Integration of Robot and Scene Kinematics for Sequential Mobile Manipulation Planning

Ziyuan Jiao, *Member, IEEE*, Yida Niu, *Student Member, IEEE*, Zeyu Zhang, *Member, IEEE*, Yangyang Wu, Yao Su, *Member, IEEE*, Yixin Zhu, *Member, IEEE*, Hangxin Liu, *Member, IEEE* and Song-Chun Zhu, *Fellow, IEEE*

***Abstract*—We present a Sequential Mobile Manipulation Planning (SMMP) framework that can solve long-horizon multi-step mobile manipulation tasks with coordinated whole-body motion, even when interacting with articulated objects. By abstracting environmental structures as kinematic models and integrating them with the robot's kinematics, we construct an Augmented Configuration Apace (A-Space) that unifies the previously separate task constraints for navigation and manipulation, while accounting for the joint reachability of the robot base, arm, and manipulated objects. This integration facilitates efficient planning within a tri-level framework: a task planner generates symbolic action sequences to model the evolution of A-Space, an optimization-based motion planner computes continuous trajectories within A-Space to achieve desired configurations for both the robot and scene elements, and an intermediate plan refinement stage selects action goals that ensure long-horizon feasibility. Our simulation studies first confirm that planning in A-Space achieves an 84.6% higher task success rate compared to baseline methods. Validation on real robotic systems demonstrates fluid mobile manipulation involving (i) seven types of rigid and articulated objects across 17 distinct contexts, and (ii) long-horizon tasks of up to 14 sequential steps. Our results highlight the significance of modeling scene kinematics into planning entities, rather than encoding task-specific constraints, offering a scalable and generalizable approach to complex robotic manipulation.**

***Index Terms*—Sequential mobile manipulation planning, kinematics, trajectory optimization, and service robot.**

I. Introduction

AUTONOMOUS robots are increasingly being integrated into diverse environments in human society. Whether assisting people in daily activities [3, 4] or operating in outposts such as space stations or extraterrestrial bases [5, 6], robot operations face significant challenges in performing sequential mobile manipulation tasks that require a range of manipulation skills and the ability to sequence these skills in expansive workspaces.

Fig. 1(a) illustrates a typical Sequential Mobile Manipulation Planning (SMMP) scenario. Operating in cluttered workspaces poses significant challenges due to complex obstacle configurations [7]. Robots are often required to balance both navigation and manipulation, *i.e.*, mobile manipulation, to accomplish their goals [8, 9]. Moreover, contact with diverse structures and objects introduces a wide range of task objectives and constraints, which are difficult to emulate in advance, particularly when dealing with articulated objects [1, 10, 11]. Compounding this difficulty, robot actions can change the environment in ways that hinder the feasibility of future steps in long-horizon tasks [12–15]. Therefore, the successful execution of an action in long-horizon mobile manipulation requires not only coordinating base-arm-object trajectories for individual steps, but also reasoning about the long-term implications of each action on future task feasibility.

Achieving coordinated trajectories of the robot's base, arm, and manipulated object can become computationally intractable in long-horizon tasks due to the inherently interdependent configuration spaces during interactions, as shown in Fig. 1(b). Consequently, the likelihood of finding connected feasible paths across consecutive steps is low, and costly backtracking is often required when the planner encounters dead ends. A hierarchical strategy is typically employed to decompose the task execution into a sequence of primitive motions [12, 14, 16], facilitating more efficient trajectory generation and reducing computation costs in the face of errors. However, current hierarchical methods such as Task and Motion Planning (TAMP) are primarily effective only for pick-and-place tasks [17–19], failing to scale to complex mobile manipulation tasks. This limitation arises because complex mobile manipulation tasks require tightly coordinated navigation and manipulation, which are difficult to express symbolically. Semantic symbols often fail to capture critical geometric constraints necessary for task success, such as valid base positioning, interdependent base and arm movements, and collision avoidance. For example, the tasks in Fig. 1 involve coupled base-arm-object interactions that would demand an intractable number of symbolic predicates to model accurately.

In stark contrast, humans exhibit fluid manipulation skills and interact adeptly with their environment. Theories in cognitive psychology and philosophy suggest the concept of body schema: humans maintain a flexible representation of their bodies, enabling them to treat manipulated objects as extensions of their limbs during interactions [20, 21]. Embodied

Corresponding author: Hangxin Liu.

This article is an extended version of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Oct., 2021 [1, 2].

This work was supported in part by the National Natural Science Foundation of China (Grant No.52305007).

Ziyuao Jiao, Yida Niu, Zeyu Zhang, Yangyang Wu, Yao Su, Hangxin Liu, and Song-Chun Zhu are with State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI), Beijing 100080, China (emails: jiaoziyuan@bigai.ai; niuyida@bigai.ai; zhangzeyu@bigai.ai; wuyangyang@bigai.ai; suyao@bigai.ai; liuhx@bigai.ai; sczhu@bigai.ai).

Yida Niu is also with Institute for Artificial Intelligence, Peking University, Beijing 100871, China.

Yixin Zhu is with the School of Psychological and Cognitive Sciences, and the Institute for Artificial Intelligence, Peking University, Beijing 100871, China (email: yixin.zhu@pku.edu.cn).

Song-Chun Zhu is also with Institute for Artificial Intelligence and School of Artificial Intelligence, Peking University, Beijing 100871, China, and also with Department of Automation, Tsinghua University, Beijing 100084, China.

