dynamic environments, respectively, significantly boosting success rates by 22.8% and 18.0% over the navigation-and-manipulation approach in both unknown static and dynamic environments, with improved collision safety.**

I. Introduction

Robots are entering household environments [27, 54]. We consider the problem of a mobile manipulator in a cluttered, unknown, and dynamic home, commanded by a seemingly simple task of mobile grasping: picking up a can from a shelf across the room. If the robot has a pre-built, perfect map of the house and all obstacles are static, this will be a classical mobile manipulation problem [38, 46]. But if the environment contains unknown clutter (e.g., a backpack left on the floor) or dynamic agents (e.g., a person walking by), the robot must rely on its sensors to guarantee safety. Classical mobile manipulation systems struggle with the severe partial observability and usually require well-defined goal configurations. Recent data-driven methods often fail to generalize to unseen room layouts and unknown obstacles [54]. Yet, this challenging setting is required every day for many object-picking tasks in human environments, like in our home. In this paper, we focus on solving this challenging task regime: the robot assumes minimal information about the environment (e.g., with most obstacles unknown), and is equipped with a movable head sensor of limited field-of-view, which is the common setting of many robots [14]; the grasp configuration is underspecified; and the trajectory to the goal is possibly blocked by unknown, changing obstacles. In this domain, the robot must plan a trajectory to the target, but it must strictly account for visibility, ensuring it does not collide with obstacles that enter its blind spots while it focuses on the task.

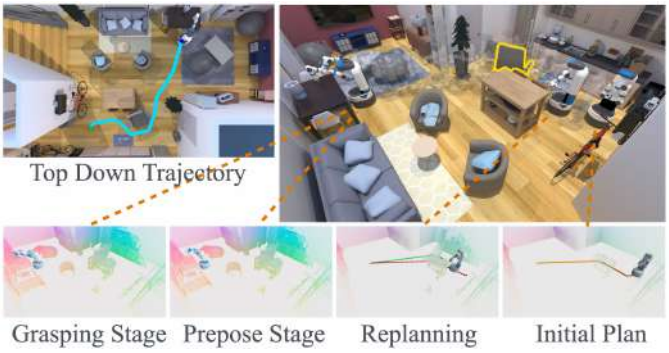

Fig. 1. **Visibility-Aware Mobile Grasping.** A mobile manipulator approaches a target while a chair appears in its path.

For this task, the objectives of reliable manipulation and safe moving among obstacles are often in direct conflict. The robot faces a critical "visibility allocation" problem regarding its gaze. For example, to successfully grasp the target, it must direct its sensors toward it to perceive and refine the underspecified grasping goal. However, to approach the target safely, it must scan the environment to detect any possible dynamic obstacles that may occlude its path. Simultaneously, after gathering new observations of the environments, the robot must update its internal representation and replan its whole-body trajectory in real-time to respond to newly modeled obstacles or the refined target configuration. We illustrate an example in Figure 1: the robot plans a trajectory toward the can, but a chair is suddenly placed in its path. By continuously updating its environment estimation through active perception, the robot detects the obstacle and replans a collision-free whole-body trajectory to complete the grasp.

Planning over a large-scale, partially observable, dynamic environment with underspecified goal configurations poses fundamental challenges. To address this issue, we propose a unified, hierarchical mobile grasping system that tightly couples perception and planning within a receding-horizon loop. It comprises two core components: (1) a low-level whole-body motion planner coupled with a velocity-aware active perception policy that prioritizes observations based on movement speed, to ensure motion efficiency and safety in dynamic environments; (2) a high-level policy that monitors the execution online, generates subgoals such as observation or pre-grasp poses, and recovers from failures if encountered. During mobile manipulation, it iteratively optimizes whole-body trajectories and gaze directions at runtime, enabling

the robot to navigate among dynamic environments while progressively refining its manipulation goals. This hierarchical design efficiently avoids the original exhaustive, intractable search problem by injecting high-level task priors. It leverages recent advances in real-time motion planners [43], which close the loop for safe low-level motions even with limited visibility.

We evaluate our system across 400 randomized simulation scenarios and real-world trials on a Fetch mobile manipulator. The experiments cover diverse settings, ranging from static environments with unknown layouts to dynamic scenarios where obstacles suddenly appear in the robot's blind spots. Results demonstrate that our approach achieves a success rate of 68.8% and 58.0% in unknown static and dynamic environments, respectively, improving by 22.8% and 18.0% over the navigation-and-manipulation baseline in unknown static and dynamic environments, respectively. The benefits mainly come from improved collision rates through the real-time interleaved whole-body motion planning and active perception, which ensures robots proceed to finish the task safely.

II. Related Work

A. Mobile Manipulation Systems

Mobile manipulation has seen significant progress in recent years, with deployed systems achieving success in controlled settings such as retail environments and long-term human-shared spaces [2, 6, 9, 33, 39, 52]. These systems typically rely on pre-built 3D maps and handle dynamic obstacles through simple 2D lidar-based detection, limiting their ability to operate safely in environments with rich 3D structure and frequent changes. Recent work addressing partial observability and dynamic environments [10, 15, 45, 47] has made progress in reactive planning under uncertainty, but these approaches are often evaluated in simplified simulation settings or low-dimensional configuration spaces, leaving unaddressed the challenges of real-time whole-body motion safety in cluttered, dynamic 3D scenes.

To manage the high dimensionality of mobile manipulation, prevailing approaches decompose the problem through base placement optimization [38, 46, 56, 58, 60], which selects a fixed robot configuration that maximizes manipulability for a known goal. While this decoupling strategy is computationally efficient, the base placement calculation usually assumes nearly complete environment knowledge and cannot handle goals that must be refined incrementally as observations accumulate. Our hierarchical subgoal generation draws inspiration from these methods but extends them to partial observability, adaptively refining intermediate configurations as the environment map evolves rather than requiring a complete model upfront.

B. Visibility-Aware Planning and Active Perception

The integration of perception and motion planning has been explored from two complementary perspectives: enforcing visibility constraints during motion and actively directing perception to reduce uncertainty. Early visibility-aware planners [17, 21] constrain motion to previously observed free space, ensuring collision-free paths but assuming static environments where observed regions remain unchanged. More recent work [32] jointly optimizes gaze and motion to maintain continuous visibility of a target object, but focuses on fixed manipulation goals rather than progressively refining ambiguous goal configurations.

Active perception approaches take the complementary strategy of selecting viewpoints to maximize information gain. Exploration planners [4, 5] target frontier regions to build complete environment maps, while task-specific methods optimize viewpoints for grasping through affordance-driven next-best-view planning [7, 34, 59] or belief space reasoning [62]. Recent work [24, 31] further couples perception with manipulation actions to actively reduce occlusions. However, these methods typically separate viewpoint selection from motion execution or do not explicitly reason about swept-volume safety during dynamic replanning. Our velocity-aware gaze policy differs by jointly considering motion safety and task objectives: we prioritize observation of the robot's future swept volume based on planned velocity, ensuring proactive collision monitoring while simultaneously refining manipulation goals.

Advances in motion planning provide the computational tools necessary for real-time replanning. Reactive controllers [19] enable local collision avoidance, while sophisticated planners based on optimization [35], sampling [29, 50, 53], and hybrid methods achieve near real-time performance. Recent algorithmic innovations, including vectorized sampling [43] and advanced tree search [49], dramatically reduce planning latency to enable frequent replanning. Our system builds on these computational advances, employing vectorized collision checking within a receding-horizon framework that tightly couples perception updates with whole-body replanning to respond to both dynamic obstacles and incremental map refinement.

C. Learning-Based Mobile Manipulation

Learning-based approaches to mobile manipulation have demonstrated impressive capabilities through diverse paradigms. Modular systems [1, 20, 27, 28, 54, 55] ground high-level language instructions into learned environment representations or pre-trained skills, achieving flexible task execution. Reinforcement learning methods [23, 36, 48, 57] learn end-to-end or modular policies through large-scale training, incorporating behavior priors to improve generalization. Most recently, vision-language-action models and embodied foundation models [8, 12, 22, 44, 51, 61] leverage web-scale data to achieve remarkable breadth in task variety and semantic reasoning, yet exhibit limitations in generalization to unseen layouts and objects.

While these learning approaches show promising results, they typically operate without explicit geometric reasoning about collision risks and can be sensitive to distributional shift. Our work provides a complementary model-based alternative that explicitly reasons about visibility constraints and swept-volume collision checking within a closed-loop planning

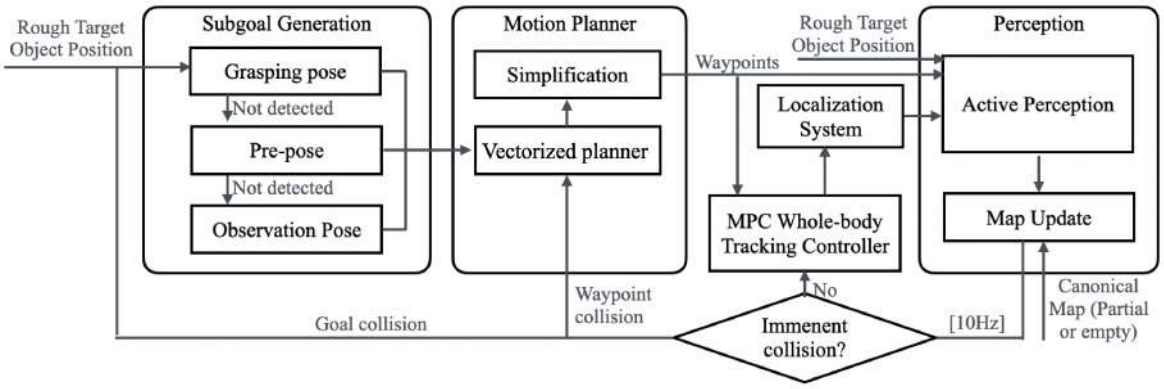

Fig. 2. **System Architecture Overview.** The receding-horizon framework iteratively refines the belief state $b_t = (q_t, M_t, g_t)$ through four coupled components: hierarchical subgoal policy generates intermediate goals g_t, active perception policy directs gaze while updating map M_t, whole-body motion policy computes collision-free trajectories ξ_t to g_t, and localization and control executes the trajectory while tracking state q_t. Replanning is triggered when observations invalidate the current trajectory or subgoals are reached.

framework, prioritizing safety through continuous perception-planning feedback in dynamic, unstructured environments.

III. Problem Formulation

We address visibility-aware mobile grasping in dynamic, partially observed environments. Given an initial configuration q_0 and a target coordinate $p \in \mathbb{R}^3$, which can be derived from a user-specified 2D point on the image or grounded through a visual grounding module [11], the objective is to compute a control policy that generates a whole-body trajectory $\xi : [0, T] \to \mathcal{C}$ to achieve a stable grasp, and realize it. We assume minimal information about the environment, including unknown maps, grasp configurations, and obstacle information. Moreover, the environment contains dynamic obstacles with unknown trajectories. The fundamental challenge of this problem is safety under partial observation, which means that the robot should minimize the risk of collision between its swept volume $V(\xi)$ and the obstacles in blind spots, while completing the grasping task efficiently.

We formulate the mobile grasping problem as a decision-making problem defined by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{G} \rangle$. The state space $\mathcal{S}$ comprises the robot configuration q and the true environment state $\underline{M_t}$, i.e., the underlying ground truth full 3-D geometric representation. The action space $\mathcal{A} = \mathcal{A}_r \times \mathcal{A}_v$ enables simultaneous whole-body motion and gaze control. The transition model $\mathcal{T}$ governs system transition, including the known robot transition and the unknown dynamic object transitions. Finally, $\mathcal{G} \subset \mathcal{C}$ defines the set of underlying true grasp configurations of the robot. Since the robot lacks access to the true state, dynamics, and grasp configurations, it maintains an internal belief state $b_t = (q_t, M_t, g_t)$, where q_t is the robot configuration at t, M_t is a volumetric partial estimation for $\underline{M_t}$ that is incrementally updated via observations $o_t \in \mathcal{O}$, and g_t is the estimated grasp configuration.

We formulate a policy $\pi = (\pi_r, \pi_v, \pi_g)$ operating on the belief state b_t: (1) Motion Policy $\pi_r(b_t) \to \xi_t$: Generates a whole-body trajectory ξ_t reaching the current estimated goal g_t that is collision-free regarding the current map M_t. (2) Perception Policy $\pi_v(\xi_t, p) \to a_v \in \mathcal{A}_v$: Optimizes sensor orientation along ξ_t to actively verify the safety of the robot swept volume against unknown obstacles, and update the current map M_t to M_{t+1}. (3) Subgoal Policy $\pi_g(b_t, p) \to g_t$: selects or refines the grasp configuration g_t, if necessary, given the current estimation of b_t.