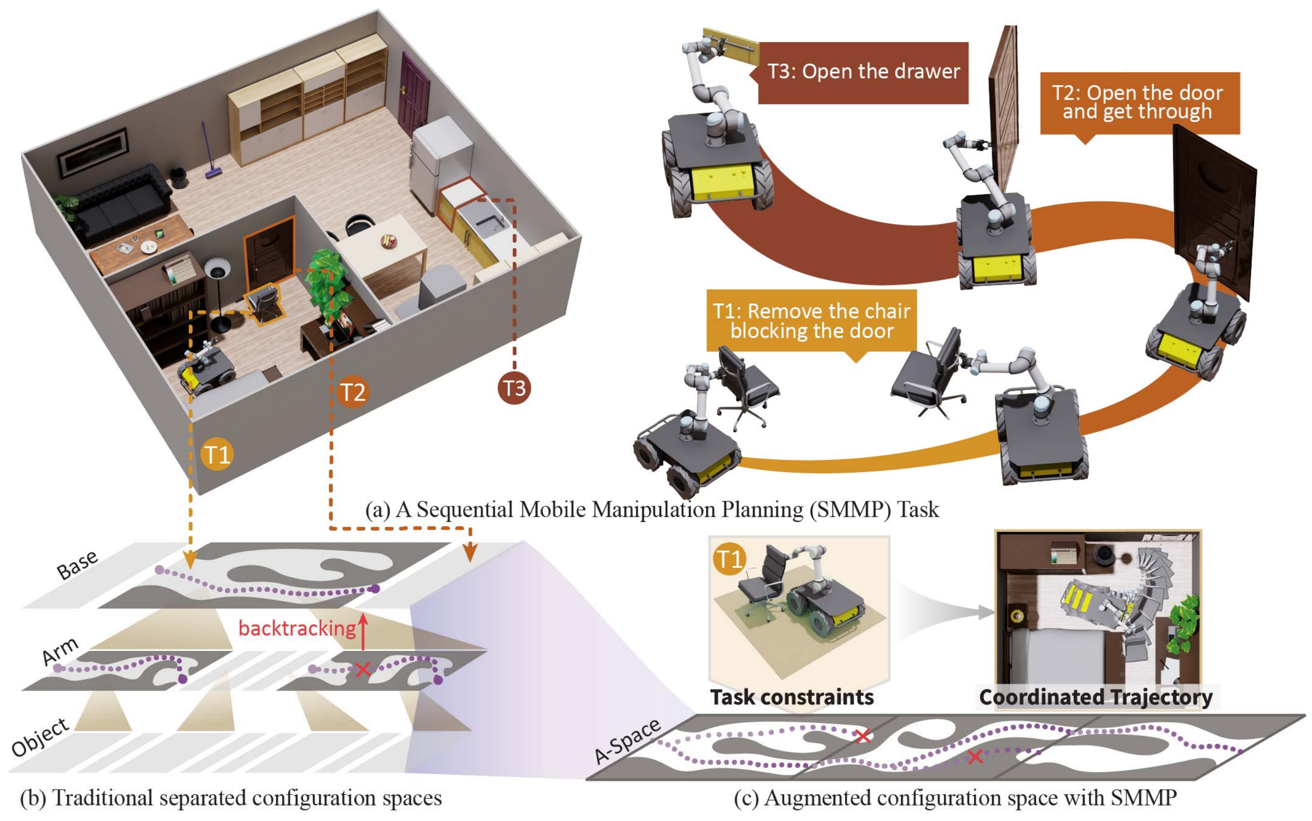

T3: Open the drawer
T2: Open the door and get through
T1: Remove the chair blocking the door
T1
T2
T3
(a) A Sequential Mobile Manipulation Planning (SMMP) Task
Base
Arm
Object
backtracking
T1
Task constraints
Coordinated Trajectory
A-Space
(b) Traditional separated configuration spaces
(c) Augmented configuration space with SMMP

Fig. 1: **An exemplar household task illustrating the advancement of proposed Sequential Mobile Manipulation Planning (SMMP) framework with A-Space compared to traditional planning.** (a) In this long-horizon task, the robot must (T1) remove the chair to approach the bedroom door, (T2) open the door and pass through it, and (T3) open the kitchen drawer. (b) Separated base-arm-object planning faces inherent challenges: as the base moves, the arm and object's configuration space evolves, frequently making initially feasible trajectories infeasible. This approach simplifies the planning complexity but may require costly backtracking across different configuration spaces, particularly in tasks demanding whole-body coordination. (c) By leveraging A-Space, separate configuration spaces are unified through Augmented Kinematic Representation (AKR), incorporating various task constraints to enable coordinated base-arm-object trajectories across multiple steps via trajectory optimization, reducing the need for hierarchical backtracking.

cognition studies further highlight that human intelligence is deeply intertwined with the environment [22, 23].

Drawing from these insights, we propose to treat the environment and the robot embodiment as a whole for an efficient solution to mobile manipulation tasks. Specifically, we consolidate kinematic abstractions of scene elements [24], robot arm, and navigational movements into an Augmented Kinematic Representation (AKR). This consolidation merges originally separated yet entangled robot and object's configuration spaces into a single one, which we termed Augmented Configuration Space (A-Space), as depicted in Fig. 1(c). From this perspective, planning sequential mobile manipulation in A-Space can jointly account for the reachability of the robot base, arm, and manipulated objects and their inherent motion constraints, thus better ensuring spatial feasibility of sophisticated robot movements and resolving action temporal dependency among long-horizon tasks.

As A-Space is highly complex due to the extended Degree of Freedoms (DoFs), we design a tri-level planning framework for efficient planning within it. By formulating an optimization-based motion planner, we can compute continuous trajectories to reach the desired configurations of both robot and scene entities, resulting in coordinated whole-body motions that satisfy relevant constraints. Furthermore, we design a task planner that defines symbolic states more intuitively and models actions along with their effects on A-Space, thereby enabling validation of motion feasibility over extended horizons by traversing temporally successive configuration spaces. Extending our previous foundation in task planning and motion planning [1, 2], a newly designed plan refinement algorithm further combines them to resolve potential conflicts between anticipated future actions. Together, these components form a scalable SMMP framework capable of handling long-horizon mobile manipulation tasks involving diverse interactions within complex environments. We demonstrate the framework's effectiveness on a real robot system by performing coordinated base-arm-object motions across 17 diverse scenarios involving complex environmental structures. Moreover, the system successfully completes a 14-step long-horizon mobile manipulation task in a cluttered living room, with each step characterized by unique contact configurations. Additional simulation studies further validate our approach, quantifying improvements in execution efficiency and planning success rates when using A-Space compared to traditional planning paradigms.

Our contribution is fourfold:

1) We introduce an SMMP framework that solves long-horizon mobile manipulation tasks, with a newly proposed plan refinement algorithm that considers future actions while generating the motion planning problem for the

current action, effectively increasing task success rates in complex long-horizon SMMP problems.
2) We model the mobile manipulation planning problem from the AKR perspective, formulating the mobile manipulation planning problem as a trajectory optimization problem within the A-Space that integrates task specifications.
3) We design a Planning Domain Definition Language (PDDL)-based task planning domain describing the evolution of the A-Space, generalizing it to various daily long-horizon indoor mobile manipulation tasks.
4) Through simulations, we validate the proposed method, achieving an 84.6% improvement in success rate over baseline methods. With extensive experiments on physical mobile manipulators, we demonstrate the proposed method's feasibility across 7 types of rigid and articulated objects in 17 different contexts, with long-horizon tasks involving up to 14 steps.

A. Overview

The remainder of this article is organized as follows. Sec. II reviews the literature and compares existing research with the contributions of this work. Sec. III introduces the proposed AKR-based modeling method for mobile manipulation. Based on the idea of AKR, Sec. IV formulates the corresponding motion planning and task planning setups, and Sec. V elaborates the newly proposed plan refinement algorithm that bridges AKR-based motion planning with task planning components into a coherent SMMP system. Finally, Sec. VI and Sec. VII demonstrate the efficacy of AKRs through simulations and experiments, respectively. Sec. VIII concludes the paper with an in-depth discussion of key findings and future directions.

II. Related Work

A. Mobile Manipulation

Recently, notable efforts have focused on algorithms and system implementations to coordinate navigation and manipulation for mobile manipulation, especially within household environments. For instance, graph search [25], equilibrium point control [26], adaptive control [10], impedance control [27], and model predictive control [28] have been introduced for tasks like opening doors and drawers. For object retrieval or relocation in confined and cluttered spaces, methods such as the coevolutionary algorithm in [29], which jointly optimizes grasping and base poses, and adaptive dimensionality reduction in [30], which manages the high DoFs search space, have shown promise. Other techniques include inverse kinematics branching for iterative optimization of base and joint motions [31] and holistic control of the arm and base as a unified structure [32]. While existing robotic planning methods achieve promising results on isolated tasks, such as door opening or object retrieval in controlled environments, their specialized, task-specific designs cannot generalize to broader scenarios requiring coordinated manipulation by the mobile base, manipulator arm, and target object. Yet, our SMMP scenario demands manipulation of objects with diverse kinematic structures in varied environments, where successful task execution critically depends on coordination among the mobile base, manipulator arm, and target object. In addition, deep Reinforcement Learning (RL) has recently gained popularity for manipulation tasks involving rich interactions. For example, [33] trains an RL policy for object retrieval on a physical manipulator, and [34] uses deep RL for whole-body control in door-opening tasks, while [35] abstracts the action space into base and arm sub-goals for long-horizon tasks in simulated environments. Although RL offers advantages for complex interaction planning, learned policies often suffer from poor transferability from simulation to real-world applications and do not scale effectively to long-horizon tasks due to substantial training time.

B. Multi-Modal Motion Planning (MMMP)

In sequential manipulation tasks, robots must repeatedly establish and release contact with various objects, exhibiting multi-modal behavior: contact states (discrete modes) constrain robot motions, effectively partitioning the environment's configuration space into interconnected manifolds. Transitions between manifolds indicate potential mode changes. Building on this concept, MMMP methods [36–40] aim to find feasible trajectories across different manifolds, producing motion plans applicable to sequential mobile manipulation tasks. For example, Hauser *et al.* [38] propose a scalable algorithm that randomly samples mode switches and motion paths on a known mode transition graph to generate a solution plan, while Toussaint *et al.* [40] abstracts contact modes using differentiable physics, enabling tool-use planning. These methods yield impressive results in planning multi-step actions, but they share a limitation common to MMMP: their planning domains are specifically designed and restricted to geometric features, necessitating extensive efforts in custom planner design and mode transition definitions for numerous contact modes. This proves inadequate for semantically rich environments where object relationships transcend simple contacts [18]. While the MMMP approach shares similarities with our work in terms of abstracting actions through contacts, the proposed AKR enables the use of off-the-shelf planning languages and aims to accommodate a broad range of mobile manipulation tasks without defining specific actions for each task.

C. TAMP

Thanks to the development of PDDL [41] and other planning languages, complex symbolic planning can be solved using standard algorithms [42, 43]. While symbolic planning effectively captures abstract concepts, it struggles to represent the feasibility of robot motions. This limitation has led the robotics community to integrate MMMP concepts with symbolic task planners, forming the field of TAMP [18]. Current TAMP approaches typically employ a bidirectional interface between task and motion planning [12, 44–46], but they remain computationally expensive due to their reliance on dense sampling in high-dimensional spaces [7]. Recent work has sought to address these inefficiencies: Zhang *et al.* [47] optimize symbolic state spaces to reduce redundant navigation actions, Yang *et al.* [48] leverage Vision-Language Models (VLMs) to propose high-level subgoals to prune search spaces, and Sung *et al.* [49] learns back-jump heuristics that identify the culprit action and bypass irrelevant backtracking steps. However, the iterative nature of TAMP approaches still imposes significant computational overhead, as failed motion planning attempts trigger backtracking and replanning of action sequences. As a result, many TAMP approaches simplify motion planning and limit themselves to basic manipulation tasks, avoiding the complexity of designing intricate planning domains and specific motion planners for complex mobile manipulation tasks.

Departing from traditional efforts in TAMP approaches that either optimize search strategies or redesign task-motion interfaces, this work proposes a new perspective by planning mobile manipulation tasks through the AKR. The proposed AKR constitutes an effective intermediate representation that can benefit TAMP in solving challenging sequential mobile manipulation tasks by improving computational efficiency through reducing intermediate variables and facilitating optimization-based motion planning.

This work builds upon our preliminary results presented in Jiao *et al*. [1, 2]. The extension features a more comprehensive literature review, the introduction of a new plan refinement algorithm that enhances planning success rates by selecting key AKR configurations throughout the action sequence, and extensive benchmarking that compares our SMMP framework to baselines using off-the-shelf motion planners. Additionally, we include implementation and large-scale experimentation on a physical mobile manipulator platform.

III. AKR Modeling

This section describes three key steps for integrating robot and scene models into one cohesive kinematic representation, termed Augmented Kinematic Representation (AKR).

A. Kinematic Representation

The kinematic representation used in this article is defined as a tree $\mathcal{T} = (V, S)$ where the rigid bodies of an articulated object are described as links $v_i \in V$, while their inherent motion constraints and spatial relations are represented by joints $s_{ij} \in S$. Specifically, the node set V includes a set of links $v_i = \langle o_i, \mathcal{F}_i \rangle$; each encodes a full geometry model o_i (*e.g.*, a triangular mesh or empty for dummy link), and a link frame $\mathcal{F}_i$. In addition, the root node of $\mathcal{T}$ is denoted as v_r. The edge set S includes a set of joints $s_{ij} = \langle r_{ij}, {}^i_jT \rangle$; each encodes the motion constraints r_{ij} (*e.g.*, bounded revolute or prismatic motion along an axis) between the parent link v_i and the child link v_j, and a spatial transformation ${}^i_jT$ from the parent link frame $\mathcal{F}_i$ to the child link frame $\mathcal{F}_j$. Based on the above notations, a kinematic chain $\mathcal{C}_{ij} = (V_{ij}, S_{ij})$ contains only nodes $V_{ij} \subseteq V$ and edges $S_{ij} \subseteq S$ that belong to a path between a node v_i and one of its descendant nodes v_j in $\mathcal{T}$.

B. AKR Modeling for Mobile Manipulation

To construct an AKR, $\mathcal{T}^A$, four key inputs are required: the manipulator's kinematics $\mathcal{T}^R$, object kinematics $\mathcal{T}^O$, a virtual mobile base $\mathcal{T}^B$, and a virtual attachment joint s_{ea} between the robot end-effector and the link to be grasping on the object. Fig. 2 illustrates the constructed AKR for opening a cabinet door with a physical mobile manipulator platform.