IV. Method

Our system implements the three policies defined in Section III through a receding-horizon framework (Figure 2). The robot maintains a belief state $b_t = (q_t, M_t, g_t)$ that is iteratively refined through observation and replanning. At each step: (1) the **subgoal policy** π_g updates the estimated grasp configuration g_t when needed (Sec. IV-A); (2) the **active perception policy** π_v directs gaze to observe critical regions while updating the map M_t with new observations (Sec. IV-B); (3) the **whole-body motion policy** π_r computes a collision-free trajectory ξ_t to g_t under the current map M_t (Sec. IV-C); and (4) a **localization and control** module executes the planned trajectory while tracking the robot state q_t (Sec. IV-D). Benefiting from real-time whole-body motion planning [43], the system triggers replanning to generate collision-free whole-body trajectories whenever map updates invalidate the current trajectory or subgoals are reached, enabling safe and robust task completion in dynamic, partially observed environments.

A. Subgoal Policy π_g

In an unknown environment, the grasp configuration g_t cannot be determined upfront: it depends on observed geometry, collision constraints, and robot reachability that become known gradually as the map M_t is updated. Moreover, an open-loop, single-shot approach may fail when the target is physically occluded or beyond the arm's reach from the current base position. The subgoal policy π_g addresses this by decomposing the task into three progressively more exploratory steps (Figure 3): direct grasping, base repositioning, and observation gathering. It also recovers from any runtime failures by transitioning among these steps. It ensures that the

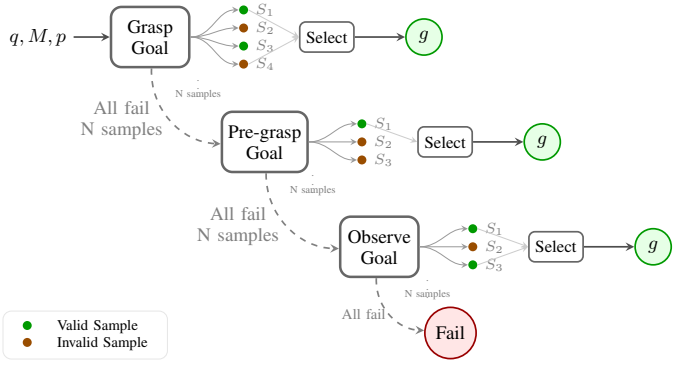

Fig. 3. **Subgoal Generation.** The process hierarchically attempts to generate a reachable subgoal (g) using three strategies: Grasp, Pre-grasp, and Observe. In each stage, N candidates ($S_1 \dots S_N$) are sampled. If all samples in a stage are invalid, the system falls back to the next strategy.

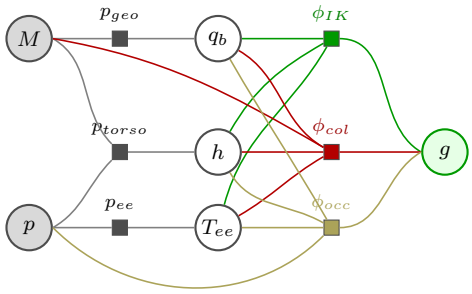

Fig. 4. **Constrained sampling for pre-grasp configuration.** Given map M and target p, we sample base pose q_b, torso height h, and end-effector pose T_{ee} via proposal distributions (black). Constraint checks enforce IK feasibility (ϕ_{IK}, green), collision-free (ϕ_{col}, red), and occlusion-free (ϕ_{occ}, yellow) via rejection sampling to produce valid goal g.

robot executes the task progressively, even with an incomplete goal specification and unknown obstacles.

1) Grasp Goal Sampling: The policy first attempts to grasp without base motion, maximizing efficiency when the target is already within reach. To do so, we use a sample-then-verify strategy. Specifically, a learning-based grasp network [40] generates candidate end-effector poses from the observed point cloud. These candidates are filtered by a pre-computed capability map [33]. We sample N diverse poses that maximize spatial coverage around the target. Each candidate is validated for whole-body inverse kinematics [3] feasibility and collision against the current map M. The first valid configuration is selected as g_t. Complete sampling and execution details are in Appendix A.

2) Pre-grasp Goal Sampling: When direct grasping fails, the robot must resample whole-body configurations to balance visibility and manipulability. Rather than solving a computationally expensive whole-body inverse kinematics problem, we decompose the search into separate subspaces for efficiency. We structure this as a constrained sampling problem with three variables and three constraint checkers, illustrated in Figure 4. Formally, given a map M_t and a target p as inputs, we need to sample simultaneously the base pose, torso height, and end-effector pose. We sample candidate configurations of them through three distributions: (1) p_{geo} samples collision-free base poses q_b from a 2D signed distance field that is updated in real-time from the point cloud; (2) p_{torso} queries a pre-built torso height map to select h that maximizes the end-effector's reach to target p given the sampled base pose; (3) p_{ee} samples end-effector poses T_{ee} in a sphere around the target, filtered by the capability map. Each sampled whole-body configuration (q_b, h, T_{ee}) is then validated through three constraint checkers via rejection sampling: inverse kinematics feasibility, collision checking, and self-occlusion checking. The complete sampling algorithm and pre-built map construction are detailed in Appendix B and Appendix C.

3) Observe Goal Sampling: When manipulation is blocked by occlusion or unknown geometry, the policy falls back to gathering observations. This improves the belief state by updating map M_t with additional sensor data. We sample N base poses at a fixed standoff distance (i.e., a safe observation distance from the target) that maintain visibility of the target p while minimizing displacement from the current configuration q_t, paired with a robust arm configuration. If all three strategies fail, the policy signals failure to the behavior tree, triggering task termination.

This hierarchical design ensures the robot either executes manipulation when feasible or improves its understanding of the environment through data gathering when blocked, maintaining progress in the receding-horizon framework.

B. Active Perception Policy π_v

A robot with a limited field of view must select its view to balance two competing demands: observing the target p for stable grasping, and monitoring the swept volume $V(\xi_t)$ for collision-free safety. Empirically, they are temporally separable, i.e., target observation matters during planning, while swept-volume monitoring matters during execution.

1) State-Dependent Gaze Optimization: We formulate the view selection, i.e., gaze control, as maximizing visibility of an importance field $\Phi(x)$ that adapts to the robot's phase:

$$a_v^* = \underset{a_v \in \mathcal{A}_v}{\operatorname{argmax}} \sum_{x \in \mathcal{W}} \Phi(x) \cdot \mathcal{V}(x, a_v) \tag{1}$$

where $\mathcal{V}(x, a_v) \in \{0, 1\}$ indicates visibility under gaze a_v. The importance field switches based on whether a trajectory ξ_t is being executed. During execution of ξ_t, the gaze is optimized to ensure trajectory safety:

$$\Phi(x) = \mathbb{I}(x \in V(\xi_t)) \cdot w_{safety}(x) \tag{2}$$

where $w_{safety}(x)$ is applied to prioritize collision-critical regions (Figure 5). It focuses more on 1) near-future areas; and 2) high-velocity areas:

$$w_{safety}(x) = \sum_{i=0}^{N} \gamma^i \cdot w_d(x, q^{(i)}) \cdot \|\dot{q}^{(i)}\| \tag{3}$$

where i indexes a waypoint in ξ_t, γ^i decays with lookahead, w_d is a distance-based Gaussian, and $\|\dot{q}^{(i)}\|$ is the velocity

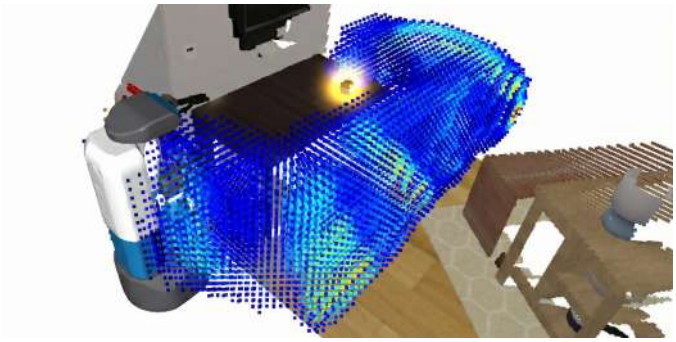

Fig. 5. **State-Dependent Gaze Optimization.** During planning ($\xi_t = \emptyset$), the importance field concentrates on the target region to enable grasp pose generation. During execution ($\xi_t \neq \emptyset$), the field shifts to the swept volume $V(\xi_t)$ (color-coded heatmap), weighted by velocity and temporal proximity to prioritize observation of collision-critical regions.

at waypoint i. Otherwise, the gaze is selected to observe the target area:

$$\Phi(x) = \mathbb{I}(x \in \mathcal{B}_\epsilon(p)) \tag{4}$$

where $\mathcal{B}_\epsilon(p)$ is a ball around the target to support grasp generation. The formulation extends [14] with velocity and manipulability. Detailed definitions and hyperparameters are provided in Appendix D.

2) Dynamic Environment Mapping: We maintain the map M_t as a point cloud updated via ray casting: new observations add perceived surface points as obstacles while removing previously occupied space along each ray. This enables the robot to detect when dynamic obstacles have moved, while preserving occluded regions. The update pipeline includes ground plane removal, robot self-filtering, and voxel-based point merging (Appendix E).

C. Whole-Body Motion Policy π_r

Reactive execution in dynamic environments requires replanning whole-body trajectories as the map M_t updates. The motion policy π_r must therefore be fast enough to replan within the control loop while producing smooth, collision-free paths. Existing approaches face a trade-off: vectorized planners like VAMP [43] achieve real-time performance but assume homogeneous configuration spaces, while state-of-the-art planners for non-holonomic mobile manipulators [50] handle heterogeneous base-arm spaces but require more computation per planning cycle.

We combine the strengths of both approaches to enable real-time whole-body motion planning. The key challenge is extending vectorized collision checking to mobile manipulators, whose base may require non-holonomic steering while the arm uses standard interpolation. Specifically, we decompose planning hierarchically directly following [50]: the algorithm first samples candidate base placements using a non-holonomic steering kernel that combines Hybrid A* [26] for global guidance with Reeds-Shepp curves for local connections. For each base placement, arm configurations are connected using standard RRT-Connect [25]. This decomposition reduces the effective search dimensionality while respecting the non-holonomic constraints of the mobile base. To enable vectorization across this heterogeneous space, we discretize base and arm motions separately, where we apply Reeds-Shepp curves for the base, linear interpolation for the arm, and then synchronize them into whole-body configurations at matching time steps. These synchronized configurations are further validated in parallel using SIMD operations [43], benefiting from the acceleration of vectorization while respecting the distinct kinematics of the base and the arm.

The raw paths from RRT-Connect typically contain redundant waypoints and exhibit jerky motion profiles. We refine paths through an iterative pipeline: randomized shortcutting [16, 18] removes unnecessary waypoints, B-spline fitting [37] smooths the trajectory, and dense re-interpolation ensures collision-free execution. This produces trajectories suitable for the MPC controller described in Section IV-D.

In our system, this implementation typically achieves planning times of 50–80 ms, enabling responsive replanning as new obstacles are observed. When planning fails or exceeds the time limit, the system halts execution. The subgoal policy triggers to generate a new goal for the next planning.

D. Localization and Control

The execution layer bridges planning and physical robot motion through localization and trajectory tracking.

1) Localization: In simulation, we directly use ground truth poses provided by the simulator. On the real robot, we employ Adaptive Monte Carlo Localization (AMCL) [30], which is a probabilistic localization method that uses a particle filter to track the robot's base pose against a pre-built 2D occupancy map. This decouples localization from dynamic obstacle handling: AMCL provides stable pose estimates using static structure (walls, furniture), while the perception module maintains a separate 3D map M_t for collision avoidance.

2) Trajectory Tracking: We employ Model Predictive Control (MPC) to convert planned trajectories into velocity commands. Given the current 11-DoF whole-body configuration $q_t \in \mathcal{C}$ (base pose, torso height, and arm joints) and the reference trajectory ξ_t, the MPC solves a finite-horizon optimization to produce a 10-dimensional velocity command: 2D for the mobile base (linear and angular velocity), 1D for the torso, and 7D for the arm joints. This formulation naturally handles whole-body coordination, generating smooth velocity profiles that synchronize base, torso, and arm motion, which is critical for mobile manipulation where uncoordinated movement causes end-effector deviation. The controller runs at 20 Hz and seamlessly accepts updated trajectories during execution, enabling reactive replanning. See Appendix F for the detailed formulation.