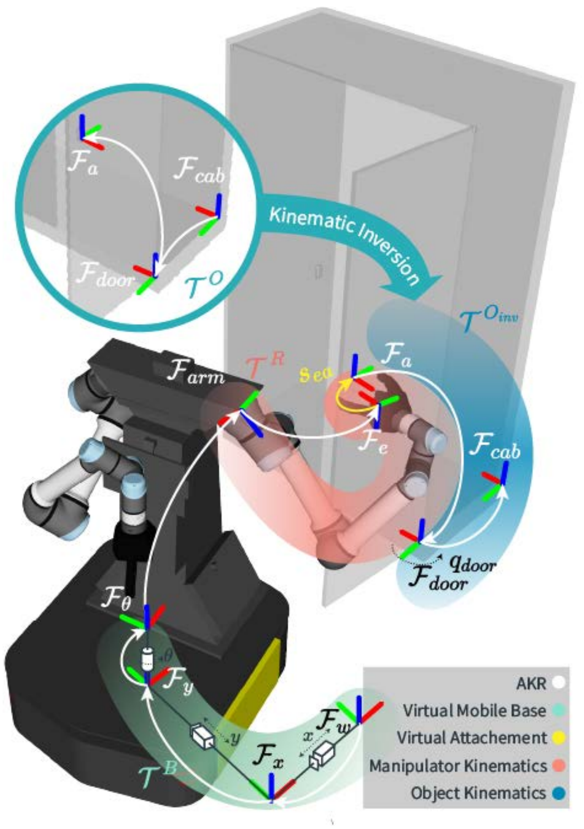


Fig. 2: **Modeling a mobile manipulation task from the proposed SMMP perspective.** Constructing an AKR involves four key inputs: the manipulator's kinematics $\mathcal{T}^R$, object kinematics $\mathcal{T}^O$, a virtual mobile base $\mathcal{T}^B$, and a virtual attachment joint s_{ea}. Given the articulated nature of the drawer, its kinematic model requires inversion to maintain a tree.

AKR modeling for a mobile manipulation planning problem first involves integrating the virtual mobile base into the kinematic model of the manipulator. During interactions, the post-inversion object's kinematics is further integrated into the AKR through a virtual attachment joint. The AKR is extended to the inverted object model's terminal link while maintaining its serial chain structure (see Fig. 2). In our application, we assume that $\mathcal{T}^R$ and $\mathcal{T}^O$ are known. However, obtaining the virtual mechanisms and constructing $\mathcal{T}^A$ involve more nuanced operations. The following section will detail these operations.

Virtual mobile base $\mathcal{T}^B$ reflects the motion possibilities of the mobile base. In Fig. 2, the manipulator with an omnidirectional mobile base can theoretically achieve free, stable motion on the ground plane. Consequently, a kinematic chain with three consecutive joints (two perpendicular prismatic joints are connected in serial to imitate linear motion, followed by one revolute joint at the rotation center of the mobile base to imitate angular motion) is sufficient to describe the motion of the base.

Virtual attachment joint s_{ea} characterizes the motion constraints and spatial relation between the robot and the scene after integration. As shown in Fig. 2, by inserting the s_{ea} between the manipulator's end-effector link v_e and an attachable link v_a in the object model, the kinematics of the mobile manipulator and the manipulated object are integrated. If a manipulated object $\mathcal{T}^O$ is articulated and the attachable

Algorithm 1: Kinematics Inversion

```
Input  : The kinematics: $\mathcal{T} = (V, S)$,
         The root node of $\mathcal{T}$: $v_r$,
         The attachable link node: $v_a$.
         (not necessarily the terminal node)
Output : The inverted kinematics: $\mathcal{T}^{\text{inv}} = (V, S^{\text{inv}})$.
1  // Initialization
2  $S^{\text{inv}} \leftarrow \{\}$;
3  // Get kinematic chain from $v_r$ to $v_a$
4  $(V_{ra}, S_{ra}) \leftarrow FindPath(\mathcal{T}, v_r, v_a)$;
5  // Inversion of the kinematic chain
6  foreach $\{s_{ij}, s_{jk}\} \subset S_{ra}$ do
7      $(r_{ij}, {}^{i}_{j}T) \leftarrow s_{ij}$, $(r_{jk}, {}^{j}_{k}T) \leftarrow s_{jk}$;
8      if $v_k$ is equal to $v_a$ then
9          $s^{*}_{ji} \leftarrow (r_{ji}, {}^{j}_{k}T^{-1})$;
10         $s^{*}_{kj} \leftarrow (r_{kj}, I_4)$;
11         $S^{\text{inv}} \leftarrow S^{\text{inv}} \cup \{s^{*}_{ji}, s^{*}_{kj}\}$;
12     else
13         $s^{*}_{ji} \leftarrow (r_{ji}, {}^{j}_{k}T^{-1})$;
14         $S^{\text{inv}} \leftarrow S^{\text{inv}} \cup \{s^{*}_{ji}\}$;
15 // Inversion of branches
16 foreach $v_j \in V_{ra}$ and $v_j \neq v_r$ do
17     foreach $s_{jk} \in S$ and $s_{jk} \notin S_{ra}$ do
18         $(r_{jk}, {}^{j}_{k}T) \leftarrow s_{jk}$;
19         $(r_{ij}, {}^{i}_{j}T) \leftarrow s_{ij}$, where $s_{ij} \in S^{\text{inv}}$;
20         $s^{*}_{jk} \leftarrow (r_{jk}, {}^{i}_{j}T^{j}_{k}T)$;
21         $S^{\text{inv}} \leftarrow S^{\text{inv}} \cup \{s^{*}_{jk}\}$;
22 foreach $v_i \in V$ and $v_i \notin V_{ra}$ do
23     if $\exists s_{ij} \in S$ then
24         $S^{\text{inv}} \leftarrow S^{\text{inv}} \cup \{s_{ij}\}$;
25 // Get the inverted kinematics
26 $\mathcal{T}^{\text{inv}} \leftarrow (V, S^{\text{inv}})$;
```

link is not the root node of $\mathcal{T}^O$, its kinematic model must be inverted before integration to ensure that $\mathcal{T}^A$ remains a tree (*i.e.*, each node within a tree has at most one parent node).

Kinematics inversion process reverses the kinematic model of the manipulated object while retaining its motion constraints and geometric consistencies, as shown in Alg. 1. Our kinematic tree representation defines transformations from parent to child link frames, with motion constraints (*i.e.*, joints) specified relative to the child frame. Therefore, kinematics inversion requires non-trivial adjustments to each joint's spatial transformation, in addition to simple parent-child inversions, since joints constrain child link motion relative to their local frame. The algorithm first identifies the main branch (Line 4) between the base link (the root node of $\mathcal{T}$) and the attachable link (identified in s_{ea}), including all intermediate joints and nodes. Transformations along this kinematic chain (between v_r and v_a) are then updated (Lines 6-14). The motion planner treats side branches as static, but their proper geometric transformation (Lines 16-24) remains critical for maintaining self-collision avoidance in the AKR representation. Fig. 2 illustrates the post-inversion cabinet kinematics and its integration into the AKR.

From the AKR perspective, we can formulate a *single-step mobile manipulation* task as motion planning in A-Space (*i.e.*, the configuration space of AKR), with task execution represented by AKR state transitions. Unlike decoupled approaches that treated the object as task-specific constraints imposed on the robot, *e.g.*, [50], the AKR simultaneously incorporates: 1) kinematic constraints for both robot and manipulated object, 2) path constraints for end-effector during interaction, and 3) self-collision avoidance—enabling generation of safe, coordinated base-arm-object motions. By generalizing to objects with known kinematics, the AKR eliminates task-specific modeling requirements for diverse objects and environments. This approach achieves effective whole-body motion optimization by eliminating the need for iterative backtracking in base-arm-object coordination at the task level.

IV. Planning in the A-Space

In this section, we first formulate motion planning problems for *single-step* mobile manipulation tasks in A-Space, and solve them via warm-started trajectory optimization. Then, we tackle the *multi-step* SMMP problem through an AKR-based task planning, supporting the generation of whole-body trajectories for interacting with multiple objects sequentially, implemented via three action predicates.

A. Motion Planning in A-Space

Consider a standard single-arm mobile manipulation task, in which a mobile manipulator interacts with an articulated object within the scene. The state vector $\boldsymbol{q}^\top = [\boldsymbol{q}^B, \boldsymbol{q}^R, \boldsymbol{q}^O]^\top \in \mathcal{Q}^{\text{free}}$ describes the state of the virtual mobile base $\mathcal{T}^B$, the manipulator $\mathcal{T}^R$, and the articulated object $\mathcal{T}^O$, respectively. Notably, these joints belong to a serial kinematic chain $\mathcal{C}$, which consists of a root node v_w and a non-root node v_b, as illustrated in Fig. 2. The remaining joints that do not belong to $\mathcal{C}$ are assumed to be fixed during motion planning. $\mathcal{Q}^{\text{free}} \subset \mathbb{R}^n$ is the collision-free subset of A-Space. The motion planning problem in A-Space is equivalent to finding a T-step path $\boldsymbol{q}_{1:T} = \langle \boldsymbol{q}_1, \boldsymbol{q}_2, \dots, \boldsymbol{q}_T \rangle \in \mathcal{Q}^{\text{free}}$, which can be formulated and solved by trajectory optimization.

Following Jiao *et al.* [1], the trajectory optimization problem is formulated as:

$$\underset{\boldsymbol{q}_{1:T}}{\text{minimize}} \sum_{t=1}^{T-1} ||\boldsymbol{W}_v \delta \boldsymbol{q}_t||_2^2 + \sum_{t=2}^{T-1} ||\boldsymbol{W}_a \delta \dot{\boldsymbol{q}}_t||_2^2, \tag{1}$$

$$h_{\text{chain}}(\boldsymbol{q}_t) = 0, \ \forall t = 1, 2, \dots, T, \tag{2}$$

$$||f_{\text{task}}(\boldsymbol{q}_T) - \boldsymbol{g}_{\text{goal}}||_2^2 - \xi_{\text{goal}} \leqslant 0, \tag{3}$$

where Eq. (1) penalizes the overall traveled distance and overall non-smoothness of the trajectory $\boldsymbol{q}_{1:T}$. $\boldsymbol{W}_v$ and $\boldsymbol{W}_a$ are two diagonal weighting matrices for each DoF, $\delta \boldsymbol{q}_t$ and $\delta \dot{\boldsymbol{q}}_t$ are the finite forward difference and second-order finite central difference of $\boldsymbol{q}_t$, respectively. The equality constraint Eq. (2) specifies the physical constraints of the object or the environment during interactions. Failing to account for this type of constraint (*e.g.*, the kinematic constraint of the robot and the scene) may damage the robot or the manipulated object, resulting in failed executions. The goal of a mobile manipulation task is bounded through an inequality constraint Eq. (3) with a tolerance ξ_{goal}. The function $f_{\text{task}} : \mathbb{R}^n \rightarrow \mathbb{R}^k$ maps $\boldsymbol{q}_T$ from the configuration space $\mathcal{Q}$ to the task-dependent goal space $\mathcal{G} \in \mathbb{R}^k$. For instance, in an object-picking task, f_{task} represents the forward kinematics used to compute the robot's end-effector pose, while $\boldsymbol{g}_{\text{goal}}$ denotes the end-effector goal pose before grasping. In a door-opening task, f_{task} maps the AKR state to the door's joint configuration, and $\boldsymbol{g}_{\text{goal}}$ represents the desired joint angle for the door.

Additional safety constraints are imposed during trajectory optimization. Without loss of generality, we assume an omnidirectional base and only kinematic constraints in this paper. However, additional constraints, such as nonholonomic constraints for non-omnidirectional mobile bases could be formulated into the optimization problem by incorporating additional terms [51]:

$$\boldsymbol{q}_{\text{min}} \leqslant \boldsymbol{q}_t \leqslant \boldsymbol{q}_{\text{max}}, \quad \forall t = 1, 2, \ldots, T \tag{4}$$

$$||\delta \boldsymbol{q}_t||_\infty \leqslant \delta \boldsymbol{q}_{\text{max}}, \quad \forall t = 1, 2, \ldots, T-1 \tag{5}$$

$$||\delta \dot{\boldsymbol{q}}_t||_\infty \leqslant \delta \dot{\boldsymbol{q}}_{\text{max}}, \quad \forall t = 2, 3, \ldots, T-1 \tag{6}$$

$$\sum_{i=1}^{N_{\text{link}}} \sum_{j=1}^{N_{\text{obj}}} |\text{dist}_{\text{safe}} - sd(L_i, O_j)|^+ \leqslant \xi_{\text{dist}}, \tag{7}$$

$$\sum_{i=1}^{N_{\text{link}}} \sum_{j=1}^{N_{\text{link}}} |\text{dist}_{\text{safe}} - sd(L_i, L_j)|^+ \leqslant \xi_{\text{dist}}, \tag{8}$$

where $|\cdot|^+$ is defined as $|x|^+ = \max(x, 0)$. Eqs. (4) to (6) are inequality constraints that define the joint capability and implicitly constrain the workspace of both the robot and the scene. Eq. (7) and Eq. (8) penalize collisions with obstacles and self-collisions, respectively. N_{link} and N_{obj} are the number of links that belong to the AKR and the number of obstacle objects within the scene, respectively. $\text{dist}_{\text{safe}}$ is a predefined safety distance, and $sd(\cdot)$ is a function that calculates the signed distance between a pair of objects. ξ_{dist} is a collision tolerance parameter. The formulated problem is solved through trajectory optimization [52].

Unlike sampling-based methods, optimization-based motion generation methods rely on gradient descent algorithms and can easily become trapped in undesired local minima near the initial guess [52, 53]. Consequently, a proper trajectory initialization (*i.e.*, warm start) is essential to improve the optimization results. However, the high dimensionality of AKR presents significant challenges. While sampling-based methods can provide paths as initialization seeds for optimization, they become computationally expensive in high-dimensional AKR spaces. Simple interpolation between start and goal states is insufficient, as base movements are often constrained by cluttered obstacles. Solving for coordinated movements simultaneously creates a complex optimization landscape with many poor local minima, making convergence difficult without good initialization. Therefore, an efficient initialization strategy that balances computational cost with solution quality is crucial for making AKR-based planning practical.