V. Experiments

We evaluated our system across 400 randomized simulation scenarios and 40 real-world settings at 5 indoor locations. These experiments focused on the challenge of mobile grasping under partial observability, requiring the robot to

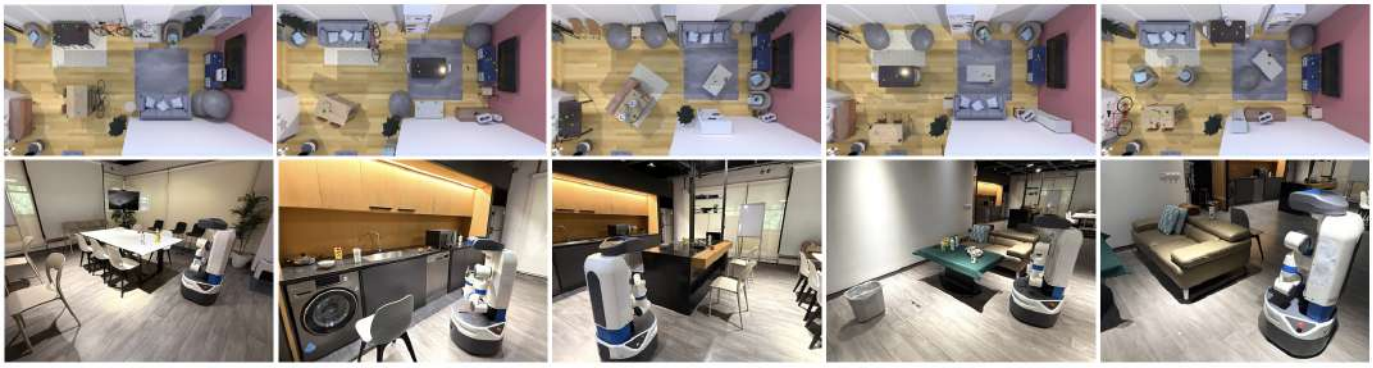

Fig. 6. **Evaluation Environments.** *Top:* Five of 20 simulation scenes (ReplicaCAD) with varied furniture layouts and target placements. *Bottom:* Real-world deployment across five indoor locations—dining table, kitchen counter, workstation, coffee table, and sofa. Complete test cases and results are provided in Appendix I and Appendix K.

integrate whole-body motion planning and active perception with unknown and dynamic obstacles. In the experiments, we mainly compared to the prevailing decoupled navigation-and-manipulation methods [38, 46]. Our results revealed the main findings of: 1) prevailing decoupled approaches suffer from frequent reachability failures and collisions due to severe partial observability; 2) velocity-aware active perception improves performance mainly by reducing collision rates; 3) high-level policy π_g is crucial for success because it provides priors to trade off among task progress, runtime failure recovery, and information gathering.

A. Experimental Setup

a) Robot Setup: For simulation experiments, we used ManiSkill3 [42] with ReplicaCAD scenes [41] for photorealistic rendering and GPU-accelerated physics. The simulator modeled a Fetch mobile manipulator with accurate kinematics and dynamics. For real-world experiments, we used a Fetch mobile manipulator with ROS. Implementation details are provided in Appendix G.

b) Evaluation Setup: We evaluated across three complementary scenarios (Figure 6): *Unknown static environments* tested perception and planning under visibility constraints with previously unseen layouts. *Dynamic environments* evaluated robustness to suddenly appearing obstacles. *Real robot deployment* validated real-world performance with localization and model errors. All scenarios guaranteed collision-free paths existed and operated at room scale ($\sim$5-10m navigation range).

Simulation Scenarios We generated 400 test scenarios (20 scenes $\times$ 20 objects) with procedural YCB object placement following [57]. Each placement underwent validation: A drop-check verified object stability. An oracle grasp-check validated grasp feasibility using physics simulation following ACRONYM [13]. For dynamic scenarios, obstacles appeared suddenly when the robot approached within a proximity, testing perception updates and replanning. Detailed benchmark generation procedures are provided in Appendix H.

Real Robot Scenarios We tested at five representative indoor locations (dining table, kitchen counter, workstation, coffee table, sofa) with 40 configurations total: 20 in static unknown and 20 in dynamic scenarios with movable objects.

c) Evaluation Metrics: We evaluated system performance using grasp success rate as the primary metric. A trial succeeded if the robot lifted the target object above a height threshold (10 cm) and maintained a stable grasp, which means the object remained stable in the gripper for more than 2 seconds. We categorized failures into three groups: *execution failures* (collision, grasp, IK) captured errors during physical interaction, *system limitations* (reachability, perception) reflected hardware and sensing constraints, and *planning failures* indicated motion planning bottlenecks. Detailed metric definitions are provided in Appendix I.

d) Baselines: We compared against three baselines representing different mobile manipulation paradigms:

Navigation-and-manipulation This is the dominant paradigm in mobile manipulation [38, 46], where navigation and arm motion planning are decoupled into separate phases. We implemented this baseline with careful adaptation to our setting: the robot used current depth observations for collision avoidance, executed closed-loop navigation with replanning upon collisions, and enabled target pose resampling. Navigation terminated when the robot reached a fixed distance from the target, after which arm motion planning began.

Direct Grasping This baseline attempted manipulation without base repositioning. It retained whole-body planning with replanning capability but executed a fixed sequence (observe, navigate, grasp) without adaptive goal resampling or gaze control. It ablated the benefit from the subgoal policy π_g, which provides priors for efficient task planning.

Velocity-agnostic This baseline retained all system components except velocity-awareness Eq. 3 in gaze optimization. It used only volume-based safety weights [14]. This ablated our velocity-aware active perception module to explore its effects on task performance and safety.

Additional experiments including system-level comparisons, statistical analysis, and limitation analysis are provided in Appendix K.

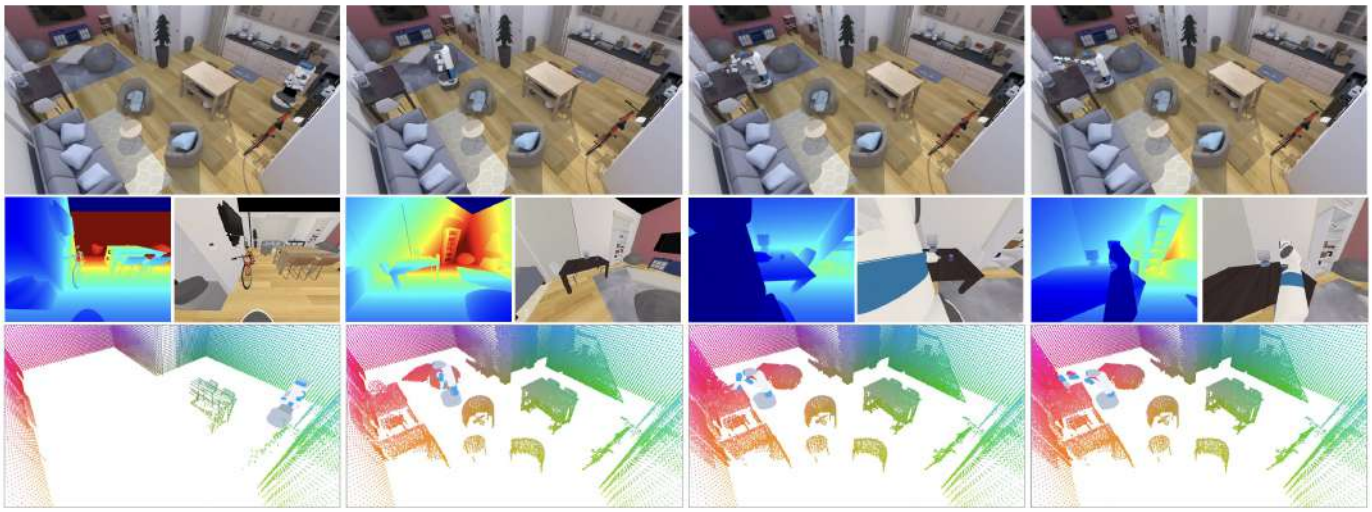

Fig. 7. Qualitative results showing the execution sequence in simulation. From left to right: starting pose, observation stage, pre-grasp stage, and grasping stage. The top row shows the third-person view, the middle row shows the first-person view from the robot head camera, and the bottom row displays the robot's belief map.

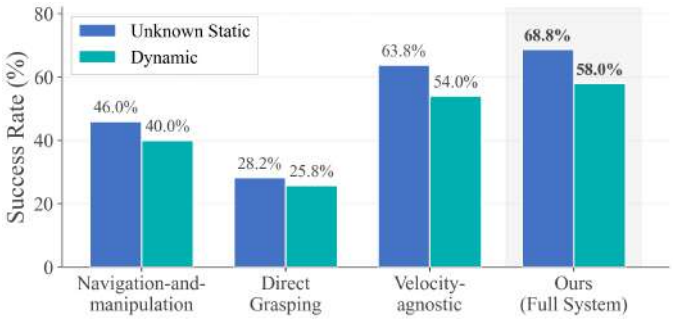

Fig. 8. **Success Rate Comparison.** Ablation study results across unknown static and dynamic simulation environments. Our full system achieves the highest success rates (68.8% and 58.0%), with subgoal generation providing the largest performance gain. Detailed failure breakdowns are provided in Appendix J.

Fig. 9. **Spatial Distribution of Success and Failure.** Green and red circles represent successful and failed grasping attempts, respectively.

B. Simulation Results

We showcase a rollout of our mobile manipulation system in Figure 7. Figure 8 and Figure 9 present quantitative results for unknown static and dynamic environments and failure distributions. Detailed failure breakdowns are in Appendix J.

a) Qualitative Results: Figure 7 demonstrates the system's runtime behavior. As the robot executed the task, the belief map was incrementally built and updated, ensuring a current representation of the environment. Simultaneously, the camera perception and motion planning modules continuously updated to adapt to the evolving scene. A key feature of our design was observed in the pre-grasp stage, where the selected pose effectively avoided self-occlusion, ensuring the target object remained visible for the final grasping action.

b) Integrated vs. Navigation-and-manipulation: Our integrated approach outperformed navigation-and-manipulation by +22.8% (unknown) and +18.0% (dynamic). Decoupling navigation from manipulation created two fundamental limitations. First, in unknown environments, determining an optimal base placement required understanding the target's surroundings, yet this information was unavailable until the robot approached and observed. The navigation-and-manipulation approach committed to a base position before acquiring sufficient environmental knowledge, frequently resulting in configurations where the target was unreachable, leading to higher planning failures (20.5% vs. 8.8%). Second, in constrained spaces, whole-body motion planning provided greater flexibility by coordinating base and arm movements simultaneously along the trajectory. The navigation-and-manipulation approach restricted manipulation to arm-only motions from a fixed base, which struggled to find and track collision-free paths in cluttered environments, resulting in higher collision rates (17.5% vs. 6.3%).

c) Integrated vs. Direct Grasping: Removing the high-level policy caused the largest performance drop: -40.5% (unknown) and -32.3% (dynamic), with planning failures increasing 4.5–6×. Without hierarchical guidance, the system attempted direct goal-reaching in a fixed subgoal sequence.

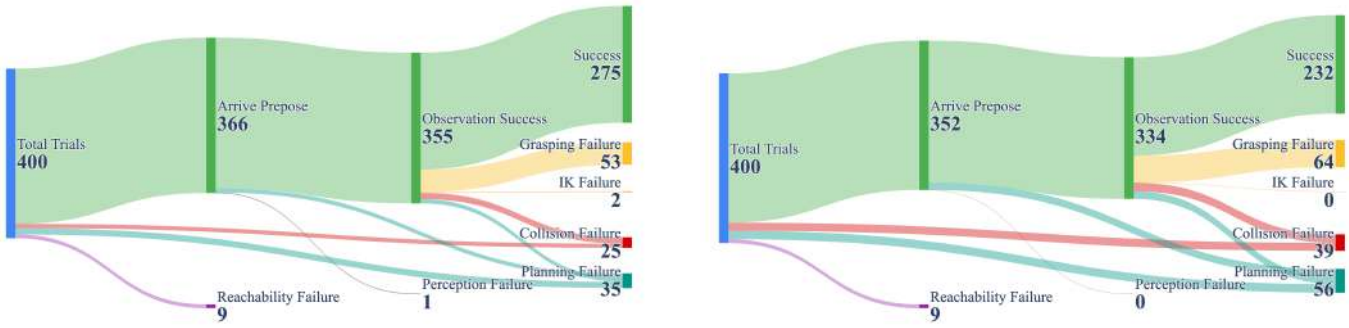

Fig. 10. Failure flow in unknown static (left) and dynamic (right) environments in simulation benchmark.