Therefore, we devise an A*-based trajectory initialization method to effectively guide trajectory optimization away from poor local minima without requiring excessive computational time. Given the initial state $\boldsymbol{q}_i$ and the goal state $\boldsymbol{q}_g$, this method utilizes A* to find a feasible path from the current location $\boldsymbol{q}_i^B$ to the goal location $\boldsymbol{q}_g^B$ of the mobile base. Subsequently, it linearly interpolates the manipulator's joint state from $\boldsymbol{q}_i^R$ to $\boldsymbol{q}_g^R$. While the method itself is presented as a simple design, it is pivotal to our framework's practical efficacy. The initialization phase aims to generate coarse, collision-free base paths (via $A^\star$) to guide subsequent trajectory optimization. Without this step, the solver often converges to local minima—for example, favoring shorter but colliding base paths over safer, longer ones. Appendix A provides quantitative comparisons with baselines, effectiveness analysis, and discussions of trade-offs and limitations.

B. Task Planning for Sequential Tasks

To solve a SMMP problem, a robot must break it down into a sequence of temporally feasible actions, necessitating task planning. Following the classic formalization of task planning, we describe the environment by a set of states $\mathcal{E}$ (of note, $\mathcal{E}$ and $\mathcal{Q}$ are unnecessarily identical). Possible transitions between these states are defined by $\mathcal{A} \subseteq \mathcal{E} \times \mathcal{E}$, where a transition $a = \langle e, e' \rangle \in \mathcal{A}$ alters the environment state from $e \in \mathcal{E}$ to $e' \in \mathcal{E}$. The task planning goal is to identify a sequence of transitions $a_{1:N}$ that alter the environment from its initial state $e_0 \in \mathcal{E}$ to a goal state $e_N \in \mathcal{E}_g$, where $\mathcal{E}_g \subseteq \mathcal{E}$ is a set of goal states. Traditional task planning involves defining meaningful symbolic actions $\mathcal{A}$ and states $\mathcal{E}$ and often assumes a robot can execute the elementary actions. However, these symbolic actions necessitate substantial manual design effort to be instantiated successfully at the motion level. From the AKR perspective, the action is defined as changes to the AKR structure and corresponding A-Space, transforming the SMMP problem into a series of AKR structural modifications.

In this section, we describe how `actions` defined using standard planning language (*e.g.*, PDDL [41]) can be used to properly formulate a task planning problem, and how the planned actions sequence can be realized by motion planning within A-Space. We start by making connections between the action semantics and the actual manipulation behaviors, before explaining how motion planners process the predicates and variables in the action definitions.

goto-akr (akr, $\boldsymbol{q}_1$, $\boldsymbol{q}_2$): This predicate moves the A-Space state from pose $\boldsymbol{q}_1$ to the desired pose $\boldsymbol{q}_2$. It represents the tasks that do not require interaction with the environment, wherein the AKR structure remains unchanged. Pure navigation is a typical action that falls into this category.

pick-akr (akr, o, s): This predicate moves the AKR to an object, `o`, with kinematics $\mathcal{T}$ and extends the current AKR's kinematics, by adding a virtual attachment joint `s`$\leftarrow s_{ea}$ to connect the object and the arm's end-effector. In practice, s_{ea} encodes both the end-effector's grasping pose and the associated grasp constraints between the robot and the object. `pick-akr` represents the group of tasks that require mobile manipulators to interact with the environment, *e.g.*, picking up an object or grasping a handle.

place-akr (akr, o, g): This predicate moves the object, `o`, connected to `akr` to an object-specific goal state `g`, while the object to be manipulated is incorporated into the AKR and imposes kinematic constraints. For example, `g` represents the target door state (*e.g.*, `opened`) in door-opening tasks or the desired object placement location (*e.g.*, `onTable`) in object relocation tasks. Once the goal state is reached, `place-akr` breaks the current AKR at the virtual attachment joint where it connects the mobile manipulator and the object, and the object will be placed where it was disconnected from the AKR. `place-akr` represents the group of tasks for which mobile manipulators stop interacting with the environment, such as placing an object on the table.

The primary challenge in generalizing actions across objects stems from heterogeneous task-specific constraints tied to scenes and objects, which is pivotal for generating executable trajectories in different mobile manipulation tasks. By embedding these constraints into scene kinematics, the AKR achieves a unified action definition and enables a general formulation of trajectory optimization for both rigid and articulated objects with known kinematics. This AKR-based formulation alleviates the need to define task-specific actions for manipulating different objects, which in turn reduces the need for intermediate subgoals (*e.g.*, moving the mobile base near the object before manipulation) and allows more dexterous exploration of the A-Space.

V. Sequential Mobile Manipulation Planning

In this section, we first present the operation of the plan refinement algorithm. We then describe how the algorithm resolves motion infeasibility in sequential tasks by selecting favorable action parameters through a goal selection process.

A. Plan Refinement for Sequential Tasks

To illustrate how symbolic action predicates (as defined in Sec. IV-B) govern the evolution of the AKR structure and how plan refinement resolves motion feasibility, consider the example in Fig. 1(a). The robot must first relocate a chair blocking a door (T1) and then open the door to exit (T2), interacting with two articulated objects. The action sequence includes four steps as shown in Fig. 3.

The AKR evolving with each action depicts possible end states for each step. The first action, `pick-akr`, generates a whole-body motion for the `akr` (virtual mobile base and manipulator) to grasp the chair. After grasping, `akr` integrates the chair's kinematics into a new `akr`. The resulting A-Space captures the kinematics of the mobile base, manipulator, and chair, while enforcing a planar constraint on the new `akr`'s end-effector (*i.e.*, the chair's base link) to emulate the chair's planar motion across the floor. These constraints are collectively considered during trajectory optimization.

For `place-akr`, the robot must choose a chair placement that avoids blocking subsequent door access. We sample valid configurations within A-Space and illustrate two representative configurations (second column in Fig. 3(c)). Without considering future actions, both placements are acceptable, as the chair no longer obstructs the door. After placing the chair, the `akr` detaches it, reverting to the mobile manipulator. The subsequent `pick-akr` action finds the path to approach the door handle. By sampling valid configurations (third column in Fig. 3), we can distinguish between motion-infeasible (red) and motion-feasible (green) pairs. Due to the presence of motion-infeasible pairs, plan refinement becomes necessary.

The plan refinement process acts as a receding horizon, evaluating feasibility across the action sequence. For instance, the last configuration in Fig. 3(c) (fourth column) is infeasible due to the subsequent action that approaches the drawer, which requires the robot to pass through the door. This example demonstrates how previously defined actions govern the evolution of the AKR structure and underscores the necessity of plan refinement in resolving motion feasibility. The subsequent section formalizes this procedure, focusing specifically on the selection of action parameters.