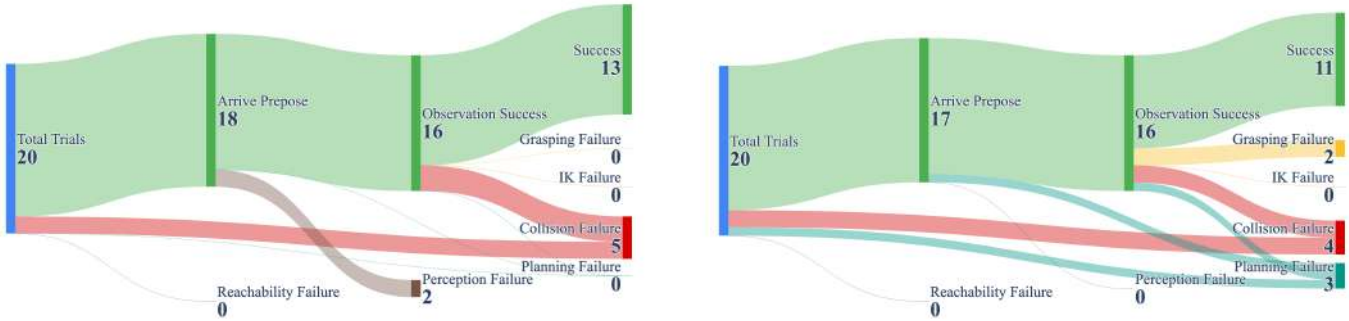

Fig. 11. Real robot evaluation results. Process flow and failure breakdown for static unknown (left) and dynamic (right) environments.

d) Velocity-Aware Active Perception: Velocity awareness improved robustness in dynamic environments. When obstacles moved, the robot prioritized observing its faster-moving parts to reduce the risk of collision. Without this, collision failures increased significantly to 13.8% from 9.8%.

e) Failure Analysis: Figure 10 shows the failure distribution on our benchmark. The primary bottleneck was grasp detection: the network lacked kinematic awareness and often selected unreachable poses, with accuracy degrading for out-of-distribution objects. Execution errors arose from accumulated tracking drift over long trajectories, occasionally causing collisions. Planning failures occurred in inherently challenging scenarios such as wall corners, deep table regions, and cluttered areas. Detailed analysis is in Appendix J.

C. Real-World Results

We validated our system on a real Fetch robot across five indoor locations with 40 test configurations (20 static, 20 dynamic). Figure 11 shows results for our full system only. As shown in Figure 11, the system achieved comparable success rates to simulation in both static (65.0% vs. 68.8%) and dynamic (55.0% vs. 58.0%) environments.

Planning failures primarily came from noisy observations (e.g., floating points in the point cloud) caused by data synchronization issues between the localization system and depth sensors. Collision failures arose from two sources: ground plane filtering that occasionally removed points from low-lying objects like chair legs, and higher execution noise where the robot got stuck due to friction, then suddenly accelerated, disrupting pose estimation. Perception failures occurred when segmentation or detection errors led to incorrect grasping. These factors accounted for the performance gap between simulation and real-world.

VI. Conclusion

This paper presented a unified framework for mobile grasping in dynamic, unknown environments under visibility constraints. By integrating velocity-aware active perception, hierarchical subgoal generation, and fast whole-body replanning, our system improves success rates by 22.8% and 18.0% over the navigation-and-manipulation method and achieves an overall 68.8% success rate in unknown static environments and 58.0% in dynamic environments. In particular, our method reduces collision rates effectively compared to baselines, benefiting from our closed-loop planning and active perception. These results demonstrate the importance of well-coordinated perception and planning for executing mobile grasping tasks in large-scale dynamic environments. Real-world experiments demonstrated strong generalization across diverse indoor settings. Future work includes incorporating predictive models for human and object motion to reduce reactive replanning, extending to multi-fingered hands for broader grasp coverage, and developing probabilistic completeness guarantees for visibility-constrained planning. The modular architecture naturally extends to multi-robot coordination and contact-rich manipulation tasks.

References

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Raefer Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding

language in robotic affordances. In *Conference on Robot Learning*, pages 287–315. PMLR, 2022. URL https://arxiv.org/abs/2204.01691.

[2] Max Bajracharya, James Borders, Richard Cheng, Dan Helmick, Lukas Kaul, Dan Kruse, John Leichty, Jeremy Ma, Carolyn Matl, Frank Michel, Chavdar Papazov, Josh Petersen, Krishna Shankar, and Mark Tjersland. Demonstrating mobile manipulation in the wild: A metrics-driven approach. In *Robotics: Science and Systems XIX*, RSS2023. Robotics: Science and Systems Foundation, July 2023. doi: 10.15607/rss.2023.xix.055. URL http://dx.doi.org/10.15607/RSS.2023.XIX.055.

[3] Patrick Beeson and Barrett Ames. Trac-ik: An open-source library for improved solving of generic inverse kinematics. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, page 928–935. IEEE Press, 2015. doi: 10.1109/HUMANOIDS.2015.7363472. URL https://doi.org/10.1109/HUMANOIDS.2015.7363472.

[4] Kostas E. Bekris and Lydia E. Kavraki. Greedy but safe replanning under kinodynamic constraints. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 704–710, 2007. doi: 10.1109/ROBOT.2007.363069.

[5] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. Receding horizon "next-best-view" planner for 3d exploration. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1462–1468, 2016. doi: 10.1109/ICRA.2016.7487281.

[6] Kenneth Blomqvist, Michel Breyer, Andrei Cramariuc, Julian Förster, Margarita Grinvald, Florian Tschopp, Jen Jen Chung, Lionel Ott, Juan Nieto, and Roland Siegwart. Go fetch: Mobile manipulation in unstructured environments, 2020. URL https://arxiv.org/abs/2004.00899.

[7] Michel Breyer, Lionel Ott, Roland Siegwart, and Jen Jen Chung. Closed-loop next-best-view planning for target-driven grasping, 2022. URL https://arxiv.org/abs/2207.10543.

[8] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. URL https://arxiv.org/abs/2212.06817.

[9] Dong Chen and Georg von Wichert. Uncertainty-aware arm-base coordinated object grasping with a mobile manipulation platform. In *ISR/Robotik 2014; 41st International Symposium on Robotics*, pages 1–6, 2014.

[10] Nermin Covic, Bakir Lacevic, Dinko Osmankovic, and Tarik Uzunovic. Real-time sampling-based safe motion planning for robotic manipulators in dynamic environments. *IEEE Transactions on Robotics*, 41:5287–5306, 2025. ISSN 1941-0468. doi: 10.1109/tro.2025.3598119. URL http://dx.doi.org/10.1109/TRO.2025.3598119.

[11] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models, 2024. URL https://arxiv.org/abs/2409.17146.

[12] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13665–13675, 2023. URL https://arxiv.org/abs/2303.03378.

[13] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. Acronym: A large-scale grasp dataset based on simulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6222–6227, 2021. doi: 10.1109/ICRA48506.2021.9560844.

[14] Mark Finean, Wolfgang Merkt, and Ioannis Havoutis. Where should i look optimised gaze control for whole-body collision avoidance in dynamic environments. *IEEE Robotics and Automation Letters*, PP:1–1, 12 2021. doi: 10.1109/LRA.2021.3137545.

[15] Mark Nicholas Finean, Wolfgang Merkt, and Ioannis Havoutis. Simultaneous scene reconstruction and whole-body motion planning for safe operation in dynamic environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3710–3717. IEEE, 2021. doi: 10.1109/iros51168.2021.9636860. URL http://dx.doi.org/10.1109/IROS51168.2021.9636860.

[16] Roland Geraerts and Mark H. Overmars. Creating high-quality roadmaps for motion planning in virtual environments. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4355–4361, 2006. doi: 10.1109/IROS.2006.282009.

[17] Gustavo Goretkin, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Look before you sweep: Visibility-aware motion planning. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics (WAFR)*, pages 1–16. Springer, 2018. URL https://arxiv.org/abs/1901.06109.

[18] Kris Hauser and Victor Ng-Thow-Hing. Fast smoothing of manipulator trajectories using optimal bounded-acceleration shortcuts. In *2010 IEEE International Conference on Robotics and Automation*, pages 2493–2498,

2010. doi: 10.1109/ROBOT.2010.5509683.

[19] Jesse Haviland and Peter Corke. Neo: A novel expeditious optimisation algorithm for reactive motion control of manipulators. *IEEE Robotics and Automation Letters*, 6(2):1043–1050, April 2021. ISSN 2377-3774. doi: 10.1109/lra.2021.3056060. URL http://dx.doi.org/10.1109/LRA.2021.3056060.

[20] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. In *Conference on Robot Learning*, pages 540–562. PMLR, 2023. URL https://arxiv.org/abs/2307.05973.

[21] Brian Ichter, Benoit Landry, Edward Schmerling, and Marco Pavone. Perception-aware motion planning via multiobjective search on gpus, 2017. URL https://arxiv.org/abs/1705.02408.

[22] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[23] Snehal Jauhri, Jan Peters, and Georgia Chalvatzaki. Robot learning of mobile manipulation with reachability behavior priors. *IEEE Robotics and Automation Letters*, 7(3):8399–8406, 2022. doi: 10.1109/LRA.2022.3188109. URL https://doi.org/10.1109/LRA.2022.3188109.

[24] Snehal Jauhri, Sophie Lueth, and Georgia Chalvatzaki. Active-perceptive motion generation for mobile manipulation, 2024. URL https://arxiv.org/abs/2310.00433.

[25] J.J. Kuffner and S.M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 2, pages 995–1001 vol.2, 2000. doi: 10.1109/ROBOT.2000.844730.

[26] Karl Kurzer. Path planning in unstructured environments : A real-time hybrid a* implementation for fast and deterministic path generation for the kth research concept vehicle. Master's thesis, KTH, Integrated Transport Research Lab, ITRL, 2016. URL https://www.diva-portal.org/smash/record.jsf?pid=diva2:1057261.

[27] Peiqi Liu, Yaswanth Orru, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Ok-robot: What really matters in integrating open-knowledge models for robotics. *arXiv preprint arXiv:2401.12202*, 2024. URL https://arxiv.org/abs/2401.12202.

[28] Peiqi Liu, Zhanqiu Guo, Mohit Warke, Soumith Chintala, Chris Paxton, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Dynamem: Online dynamic spatio-semantic memory for open world mobile manipulation, 2025. URL https://arxiv.org/abs/2411.04999.

[29] Yunfan Lu, Yuchen Ma, David Hsu, and Panpan Cai. Neural randomized planning for whole body robot motion, 2024. URL https://arxiv.org/abs/2405.11317.

[30] Steven Macenski, Francisco Martin, Ruffin White, and Jonatan Ginés Clavero. The marathon 2: A navigation system. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. doi: 10.1109/IROS45743.2020.9341207. URL https://doi.org/10.1109/IROS45743.2020.9341207.

[31] Joao Marcos Correia Marques, Nils Dengler, Tobias Zaenker, Jesper Mucke, Shenlong Wang, Maren Bennewitz, and Kris Hauser. Map space belief prediction for manipulation-enhanced mapping, 2025. URL https://arxiv.org/abs/2502.20606.

[32] Qingxi Meng, Emiliano Flores, Carlos Quintero-Pe na, Peizhu Qian, Zachary Kingston, Shannan K. Hamlin, Vaibhav Unhelkar, and Lydia E. Kavraki. Look as you leap: Planning simultaneous motion and perception for high-dof robots, 2025. URL https://arxiv.org/abs/2509.19610.

[33] Yuhao Meng, Yujing Chen, and Yunjiang Lou. Uncertainty aware mobile manipulator platform pose planning based on capability map. In *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 123–128, 2021. doi: 10.1109/RCAR52367.2021.9517662.

[34] Douglas Morrison, Peter Corke, and Jürgen Leitner. Multi-view picking: Next-best-view reaching for improved grasping in clutter. In *2019 International Conference on Robotics and Automation (ICRA)*, page 8762–8768. IEEE Press, 2019. doi: 10.1109/ICRA.2019.8793805. URL https://doi.org/10.1109/ICRA.2019.8793805.

[35] Mustafa Mukadam, Jing Dong, Xinyan Yan, Frank Dellaert, and Byron Boots. Continuous-time gaussian process motion planning via probabilistic inference. *The International Journal of Robotics Research*, 37(11):1319–1340, September 2018. ISSN 1741-3176. doi: 10.1177/0278364918790369. URL http://dx.doi.org/10.1177/0278364918790369.

[36] Lakshadeep Naik, Sinan Kalkan, and Norbert Krüger. Pre-grasp approaching on mobile robots: A pre-active layered approach. *IEEE Robotics and Automation Letters*, 9(3):2606–2613, 2024. doi: 10.1109/LRA.2024.3358077.

[37] Jia Pan, Liangjun Zhang, and Dinesh Manocha. Collision-free and smooth trajectory computation in cluttered environments. *The International Journal of Robotics Research*, 31(10):1155–1175, 2012. doi: 10.1177/0278364912453186. URL https://doi.org/10.1177/0278364912453186.

[38] Fabian Reister, Markus Grotz, and Tamim Asfour. Combining navigation and manipulation costs for time-

efficient robot placement in mobile manipulation tasks. *IEEE Robotics and Automation Letters*, 7(4):9913–9920, 2022. doi: 10.1109/LRA.2022.3191215.

[39] Max Spahn, Corrado Pezzato, Chadi Salmi, Rick Dekker, Cong Wang, Christian Pek, Jens Kober, Javier Alonso-Mora, Carlos Hernández Corbato, and Martijn Wisse. Demonstrating adaptive mobile manipulation in retail environments. In *Robotics: Science and Systems (R:SS)*, 2024. doi: 10.15607/RSS.2024.XX.047. URL https://www.roboticsproceedings.org/rss20/p047.html.

[40] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, page 13438–13444. IEEE Press, 2021. doi: 10.1109/ICRA48506.2021.9561877. URL https://doi.org/10.1109/ICRA48506.2021.9561877.

[41] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/021bbc7ee20b71134d53e20206bd6feb-Abstract.html.

[42] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025. URL https://arxiv.org/abs/2410.00425.

[43] Wil Thomason, Zachary Kingston, and Lydia E. Kavraki. Motions in microseconds via vectorized sampling-based planning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8749–8756, 2024. doi: 10.1109/ICRA57147.2024.10611190.

[44] Ruisen Tu, Arth Shukla, Sohyun Yoo, Xuanlin Li, Junxi Li, Jianwen Xie, Hao Su, and Zhuowen Tu. Sg-vla: Learning spatially-grounded vision-language-action models for mobile manipulation, 2026. URL https://arxiv.org/abs/2603.22760.

[45] Mariliza Tzes, Vasileios Vasilopoulos, Yiannis Kantaros, and George J. Pappas. Reactive informative planning for mobile manipulation tasks under sensing and environmental uncertainty. In *2022 International Conference on Robotics and Automation (ICRA)*, page 7320–7326. IEEE Press, 2022. doi: 10.1109/ICRA46639.2022.9811642. URL https://doi.org/10.1109/ICRA46639.2022.9811642.

[46] Nikolaus Vahrenkamp, Tamim Asfour, and Rüdiger Dillmann. Robot placement based on reachability inversion. In *2013 IEEE International Conference on Robotics and Automation*, pages 1970–1975, 2013. doi: 10.1109/ICRA.2013.6630839.

[47] Vasileios Vasilopoulos, Yiannis Kantaros, George J. Pappas, and Daniel E. Koditschek. Reactive planning for mobile manipulation tasks in unexplored semantic environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6385–6392, 2021. doi: 10.1109/ICRA48506.2021.9561958.

[48] Jilong Wang, Javokhirbek Rajabov, Chaoyi Xu, Yiming Zheng, and He Wang. Quadwbg: Generalizable quadrupedal whole-body grasping, 2025. URL https://arxiv.org/abs/2411.06782.

[49] Tyler S Wilson, Wil Thomason, Zachary Kingston, Lydia E Kavraki, and Jonathan D Gammell. Nearest-neighbourless asymptotically optimal motion planning with Fully Connected Informed Trees (FCIT$*$). In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 14140–14146, Atlanta, GA, USA, 19–23 May 2025. doi: 10.1109/ICRA55743.2025.11127785.

[50] Chengkai Wu, Ruilin Wang, Mianzhi Song, Fei Gao, Jie Mei, and Boyu Zhou. Real-time whole-body motion planning for mobile manipulators using environment-adaptive search and spatial-temporal optimization. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1369–1375, 2024. doi: 10.1109/ICRA57147.2024.10610192.

[51] Zhenyu Wu, Yuheng Zhou, Xiuwei Xu, Ziwei Wang, and Haibin Yan. Momanipvla: Transferring vision-language-action models for general mobile manipulation, 2025. URL https://arxiv.org/abs/2503.13446.

[52] Anxing Xiao, Nuwan Janaka, Tianrun Hu, Anshul Gupta, Kaixin Li, Cunjun Yu, and David Hsu. Robi butler: Multimodal remote interaction with a household robot assistant, 2025. URL https://arxiv.org/abs/2409.20548.

[53] Yuqiang Yang, Fei Meng, Zehui Meng, and Chenguang Yang. Rampage: Toward whole-body, real-time, and agile motion planning in unknown cluttered environments for mobile manipulators. *IEEE Transactions on Industrial Electronics*, 71(11):14492–14502, 2024. doi: 10.1109/TIE.2024.3370969.

[54] Sriram Yenamandra, Arun Ramachandran, Karmesh Yadav, Austin Wang, Mukul Khanna, Theophile Gervet, Tsung-Yen Yang, Vidhi Jain, Alexander William Clegg, John Turner, Zsolt Kira, Manolis Savva, Angel Chang, Devendra Singh Chaplot, Dhruv Batra, Roozbeh Mottaghi, Yonatan Bisk, and Chris Paxton. Homerobot: Open-vocabulary mobile manipulation, 2024. URL https://arxiv.org/abs/2306.11565.

[55] Naoki Yokoyama, Alex Clegg, Joanne Truong, Eric Undersander, Tsung-Yen Yang, Sergio Arnaud, Sehoon Ha, Dhruv Batra, and Akshara Rai. Asc: Adaptive skill coordination for robotic mobile manipulation, 2023. URL

https://arxiv.org/abs/2304.00410.

[56] Huiwen Zhang, Kai Mi, and Zhijun Zhang. Base placement optimization for coverage mobile manipulation tasks, 2023. URL https://arxiv.org/abs/2304.08246.

[57] Jiazhao Zhang, Nandiraju Gireesh, Jilong Wang, Xiaomeng Fang, Chaoyi Xu, Weiguang Chen, Liu Dai, and He Wang. Gamma: Graspability-aware mobile manipulation policy learning based on online grasping pose fusion, 2024. URL https://arxiv.org/abs/2309.15459.

[58] Xiaohan Zhang, Yifeng Zhu, Yan Ding, Yuqian Jiang, Yuke Zhu, Peter Stone, and Shiqi Zhang. Symbolic state space optimization for long horizon mobile manipulation planning. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 866–872, 2023. doi: 10.1109/IROS55552.2023.10342224.

[59] Xuechao Zhang, Dong Wang, Sun Han, Weichuang Li, Bin Zhao, Zhigang Wang, Xiaoming Duan, Chongrong Fang, Xuelong Li, and Jianping He. Affordance-driven next-best-view planning for robotic grasping, 2023. URL https://arxiv.org/abs/2309.09556.

[60] Zihang Zhao, Leiyao Cui, Sirui Xie, Saiyao Zhang, Zhi Han, Lecheng Ruan, and Yixin Zhu. B*: Efficient and optimal base placement for fixed-base manipulators. *IEEE Robotics and Automation Letters*, 10(10):10634–10641, October 2025. ISSN 2377-3774. doi: 10.1109/lra.2025.3604741. URL http://dx.doi.org/10.1109/LRA.2025.3604741.

[61] Brianna Zitkovich, Anthony Apple, Denys Bodnar, Thanh Nguyen, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, et al. Rt-2: Vision-language-action models transferred from web-scale real-world data. *arXiv preprint arXiv:2307.15818*, 2023. URL https://arxiv.org/abs/2307.15818.

[62] Claudio Zito, Valerio Ortenzi, Maxime Adjigble, Marek Kopicki, Rustam Stolkin, and Jeremy L. Wyatt. Hypothesis-based belief planning for dexterous grasping, 2019. URL https://arxiv.org/abs/1903.05517.

APPENDIX

This appendix provides supplementary material organized as follows: implementation details for grasp sampling, pre-grasp configuration generation, kinematic map construction, active perception, environment mapping, and path tracking (Sections A–F); real robot deployment specifics (Section G); benchmark generation and evaluation protocol (Sections H and I); detailed failure analysis (Section J); and additional experiments including system-level comparisons, path efficiency, obstacle distance sensitivity, and sampling completeness (Section K).

A. Grasp Goal Sampling and Execution

Algorithm 1 details the grasp goal sampling procedure for in-place manipulation without base motion.

Algorithm 1 Grasp Goal Sampling

Require: Point cloud P, target p, current config q_{curr}, capability map $\mathcal{M}_{cap}$
Ensure: Valid grasp configuration g or failure

$\mathcal{G}_{raw} \leftarrow$ CONTACTGRASPNET(P, p) ▷ Generate grasp candidates
$\mathcal{G}_{feas} \leftarrow \{T_{ee} \in \mathcal{G}_{raw} : \mathcal{M}_{cap}(T_{ee}) > 0\}$ ▷ Filter by capability
$\mathcal{G}_{div} \leftarrow$ FARTHESTPOINTSAMPLE$(\mathcal{G}_{feas}, N)$ ▷ Select diverse subset
Sort $\mathcal{G}_{div}$ by $\hat{s}(T_{ee})$ descending ▷ Rank by proximity-weighted score
for $T_{ee} \in \mathcal{G}_{div}$ **do**
  $q_{arm} \leftarrow \text{IK}(T_{ee}, q_{curr})$ ▷ Solve inverse kinematics
  if $q_{arm} =$ `None` **then continue**
  end if
  $g \leftarrow (q_{curr,base}, q_{curr,torso}, q_{arm})$
  if COLLISIONFREE(g, M) **then**
    return g
  end if
end for
return `Failure`

a) Grasp Candidate Generation: We use Contact-GraspNet [40] to generate 6-DoF grasp pose candidates from the observed point cloud. The network takes as input the segmented point cloud around the target and outputs a set of grasp poses $\{T_{ee}^{(i)}\}$ with associated confidence scores. We filter candidates to retain only those within a radius of 10 cm from the target centroid.

b) Capability Map Filtering: Each grasp candidate is transformed to the base frame and queried against the pre-computed capability map (Section C). Candidates mapping to voxels with zero visitation frequency are rejected, as they correspond to end-effector poses unreachable by the arm from the current base position.

c) Diversity Selection: From the feasible candidates, we select $N_{diverse} = 20$ poses using farthest point sampling in $SE(3)$ to maximize spatial coverage. Starting from the highest-confidence grasp, we iteratively add the candidate maximizing the minimum distance to already-selected grasps, using the metric:

$$d(T_1, T_2) = \|t_1 - t_2\| + \lambda \cdot \arccos\left(\frac{\text{tr}(R_1^T R_2) - 1}{2}\right) \quad (5)$$

where $\lambda = 0.1$ balances translation and rotation differences.

d) Proximity-Aware Ranking: After diversity selection, candidates are sorted by a proximity-weighted score that provides a weak preference for grasps closer to the current end-effector pose:

$$\hat{s}(T_{ee}) = s(T_{ee}) - \lambda_d \cdot \|t_{ee} - t_{ee,curr}\| \quad (6)$$

where $s(T_{ee})$ is the grasp confidence from Contact-GraspNet, $t_{ee,curr}$ is the current end-effector position, and $\lambda_d = 0.01$. This small weight acts primarily as a tiebreaker among similarly confident grasps, favoring configurations that require less arm motion.

e) Validation and Execution: Each selected grasp is validated by: (1) solving whole-body inverse kinematics with the current base and torso configuration fixed; (2) checking the resulting configuration for collision against the current map M_t. The first valid configuration is selected as the grasp goal g_t. Note that this greedy first-valid strategy prioritizes computational efficiency over grasp quality optimization; tighter coupling between grasp selection and motion feasibility remains a direction for future work.

During execution, even in-place arm motion uses the same receding horizon framework as whole-body motion. Algorithm 2 details this procedure: the system plans an initial arm trajectory, then executes it asynchronously while continuously monitoring for collisions. At each control cycle, it updates the collision environment from the latest depth observation, checks the remaining trajectory for collisions, and triggers replanning if obstacles are detected. The gaze optimizer runs concurrently at 10 Hz—limited by the scene management and map update pipeline rather than the optimization itself (which completes in under 5 ms)—directing the camera toward regions along the planned trajectory, weighted by temporal proximity and velocity (Section D). The robot first moves to a pre-grasp pose (5 cm offset along the approach direction), then executes a linear approach motion before closing the gripper.