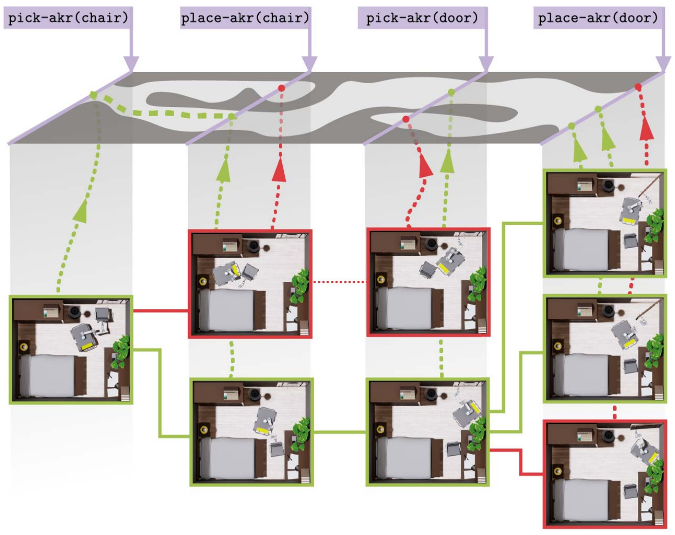


Fig. 3: **An illustration of the proposed plan refinement algorithm for a sequential task.** The task planner first generates a sequence of symbolic actions representing the evolution of the AKR configuration space, indicating robot-environment interactions. Then, the plan refinement algorithm ensures motion feasibility among sampled AKR end configurations for consecutive actions.

B. Action Parameter Selection from Key Configuration Set

The AKR-based motion planner requires two sets of action parameters to generate trajectories. The first, end-effector poses `s`, specifies grasps between the robot's end-effector and objects during `pick-akr` actions. These poses are obtained through grasp synthesis methods (*e.g.*, [54]) or predefined for known objects, as grasp generation lies beyond the scope of this work. The second set, `g`, defines object-centric states aligned with symbolic predicates, such as an `opened` door or an object placement goal like `onTable`, and must be instantiated appropriately within the object's configuration space for trajectory optimization [55].

Improper action parameters values can lead to motion infeasibility, as they do not fully capture the state of the `akr`, and variations in `akr` states impose different feasibility conditions on subsequent actions, as illustrated in Fig. 3. While multiple chair placements are feasible in a bedroom, some configurations (marked red) obstruct the robot from approaching the door due to self-blocking. This challenge is exacerbated in confined spaces with limited configuration space. To ensure feasibility across action sequences, our method jointly optimizes goal states with future steps, resolving conflicts during plan refinement to avoid such pitfalls.

The exhaustive motion planning for all possible action parameters is computationally demanding, with time complexity growing exponentially with action sequence length, rendering it impractical. To address this challenge, we propose a plan refinement algorithm designed to efficiently select the goal AKR state by considering a given number of anticipated subsequent actions. This approach aims to improve the likelihood of success for sequential tasks.

Specifically, let $\boldsymbol{q}_{a_n}$ be a possible goal AKR configuration for the action a_n and $\mathcal{Q}_{a_n}$ be the A-Space during that action, and $\mathcal{Q}_{a_{n:n+l}}$ be the Cartesian product of A-Spaces: $\mathcal{Q}_{a_{n:n+l}} = \mathcal{Q}_{a_n} \times \mathcal{Q}_{a_{n+1}} \times \ldots \times \mathcal{Q}_{a_{n+l}}$, where l is the

Algorithm 2: Select KCS

```
Input  : Action sequence segment: $a_{n:n+l}$
         Current AKR: $\mathcal{T}^A_{a_{n-1}}$
         Current AKR state: $\boldsymbol{q}_{a_{n-1}}$
Ouput  : Preferred key configuration set: $\boldsymbol{q}^*_{a_{n:n+l}}$
Params: No. of candidate configurations: $N_c$
        No. of clusters: $N_k$
        No. of anticipated subsequent actions: $l$
1  $\boldsymbol{q}_{a_{n-1:n-1}} \leftarrow \langle \boldsymbol{q}_{a_{n-1}} \rangle$
2  $K \leftarrow \{\boldsymbol{q}_{a_{n-1:n-1}}\}$
3  for $i \in n : n+l$ do
4      $K_{temp} \leftarrow \varnothing$
5      // Update A-Space according to Sec. IV-B.
6      $\mathcal{T}^A_{a_i} \leftarrow ConstructAKR(\mathcal{T}^A_{a_{i-1}}, a_i)$
7      // Generate valid configurations within
        A-Space.
8      $Q_{a_i} \leftarrow SampleValidConfigurations(\mathcal{T}^A_{a_i}, N_c)$
9      // Pruning similar configurations through
        down-sampling.
10     $Q'_{a_i} \leftarrow Downsample(Q_{a_i}, N_k)$
11     // Predict and store feasible KCS
12     for $\boldsymbol{q}_{a_{n-1:i-1}} \in K$ do
13         for $\boldsymbol{q}_{a_i} \in Q'_{a_i}$ do
14             if $CheckMotionFeasibility(\mathcal{T}^A_{a_i}, \boldsymbol{q}_{a_{n-1:i-1}}, \boldsymbol{q}_{a_i})$
                then
15                 $\boldsymbol{q}_{a_{n-1:i}} \leftarrow \boldsymbol{q}_{a_{n-1:i-1}}.append(\boldsymbol{q}_{a_i})$
16                 $K_{temp} \leftarrow K_{temp} \cup \{\boldsymbol{q}_{a_{n-1:i}}\}$
17     $K \leftarrow K_{temp}$
18 // Select KCS of lowest cost
19 $\boldsymbol{q}^*_{a_{n-1:n+l}} \leftarrow SelectBest(K)$
20 $\boldsymbol{q}^*_{a_{n:n+l}} \leftarrow \boldsymbol{q}^*_{a_{n-1:n+l}} \backslash \{\boldsymbol{q}_{a_{n-1}}\}$
```

window length suggesting the number $(l+1)$ of anticipated subsequent actions. Our aim is to find a Key Configuration Set (KCS) $\boldsymbol{q}^*_{a_{n:n+l}} = \langle \boldsymbol{q}_{a_n}, \boldsymbol{q}_{a_{n+1}}, \dots, \boldsymbol{q}_{a_{n+l}} \rangle \in \mathcal{Q}_{a_{n:n+l}}$ so that transition among every two consecutive configurations is valid and efficient.

Alg. 2 details the process. The algorithm takes three inputs: 1) a segment of the action sequence $a_{n:n+l}$, 2) the current AKR structure $\mathcal{T}^A_{a_{n-1}}$, and 3) the current AKR state $\boldsymbol{q}_{a_{n-1}}$. Parameters include N_c, the number of candidate configurations sampled per action; N_k, the number of clusters for down-sampling; and l, the horizon length for anticipated actions. The subsequent paragraphs detail phases of the workflow.

A-Space construction (line 6): For each action a_i in the sequence $a_{n:n+l}$, the algorithm first updates the AKR structure $\mathcal{T}^A_{a_i}$ by integrating the kinematics of the manipulated object (*e.g.*, a door or chair) into the robot's kinematics, as detailed in Sec. III-B. This constructs the A-Space, which encodes the combined configuration space of the robot and object.