B. Pre-Grasp Configuration Sampling

Algorithm 3 details the constrained sampling procedure for generating pre-grasp configurations when direct grasping fails.

a) Base Pose Sampling (p_{geo}): We maintain a 2D costmap derived from the 3D belief point cloud. The SDF generation proceeds as follows: (1) filter points by height (0.05–1.5 m) to capture obstacles at robot operating height; (2) project filtered points onto the XY ground plane; (3) rasterize into a 2D occupancy grid at 0.05 m resolution; (4) apply binary closing (3×3 kernel) to fill small gaps; (5) identify the navigable region as the largest connected free-space component; (6) compute the Euclidean distance transform from obstacle cells; (7) convert distances to costs via exponential decay $c = \exp(-d/d_{max})$ where $d_{max} = 2.0$ m. Base poses

Algorithm 2 Arm Motion with Replanning

Require: Current config q_{curr}, goal config g, map M, max replan attempts K
Ensure: Motion success or failure

$\xi \leftarrow$ PLANARM(q_{curr}, g, M) ▷ Initial plan
STARTASYNCEXECUTION(ξ)
$k \leftarrow 0$
while not MOTIONCOMPLETE() **and** $k \leq K$ **do**
  $M \leftarrow$ UPDATEMAP(depth, T_{wc}) ▷ Ray casting update
  $i \leftarrow$ CURRENTWAYPOINTINDEX()
  UPDATEGAZE(ξ, i) ▷ Look ahead at trajectory
  if CHECKCOLLISION$(\xi[i:], M)$ **then**
    STOPMOTION()
    $q_{curr} \leftarrow$ GETCURRENTCONFIG()
    $\xi \leftarrow$ PLANARM(q_{curr}, g, M) ▷ Replan
    if $\xi =$ `None` **then return** `Failure`
    end if
    STARTASYNCEXECUTION(ξ)
    $k \leftarrow k + 1$
  end if
end while
if $k > K$ **then return** `Failure`
end if
return `Success`

Algorithm 3 Pre-Grasp Configuration Sampling

Require: Map M, target p, current config q_{curr}, number of samples N
Ensure: Valid pre-grasp configuration g or failure

for $i = 1$ to N **do**
  $q_b \sim p_{geo}(q_b \mid M)$ ▷ Sample base pose from SDF
  $h \sim p_{torso}(h \mid q_b, p)$ ▷ Query torso height map
  $T_{ee} \sim p_{ee}(T_{ee} \mid p)$ ▷ Sample EE pose around target
  $q_{arm} \leftarrow \text{IK}(T_{ee}, q_b, h)$ ▷ Solve inverse kinematics
  if $q_{arm} =$ `None` **then continue**
  end if
  $g \leftarrow (q_b, h, q_{arm})$
  if COLLISIONFREE$(g, M) \wedge$ OCCLUSIONFREE(g, p) **then**
    return g
  end if
end for
return `Failure`

are sampled with probability proportional to $(1 - c)^2$, biased toward low-cost regions within manipulation radius of the target. Orientation is set to face the target.

b) Torso Height Selection (p_{torso}): We pre-compute a torso height map $H : \mathbb{R}^3 \to \mathbb{R}$ that maps end-effector position (relative to the base frame) to the optimal torso height that maximizes the end-effector's reachability. This map is computed offline by selecting the torso height that maximizes the volume of feasible joint configurations reaching each position bin (Section C).

c) End-Effector Pose Sampling (p_{ee}): Candidate end-effector poses are sampled on a sphere of radius 10 cm centered at the target p. Orientations are sampled from approach directions pointing toward p, filtered by the pre-computed capability map to ensure kinematic feasibility.

C. Pre-computed Kinematic Maps

We pre-compute three maps offline to accelerate online sampling. All maps are robot-specific and computed once.

a) Reachability Map: We construct a 6D reachability map by uniformly sampling $N = 1.28 \times 10^8$ joint configurations within joint limits and computing forward kinematics for each. End-effector poses are discretized into a 6D voxel grid (position at 5 cm resolution, orientation at $\pi/8$ rad resolution). For each voxel, we store the visitation frequency and the maximum Yoshikawa manipulability $\sqrt{\det(JJ^T)}$ across all configurations mapping to that voxel. Voxels below the ground plane are filtered out.

b) Capability Map: The capability map [33] is derived by inverting the reachability map transforms, converting from "given base pose, where can the end-effector reach?" to "given a desired end-effector pose relative to the base, is it reachable?" At runtime, we query this map to filter candidate grasp poses: voxels with zero visitation frequency are marked unreachable, allowing early rejection before expensive IK computation.

c) Torso Height Map: For robots with a prismatic torso joint (e.g., Fetch), we build a torso height policy map. We sample 3×10^8 joint configurations uniformly, compute FK, and discretize the end-effector position into 3D bins (2 cm resolution) while separately binning the torso height (5 cm resolution). For each position bin (x, y, z), we select the torso height with the highest sample count:

$$h^*(x, y, z) = \underset{h}{\text{argmax}}\, \text{Count}\big[\text{FK}(q) \in \text{bin}(x, y, z) \wedge \text{torso}(q) \in \text{bin}(h)\big] \quad (7)$$

This yields the torso height that maximizes the volume of feasible joint configurations reaching each position, providing a kinematically-informed default without online search.

D. Active Perception Details

The safety weight $w_{safety}(x)$ in the gaze optimization combines temporal, spatial, and velocity terms. The distance weight $w_d(x, q^{(i)})$ is a Gaussian function centered at moderate distance:

$$w_d(x, q^{(i)}) = \exp\left(-\frac{1}{2}\left(\frac{d(x, q^{(i)}) - d_{mid}}{\sigma}\right)^2\right) \quad (8)$$

where $d(x, q^{(i)})$ is the distance from point x to the robot at configuration $q^{(i)}$, d_{mid} is the center distance (regions too close offer no reaction time; regions too far pose less immediate threat), and $\sigma = \max(1.0, 0.25 \cdot N)$ controls the width based on the lookahead window N.

The complete safety weight is:

$$w_{safety}(x) = \sum_{i=0}^{N} \gamma^i \cdot w_d(x, q^{(i)}) \cdot \|\dot{q}^{(i)}\| \tag{9}$$

where $\gamma = 0.999$ provides temporal decay (prioritizing near-term waypoints), and $\|\dot{q}^{(i)}\|$ is the velocity magnitude at waypoint i. This formulation emphasizes regions that are (1) temporally imminent, (2) at moderate distance where corrective action is still possible, and (3) near fast-moving links where collision consequences are most severe.

The gaze action space $\mathcal{A}_v$ is discretized into a grid of pan-tilt angles for the head camera. At each control cycle, we evaluate the importance field integral for each candidate gaze direction and select the maximum. With typical discretization of 8 pan $\times$ 5 tilt angles, this optimization completes in under 5 ms.

E. Dynamic Environment Mapping

The map M_t is maintained as a point cloud that evolves through ray casting updates. Each RGB-D observation triggers the following pipeline:

a) Point Cloud Generation: The depth image is converted to a world-frame point cloud by back-projecting pixels using the camera intrinsics K and transforming via the camera-to-world pose T_{wc}. Points are filtered by depth range (0.2–3.0 m) and cropped to a 2.5 m radius sphere around the camera to focus on the relevant workspace.

b) Filtering Pipeline: Before integration, points undergo two filtering stages: (1) ground plane removal, which discards points below a height threshold ($z < 0.3$ m) to eliminate floor returns that would interfere with navigation planning; (2) robot self-filtering, which removes points that fall within the robot's own body volume using the current joint configuration to prevent self-collision artifacts in the map.

c) Ray Casting Update: The core update uses ray casting to distinguish free space from occluded regions. For each existing point $x \in M_t$, we project it into the current camera frame and compare its depth z_x against the measured depth z_{obs} at the corresponding pixel:

- If $z_x < z_{obs} - \delta$: the point lies in free space (between camera and observed surface) and is removed.
- If $z_x \geq z_{obs} - \delta$: the point is either on the surface or occluded behind it, and is preserved.

Here $\delta = 0.03$ m is a merge tolerance. This approach enables the robot to detect when dynamic obstacles have moved away while preserving geometry that remains occluded.

d) Point Merging: New observation points are merged with the existing map using voxel-based deduplication. Points are quantized to a voxel grid (resolution 0.03 m), and only points falling in previously unoccupied voxels are added. This prevents unbounded point cloud growth while maintaining surface detail.

F. Whole-Body Path Tracker

We implement a whole-body velocity controller running at 20 Hz to track planned trajectories. The whole-body state is defined as $\mathbf{x} = [x, y, \theta, h, \mathbf{q}_{arm}]^\top \in \mathbb{R}^{11}$, comprising the base pose (x, y, θ), torso height h, and 7 arm joint angles $\mathbf{q}_{arm}$. The control input is $\mathbf{u} = [v, \omega, \dot{h}, \dot{\mathbf{q}}_{arm}]^\top \in \mathbb{R}^{10}$, consisting of the base linear and angular velocities, torso velocity, and 7 arm joint velocities.

At each control cycle, the controller selects a look-ahead reference $\mathbf{x}_{ref}$ from the planned waypoint sequence. The controller first identifies the nearest waypoint to the current state using a weighted distance metric that combines base position error, heading error, and joint error. From this nearest waypoint, it selects the reference state at a fixed look-ahead offset along the trajectory.

The control law is derived by linearizing the nonholonomic base dynamics around the current heading θ:

$$\mathbf{x}_{k+1} \approx \mathbf{x}_k + \mathbf{B}(\theta)\, \mathbf{u}_k \tag{10}$$

where the input matrix $\mathbf{B}(\theta) \in \mathbb{R}^{11\times 10}$ encodes the unicycle kinematics for the base and identity mappings for the joints:

$$\mathbf{B}(\theta) = \begin{bmatrix} \cos\theta & 0 & 0 & \cdots & 0 \\ \sin\theta & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & \mathbf{I}_7 \end{bmatrix} \tag{11}$$

Given the state error $\Delta\mathbf{x} = \mathbf{x}_{ref} - \mathbf{x}$, the controller solves a one-step quadratic program:

$$\min_{\mathbf{u}} \ \|\Delta\mathbf{x} - \mathbf{B}\mathbf{u}\|^2_{\mathbf{Q}} + \|\mathbf{u}\|^2_{\mathbf{R}} \tag{12}$$

where $\mathbf{Q} = \mathrm{diag}(20, 20, 15, 12, \ldots, 12)$ weights the tracking error and $\mathbf{R} = \mathrm{diag}(0.5, 0.8, 1.0, \ldots, 1.0)$ regularizes the control effort. The closed-form solution is:

$$\mathbf{u}^* = (\mathbf{B}^\top\mathbf{Q}\mathbf{B} + \mathbf{R})^{-1}\mathbf{B}^\top\mathbf{Q}\,\Delta\mathbf{x} \tag{13}$$

The resulting command is scaled by a gain factor $\alpha = 2.5$ and clamped to velocity limits ($v_{max} = 1.5$ m/s, $\omega_{max} = 1.0$ rad/s, with per-joint velocity limits of 0.1 rad/s for the torso and 1.0 rad/s for arm joints).

G. Real Robot Deployment

a) Goal Specification and Target Tracking: The user specifies the target by pointing at an object in an initial RGB-D image. We extract the target's 3D pose via instance segmentation and establish a semantic label using open-vocabulary object detection from a predefined list of graspable objects (e.g., apple, mug, bottle). During execution, as the robot observes from different viewpoints, we re-identify the target by finding the detected object that (1) matches the semantic label and (2) has the nearest bounding box center to the projected previous target coordinate. This semantic-geometric association enables robust tracking despite viewpoint changes and localization drift.

b) ROS Integration: Our system integrates several ROS packages: `ros_control` and MoveIt for arm trajectory execution, `message_filters` for synchronized RGB-D observation callbacks, `tf` for coordinate frame transformations, and the ROS Navigation stack (AMCL) for localization.

H. Benchmark Generation

We generate a benchmark of 400 test scenarios (20 scenes $\times$ 20 objects per scene) through a multi-stage validation pipeline.

a) Scene Selection: We select 20 scenes from ReplicaCAD [41] that represent diverse indoor environments: living rooms, kitchens, bedrooms, and offices. Scenes are chosen to vary in room size (15–50 m^2), furniture density (3–12 major pieces), and navigational complexity (open spaces vs. narrow passages).

b) Object Placement: For each scene, we procedurally place YCB objects on horizontal support surfaces (tables, counters, shelves). Placement surfaces are detected via point-cloud-based plane segmentation: we sample surface points, filter by upward-facing normals (> 0.95 dot product with Z-axis), cluster spatially adjacent cells, and retain only the topmost surface for each horizontal footprint. Objects are sampled from 50 YCB objects: master chef can, sugar box, tomato soup can, mustard bottle, tuna fish can, potted meat can, banana, strawberry, apple, lemon, peach, pear, orange, plum, bleach cleanser, bowl, mug, power drill, flat screwdriver, hammer, medium clamp, extra large clamp, mini soccer ball, softball, baseball, tennis ball, racquetball, golf ball, foam brick, marbles (a, b), cups (a–j), colored wood blocks, toy airplane, lego duplo (a–f), and rubiks cube. Initial placement positions are sampled with rejection sampling to ensure minimum separation between object centers (20 cm).

c) Drop Checking: Each placed object undergoes stability validation. We spawn the object 20 cm above the sampled surface point and simulate 100 physics steps using ManiSkill3. An object passes drop checking if:

1) The object's final height is within the expected range relative to the support surface ($\pm$5 cm below to 30 cm above)
2) The object's total displacement from spawn is less than 30 cm (no excessive rolling/bouncing)
3) No collision with existing objects at spawn time

Objects failing these checks are re-sampled at alternative positions until a stable placement is found or the maximum attempts (100) are exceeded.

d) Oracle Grasp Checking: Stable objects undergo graspability validation to ensure each benchmark object is actually graspable under ideal conditions. The procedure is:

1) **Multi-view rendering:** We render 20 RGB-D images of each object from viewpoints distributed on a circle (radius 0.3 m, height 0.3 m above the object center, uniformly spaced in azimuth).
2) **Grasp candidate generation:** For each view, we run Contact-GraspNet [40] on the segmented point cloud, filter by confidence score (≥ 0.4), and aggregate candidates across views. We apply farthest-point sampling to select up to 256 diverse grasps.
3) **Physics-based validation:** Each grasp candidate is tested using a virtual parallel-jaw gripper (box approximation with 10 cm finger length, 10 cm max opening):
   - The gripper approaches from a 10 cm pre-grasp offset along the approach direction
   - The gripper closes to 2 mm opening width
   - The gripper retreats to the pre-grasp pose, then lifts vertically by 10 cm
   - Success requires: both fingers contact the object, object displacement $<$3 cm during a 30-step stability hold, and object lifts at least 3 cm
4) **Inclusion criterion:** An object is included in the benchmark if at least one grasp candidate succeeds. Objects with zero successful grasps are replaced with alternative placements.

e) Dynamic Scenario Generation: For dynamic environment evaluation, we augment static scenarios with sudden obstacle appearances:

- **Pedestrian obstacles:** A cylindrical obstacle (radius 0.3 m, height 1.7 m) appears at a predefined position along the robot's path when the robot approaches within 1.7 m. Positions are manually selected to block the initial planned trajectory while ensuring alternative paths exist.
- **Table-top obstacles:** A small object appears on the table surface near the target when the robot reaches the pre-grasp pose. This tests perception updates and grasp replanning under tight spatial constraints.

We note that these dynamic scenarios use suddenly appearing static obstacles rather than continuously moving agents (e.g., walking pedestrians). This design ensures reproducibility and guarantees that an alternative path to the target always exists, enabling controlled evaluation of reactive replanning. Evaluating under continuously moving obstacles remains a direction for future work.

I. Evaluation Metrics

We evaluate system performance using grasp success rate as the primary metric and categorize failures to diagnose system behavior.

a) Success Criterion: A trial is marked successful if the robot:

1) Approaches the target object without collision
2) Executes a grasp that achieves stable contact with the object
3) Lifts the object above a height threshold (10 cm above the support surface)
4) Maintains stable grasp for 2 seconds after lifting

This physics-based validation accounts for the robot's full kinematic and dynamic constraints during execution.

b) Failure Categories: We categorize failures into three groups to identify system bottlenecks:

Execution Failures occur during physical interaction:

- **Collision:** Any contact between the robot body (excluding gripper fingers) and environment obstacles or the target object before grasp initiation.
- **Grasp Failure:** The gripper closes but fails to establish stable contact, or the object slips during lifting. This

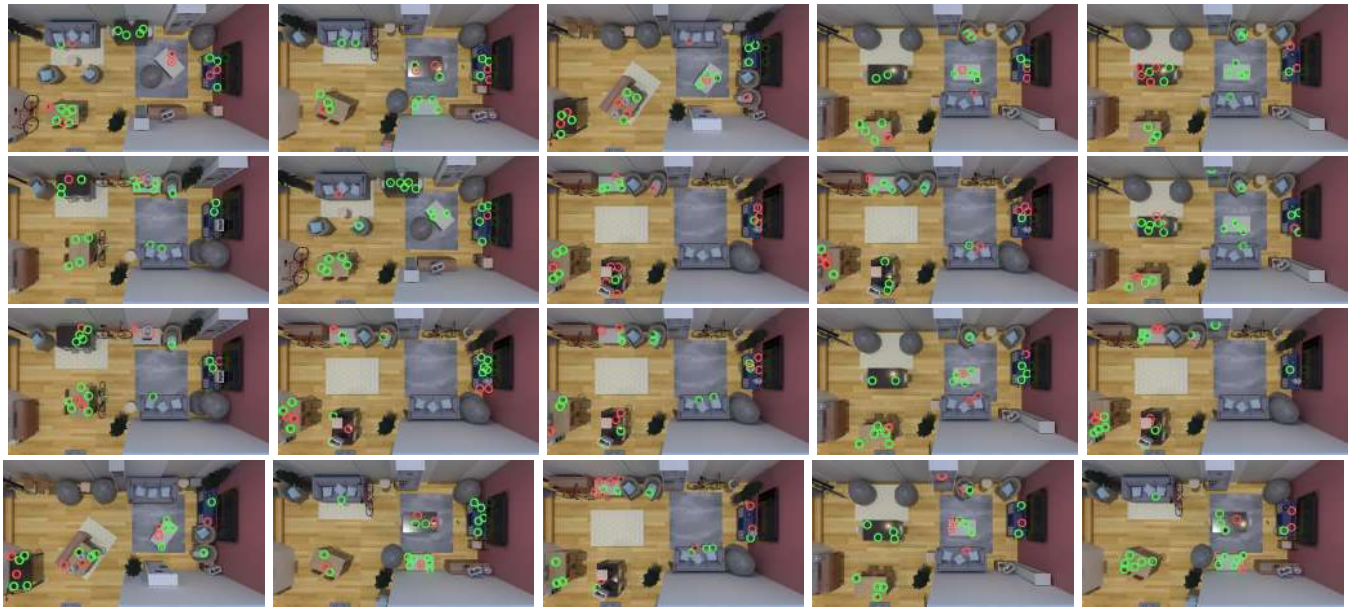

Fig. 12. **Simulation Test Scenarios.** Top-down views of all 20 ReplicaCAD scenes used for evaluation. Green dots indicate robot starting positions; red dots indicate target object locations. Scenes vary in room layout, furniture density, and navigational complexity.

includes cases where the grasp pose is geometrically valid but execution errors (positioning, timing) cause failure.

- **IK Failure:** The inverse kinematics solver cannot find a valid joint configuration for the commanded end-effector pose during execution.

System Limitations represent fundamental constraints:

- **Reachability:** No valid whole-body configuration exists that places the end-effector at any feasible grasp pose. This occurs when the target is too far, too high, or in a confined space that the robot cannot physically access.
- **Perception:** The target object cannot be reliably detected or localized due to occlusion, lighting conditions, or sensor noise. Also includes cases where environmental geometry is incorrectly estimated, leading to invalid planning.

Planning Failures indicate algorithmic limitations:

- **Planning Timeout:** The motion planner cannot find a collision-free path within the allocated time budget (200 ms per attempt, up to 3 attempts). While the average planning time is 50–80 ms for feasible problems, failures at the timeout typically indicate that the target configuration is infeasible under the current map or that the environment is too constrained for a collision-free path to exist, rather than reflecting mere computational limitations.
- **Subgoal Generation Failure:** No valid pre-grasp or observation configuration can be sampled that satisfies all constraints (collision-free, reachable, non-occluded).

c) Evaluation Scenarios: Figure 12 shows all 20 simulation scenes with test case distributions. Green and red dots indicate robot starting positions and target object locations, respectively.

d) Real-World Locations: We test in five indoor settings: (1) dining table, (2) kitchen counter, (3) workstation, (4) coffee table, and (5) sofa. These locations span different approach constraints and obstacle configurations. For each location, we configure 4 unique object–start pairs (varying target object identity and robot starting position), yielding 20 static configurations and 20 dynamic configurations (40 total). This doubled evaluation set (compared to a minimal 10+10 split) improves statistical reliability of real-world results. We evaluate only the full system in real-world experiments; baseline methods lacking safety-critical components (e.g., active perception, reactive replanning) pose risks of physical damage to the robot and environment during unguarded execution.

J. Detailed Failure Analysis

a) Spatial Failure Patterns: Figure 9 in the main paper visualizes the spatial distribution of successes and failures in a representative scene. Three distinct failure regions emerge: (1) objects located deep on tables fail due to limited arm reachability from feasible base positions; (2) objects near walls fail because environmental constraints eliminate viable approach directions, yet the grasp detector lacks kinematic awareness to account for this; (3) objects in cluttered areas fail due to the difficulty of isolating individual items for precision grasping.

b) Grasping Failures: Grasping represents the most significant bottleneck (13–16% of trials). The grasp detection model exhibits sensitivity to out-of-distribution scenarios: large objects frequently produce low-quality grasp poses due to training data bias toward smaller objects, and grasp quality degrades with distance from the observation point as depth uncertainty increases. The core limitation stems from decoupling grasp detection from kinematic feasibility. In constrained spaces like shelf corners, the model often predicts top-down

grasps that are unreachable due to gripper-environment collisions or joint limit violations. Additionally, small lightweight objects exhibit high sensitivity to positioning errors, and certain geometries (e.g., objects with handles) are incompatible with parallel-jaw grippers. To further diagnose this bottleneck, we manually labeled 46 grasp failures into four modes (Table I): *infeasible* (all motion planning attempts fail or the IK solution lies in collision), *inaccurate* (the grasp pose is a false positive—geometrically misaligned with the object), *instability* (the grasp completes but the object slips during lift), and *out-of-distribution* (no valid candidate is returned by the grasp network). The dominance of the *infeasible* mode (43.5%) reveals a domain shift between mobile grasping and tabletop grasping: the off-the-shelf grasp detector ranks candidates by geometric quality alone, so geometrically optimal grasps are frequently kinematically infeasible in the constrained spaces our robot encounters (e.g., shelves, walls, deep tables). These failures are largely independent of the system architecture and would affect any pipeline that uses decoupled grasp detection. Kinematically-informed grasp scoring is therefore a clear direction for future work.

TABLE I
DISTRIBUTION OF GRASP FAILURE MODES ACROSS 46 LABELED TRIALS.

Mode	Infeasible	Inaccurate	Instability	OOD
Count (%)	20 (43.5%)	14 (30.4%)	11 (23.9%)	1 (2.2%)

c) Planning Failures: Planning failures (8–14% of trials) arise from four sources: (1) perception noise creating false obstacle detections that block valid paths; (2) planner false negatives where feasible paths exist but are not found within time limits; (3) insufficient path smoothness causing controller tracking errors; (4) paths passing too close to obstacles with insufficient safety margin. Our system incorporates fallbacks and multiple planning attempts, but these issues still contribute to failures, highlighting the importance of robust planning under perceptual uncertainty.

d) Collision Failures: Collision failures (6–10% of trials) expose perception and execution limitations. The active perception system prioritizes high-velocity and near-target regions, creating blind spots where thin or non-convex objects (plant stems, chair legs) are missed. Conservative ground plane removal, necessitated by sensor noise, inadvertently filters low-lying obstacles. Execution uncertainty also contributes: in-place rotations exhibit translational drift sufficient to cause collisions in tight spaces. In dynamic environments, sensor synchronization delays create spurious obstacle detections during reactive replanning. The static-to-dynamic gap further reflects two compounding effects: the planner produces tighter trajectories around newly appearing obstacles (raising collisions from 6.3% to 9.8%), and new obstacles occlude previously observed regions, degrading observation efficiency and contributing to the rise in planning failures (8.8% to 14.0%). These effects motivate two future directions: incorporating predictive motion models for humans and dynamic objects to reduce reactive replanning, and developing probabilistic completeness guarantees for visibility-constrained planning.

K. Additional Experiments

Table II provides detailed failure breakdowns for all six methods across both simulation environments. The reported numbers are stable across 5 independent runs of the 400-scenario benchmark (2,000 trials per condition): our system achieves $69.5\% \pm 1.0\%$ (static) and $61.2\% \pm 2.4\%$ (dynamic) mean $\pm$ std, with 95% Wilson confidence intervals of $[67.4\%, 71.5\%]$ and $[59.0\%, 63.3\%]$ that contain the single-run results reported in the main paper. Per-scene success rates range from 51%–81% (static) and 47%–73% (dynamic), with cross-scene standard deviations of $\sigma_{\text{scene}} = 8.2\%$ and 6.9%, reflecting layout and clutter diversity rather than system instability.

1) System-Level Comparisons: We evaluate two alternative system designs that make different architectural choices for integrating navigation and manipulation, complementing the ablation study in the main paper.

a) CapMap Placement: This system design follows the sequential navigation-then-manipulation paradigm but incorporates manipulation-aware base placement using the capability-map-based method of Reister et al. [38]. Rather than terminating navigation at a fixed distance from the target (as in the navigation-and-manipulation design), this architecture uses the pre-computed capability map to select a base pose that ensures the target lies within the arm's reachable workspace.

The system uses the same pre-grasp sampler as ours (Section B), which samples complete configurations (base + torso + arm). However, by design only the base pose is extracted for navigation. Once the robot reaches the selected base pose, it switches to arm-only motion planning for the grasping phase, maintaining a strict separation between the navigation and manipulation stages.

As shown in Table II and Figure 13, this design achieves 44.25% and 40.00% success rates in the unknown static and dynamic environments, respectively. Interestingly, despite the kinematically-informed base placement, it performs slightly *worse* than the simpler fixed-distance navigation design (46.00% and 40.00%) in the static environment. The manipulation-awareness constraints narrow the set of feasible base poses: by requiring the target to lie within the arm's reachable workspace, the capability map excludes base positions that, while not optimal for reachability, would have been safer in terms of collision avoidance. This over-constraining effect is evident in its collision rate (23.25% static, 23.50% dynamic)—among the highest of all methods—as the robot is forced into tighter base placements near furniture and obstacles. Planning failures remain significant (17.25% static, 22.25% dynamic), as the fixed base pose may not provide feasible arm trajectories once the full scene geometry is revealed. These results highlight a fundamental limitation of the sequential paradigm: even with manipulation-aware positioning, decoupling navigation and manipulation prevents

TABLE II

SIMULATION EVALUATION RESULTS. SUCCESS RATES AND FAILURE BREAKDOWN (EXECUTION: COLLISION/GRASP/IK; SYSTEM LIMITATIONS: REACHABILITY/PERCEPTION; PLANNING) FOR ALL METHODS ACROSS TWO SIMULATION SCENARIOS. ALL VALUES ARE PERCENTAGES (%).

Method	Success	Execution			System Limitations		Planning
		Collision	Grasp	IK	Reachability	Perception	
Unknown Environment (Simulation)							
Navigation-and-manipulation	46.00	17.50	15.75	0.00	0.00	0.25	20.50
CapMap Placement	44.25	23.25	14.50	0.00	0.25	0.50	17.25
Direct Grasping	28.25	3.25	4.00	8.75	1.75	0.75	53.25
Closed-Loop Replanning	61.25	6.75	15.25	0.75	0.50	0.25	15.25
Velocity-agnostic	63.75	8.00	15.50	0.25	2.50	0.25	9.75
Ours (Full System)	68.75	6.25	13.25	0.50	2.25	0.25	8.75
Dynamic Environment (Simulation)							
Navigation-and-manipulation	40.00	24.75	14.25	0.00	0.00	1.50	19.50
CapMap Placement	40.00	23.50	13.00	0.00	0.00	1.25	22.25
Direct Grasping	25.75	1.50	2.00	5.75	1.00	0.75	63.25
Closed-Loop Replanning	56.75	8.50	14.75	0.50	0.75	0.25	18.50
Velocity-agnostic	54.00	13.75	15.50	0.75	3.25	0.00	12.75
Ours (Full System)	58.00	9.75	16.00	0.00	2.25	0.00	14.00

the system from jointly optimizing for both reachability and safety.

b) Closed-Loop Replanning: This system design integrates navigation and manipulation into a single planning loop, similar to our approach, but forgoes structured task management. The system continuously replans whole-body motions toward the sampled pre-grasp configuration until the robot reaches it, then initiates the grasping phase. Unlike our behavior tree architecture, which provides distinct observation, navigation, and manipulation stages with explicit state transitions, this design relies solely on continuous replanning to handle environmental changes without high-level task coordination.

This design achieves 61.25% and 56.75% success rates in the unknown static and dynamic environments, respectively (Figure 13). While substantially better than the sequential designs, it still falls short of our full system (68.75% and 58.00%). Without the structured state transitions of a behavior tree, continuous replanning can oscillate between configurations without converging, leading to higher planning failures (15.25% static, 18.50% dynamic). Collision rates (6.75% static, 8.50% dynamic) are comparable to our system, confirming that the integrated approach to navigation and manipulation is inherently effective for collision avoidance. However, the absence of explicit observation and subgoal management stages limits the system's ability to gather sufficient environmental information before committing to a plan, reducing overall task success.

2) Additional Analysis:

a) Path Efficiency Analysis: We evaluate path efficiency using Success weighted by Path Length (SPL), which measures how efficiently the robot reaches the target relative to the optimal path length. SPL is defined as:

$$\text{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \cdot \frac{l_i}{\max(p_i, l_i)} \tag{14}$$

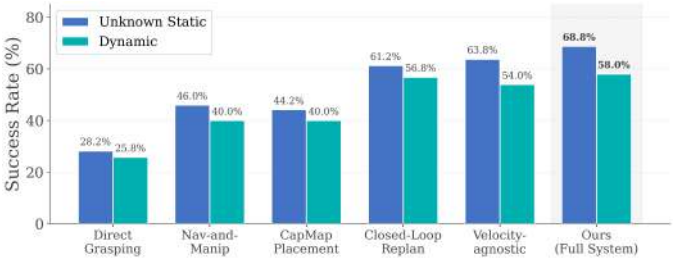

Fig. 13. **System-level comparison.** Success rates for all six methods across unknown static and dynamic environments. The CapMap Placement design suffers from high collision rates due to its sequential architecture, while the Closed-Loop Replanning design improves over sequential designs but lacks the structured task management of our behavior tree.

where N is the number of episodes, $S_i \in \{0, 1\}$ is a binary success indicator, l_i is the shortest path distance from start to goal, and p_i is the actual path length taken by the robot. This metric rewards successful episodes while penalizing inefficient paths. A perfect score of 1.0 indicates all episodes succeeded via optimal paths.

To compute the optimal path length l_i, we build a 2D occupancy grid from the scene point cloud by projecting points within a height range of $[0.06, 1.5]$ m onto the XY plane at 0.05 m resolution. Obstacles are inflated by the robot's base radius (0.29 m) using a circular structuring element, so that paths on the grid are guaranteed to be collision-free for the robot footprint. We then run BFS on the 8-connected grid from the robot's initial position to the nearest reachable cell to each target object, yielding the shortest collision-free path in meters. We note that this 2D shortest path is an approximation of the true optimal whole-body path, as computing the optimal trajectory in the full configuration space is intractable; nevertheless, it provides a consistent and comparable baseline across all methods.

Figure 14 shows the SPL results for all methods. Our full

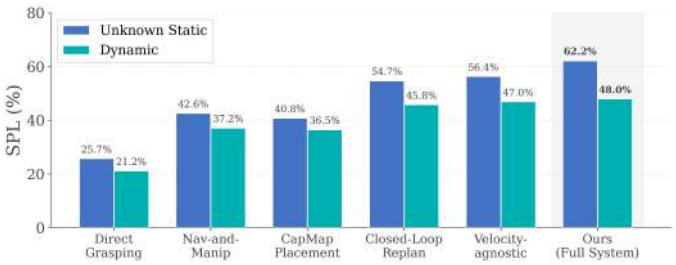

Fig. 14. **Path efficiency analysis (SPL).** Success weighted by Path Length for all methods. Higher SPL indicates both higher success rate and more efficient paths relative to the shortest collision-free path. Our system achieves the highest SPL across both environments.

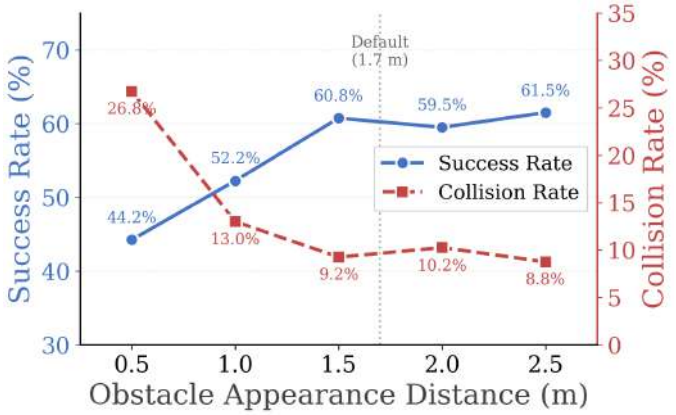

Fig. 15. **Obstacle distance analysis.** Success rate and collision rate as a function of obstacle appearance distance. The system requires at least 1.5 m of reaction distance to achieve near-optimal avoidance performance. Beyond this threshold, success rate plateaus as collision avoidance saturates and remaining failures are dominated by grasp and planning limitations.

system achieves the highest SPL in both environments (62.2% static, 48.0% dynamic), indicating that it not only succeeds more often but also follows more efficient paths. The Velocity-agnostic ablation (56.4% static, 47.0% dynamic) and Closed-Loop Replanning (54.7% static, 45.8% dynamic) achieve similar SPL, despite different success rates, suggesting that the closed-loop baseline takes longer paths when it does succeed. The sequential baselines (Nav-and-Manip: 42.6%/37.2%, CapMap Placement: 40.8%/36.5%) show moderate SPL, while Direct Grasping (25.7%/21.2%) has the lowest SPL due to both low success rate and the absence of navigation planning. Notably, the gap between our system and baselines is larger in SPL than in raw success rate, confirming that our integrated approach produces more direct, efficient trajectories.

b) Obstacle Distance Analysis: We analyze the system's reactive capability by measuring success rate as a function of obstacle appearance distance, i.e., the distance between the robot and a dynamic obstacle when it first appears. This characterizes the minimum reaction distance required for successful obstacle avoidance. We evaluate trigger distances from 0.5 m to 2.5 m rather than starting from zero because the

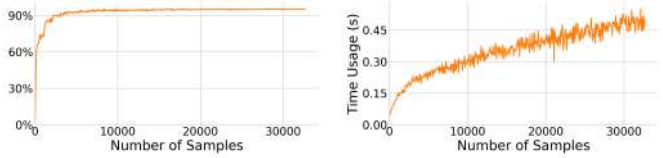

Fig. 16. **Sampling completeness analysis.** Empirical completeness of pre-grasp configuration sampling. **Left:** Success rate as a function of the number of samples, showing rapid convergence to approximately 95%. **Right:** Computational time scales approximately linearly with the number of samples.

dynamic obstacles in our benchmark are manually positioned along likely robot paths, with each placement verified to leave at least one alternative route to the target (Section H). Since the motion planner is stochastic, at very short trigger distances the robot's planned path may not pass through the trigger zone at all, causing the obstacle to never appear and reducing the experiment to the static setting. Starting at 0.5 m ensures the obstacle reliably triggers across the majority of trials.

Figure 15 shows the results across five trigger distances (0.5–2.5 m). Success rate increases sharply from 44.2% at 0.5 m to 60.8% at 1.5 m, then plateaus at approximately 60–61% for distances beyond 1.5 m. The primary driver of this trend is collision rate, which drops from 26.8% at 0.5 m to 9.2% at 1.5 m as the robot gains sufficient reaction time and distance to replan around suddenly appearing obstacles. At trigger distances below 1.0 m, the robot often cannot decelerate or replan quickly enough to avoid the obstacle, as the perception–planning–execution loop requires a minimum reaction distance to execute corrective maneuvers. Beyond 1.5 m, collision rates stabilize at 8.8–10.2%, suggesting the system achieves near-optimal reactive avoidance at this range. Grasp and planning failure rates remain relatively stable across all distances (9.5–13.8% and 12.0–18.0%, respectively), confirming that these failure modes are independent of obstacle appearance timing and instead reflect inherent task difficulty. The default trigger distance of 1.7 m used in our dynamic environment experiments (Section H) lies within the plateau region, confirming it provides sufficient reaction margin.

To verify that the conclusions hold under continuously moving obstacles rather than the suddenly-appearing obstacles used by default, we re-ran the same 400 dynamic scenarios with the obstacles moving across the robot's planned path at 0.2 m/s. Our full system achieves a success rate of 54.4%, slightly lower than the sudden-appearance setting (58.0%). The small difference is consistent with our system being purely reactive: a moving obstacle entering the camera's field of view via closed-loop detection is algorithmically equivalent to a suddenly appearing one from the planner's perspective. The slight degradation arises from the moving obstacle additionally invalidating recently observed free space along its trajectory, which is not the case in the sudden-appearance setting.

c) Sampling Completeness Analysis: We analyze the empirical completeness of the proposed pre-grasp configuration sampling procedure (Section B) by examining how the success rate and computational time scale with the number of samples.

As shown in Figure 16, the success rate increases rapidly with the number of samples and converges to approximately 95% at around 2,000 samples, after which additional samples yield diminishing returns. Note that the success rate plateaus below 100% because a fraction of target configurations are inherently infeasible due to kinematic reachability limits, rather than due to insufficient sampling. This confirms that the sampler efficiently covers the feasible configuration space with a moderate number of samples. Meanwhile, the computational time grows approximately linearly with the number of samples. In practice, we use 2,560 samples (20 base poses $\times$ 128 arm/torso configurations), which achieves near-convergence success rates while maintaining real-time computational performance.

d) Computational Cost Breakdown: We profile the runtime of each module to characterize the real-time feasibility of the full pipeline. Each control cycle comprises three components that run on every iteration: map update via ray casting (69 ms on average), gaze optimization (26 ms), and collision checking against the current trajectory (3 ms), totaling approximately 100 ms. When motion replanning is triggered (e.g., by a path collision or subgoal change), the cycle extends to roughly 150 ms. Pre-grasp subgoal sampling and grasp detection are invoked only at phase transitions and not on every cycle: subgoal sampling takes 369 ms on average and is invoked roughly 3.6 times per task, while grasp detection (ContactGraspNet [40]) takes 1.2 s and is invoked roughly 1.3 times per task. The velocity-aware gaze policy itself contributes only 26 ms per cycle on top of the volume-based variant; all other modules are shared between our system and the baselines, so this overhead is the primary computational difference between our system and the velocity-agnostic ablation.