Configuration sampling (line 8): We define $Q_{a_i} \subset \mathcal{Q}_{a_i}$ as the finite set of sampled configurations for AKR $\mathcal{T}^A_{a_{i-1}}$, where each $\boldsymbol{q}_{a_i} \in Q_{a_i}$ satisfies task-specific goal constraints and collision-free conditions. To fully explore possible goal configurations, we formulate an optimization problem:

$$\min_{\boldsymbol{q}_{a_i}} \quad ||h_{\text{chain}}(\boldsymbol{q}_{a_i})||_2^2 + ||f_{\text{task}}(\boldsymbol{q}_{a_i}) - \boldsymbol{g}_{a_i}||_2^2 \tag{9}$$
$$\text{s.t.} \quad ||h_{\text{chain}}(\boldsymbol{q}_{a_i})||_2^2 \leqslant \xi_{\text{chain}} \tag{10}$$
$$||f_{\text{task}}(\boldsymbol{q}_{a_i}) - \boldsymbol{g}_{a_i}||_2^2 \leqslant \xi_{\text{goal}} \tag{11}$$
$$\boldsymbol{q}^{\min} \leqslant \boldsymbol{q}_{a_i} \leqslant \boldsymbol{q}^{\max} \tag{12}$$
$$\boldsymbol{q}_{a_i} \in \mathcal{Q}^{\text{free}}_{a_i} \tag{13}$$

Algorithm 3: SampleValidConfigurations

```
Input  : An AKR: $\mathcal{T}^A_{a_i}$
Ouput  : The Set of Valid Configurations: $Q_{a_i}$
Params: Max. Cardinality of $Q_{a_i}$: $|Q_{a_i}|^{max}$
        Max. Tries of IK Calculation: MAX_TRIES
1  $Q_{a_i} \leftarrow \varnothing$
2  $counts \leftarrow 0$
3  while $|Q_{a_i}| < |Q_{a_i}|^{max}$ or $counts <$MAX_TRIES do
4      $\boldsymbol{q}_{a_i} \leftarrow computeIK(\mathcal{T}^A_{a_i})$ //w.r.t. Eq. (9)
5      if $\boldsymbol{q}_{a_i}$ satisfy Eqs. (10) to (13) then
6          $Q_{a_i} \leftarrow Q_{a_i} \cup \{\boldsymbol{q}_{a_i}\}$
7      $counts$++
```

where Eq. (9) penalizes the violation of the environment and the goal constraint corresponding to Eqs. (2) and (3), Eqs. (10) and (11) bound the objective with a small tolerance to reduce undesirable results, Eq. (12) constrains the $\boldsymbol{q}_{a_i}$ to be within the joint limit of the AKR, including both the robot and the manipulated object, Eq. (13) ensures $\boldsymbol{q}_{a_i}$ is collision-free in the environment. Alg. 3 details how to solve the problem to generate a finite Q_{a_i} for a_i. We first randomly sample goal AKR configurations based on the constructed AKR $\mathcal{T}^A_{a_i}$ from a uniform distribution to initialize $computeIK$ and compute the inverse kinematics problem (*i.e.*, Eq. (9)) numerically on $\mathcal{T}^A_{a_i}$. Solutions satisfying Eqs. (10) to (12) are added to Q_{a_i} until reaching its maximum cardinality. Then, we prune out configurations that are in collisions (*i.e.*, violating Eq. (13)). Note that the collision check is reserved until the last step because collisions frequently happen in a confined and cluttered environment, and checking collisions is computationally heavy.

Configuration down-sampling (line 10): Even after discarding configurations that are in collision, the candidate set Q_{a_i} often remains large. This can make motion feasibility checking computationally expensive due to the combinatorial nature of validating transitions across a sequence of actions—requiring up to $|Q_{a_n}| \times |Q_{a_{n+1}}| \times \ldots \times |Q_{a_{n+l}}|$ checks. While retaining all candidates helps preserve completeness, in practice, down-sampling the configuration set can significantly improve planning efficiency by reducing the number of costly, repetitive feasibility checks. Thus, we down-sample configurations based on the assumption that those located close to each other in configuration space (*i.e.*, with similar joint values and small Euclidean distances) exhibit similar motion feasibility. This is justified by the nature of our AKR-based trajectory optimization: Eq. (3) constrains only a subset of the AKR state variables via f_{task}. As a result, nearby configurations often converge to the same local minima during trajectory optimization, making it redundant to plan from each configuration individually.

To down-sample Q_{a_i} and avoid redundant computations for similar configurations, we use the k-means++ method [56] to partition Q_{a_i} into N_k clusters by minimizing the variance within the cluster: $Q_{a_i} = \{Q^1_{a_i}, \dots, Q^k_{a_i}\}$ with the corresponding cluster centroid $\bar{Q}^k_{a_i}$. Then we construct a downsampled set $Q'_{a_i} = \{\boldsymbol{q}'^1_{a_i}, \boldsymbol{q}'^2_{a_i}, \dots, \boldsymbol{q}'^k_{a_i}\}$ by selecting $\boldsymbol{q}'^k_{a_i}$ that is closest to the centroid in each cluster as the key configuration for the whole action sequence. Note that the cluster centroid itself may not be a valid configuration.

We acknowledge that the down-sampling step introduces some incompleteness. However, we wish to clarify that downsampling is primarily a practical strategy to improve efficiency, as further experiments in Sec. VI-D demonstrate. While the current implementation focuses on empirical performance, we believe that completeness can be achieved through a more sophisticated and structured sampling strategy, as suggested by previous work [47, 49, 55].

Feasibility Checking (line 12-17): As the above procedure produces a much more compact $Q'_{a_{n:n+l}}$, checking the motion feasibility among its elements becomes feasible. Specifically, $checkMotionFeasibility$ estimates the motion feasibility for $\langle \boldsymbol{q}_{a_i}, \boldsymbol{q}_{a_{i+1}} \rangle$ by applying the $A^\star$ algorithm (the map and base path are reused for trajectory initialization to reduce computational effort) to find a path between the mobile base poses encoded in key configurations. We will record the key configuration in K if there is a feasible base path. In the example shown in Fig. 3(c), the key configuration in red is removed because no viable path connects it to the upcoming action of grasping the door handle or passing through the door.

Optimal KCS Selection (line 19): The procedure iterates until all actions within horizon l are checked, resulting in the construction of K, which consists of feasible KCS. Subsequently, we employ an objective function to penalize the total traveling distance and select the best KCS $\boldsymbol{q}^*_{a_{n:n+l}}$ with minimal cost:

$$\min_{\boldsymbol{q}_{a_{n-1:n+l}}} \sum_{i=n}^{n+l} (||\boldsymbol{W}_R(\boldsymbol{q}^R_{i-1} - \boldsymbol{q}^R_i)||_2^2 + ||\boldsymbol{W}_B(\boldsymbol{q}^B_{i-1} - \boldsymbol{q}^B_i)||_2^2) \quad (14)$$

where $\boldsymbol{W}_R$ and $\boldsymbol{W}_B$ represent weight matrices, and the cost function exclusively penalizes the traveling distance for both the mobile base and the manipulator joints between two configurations.

VI. Simulation

This section presents the results from extensive simulations that evaluate the proposed SMMP framework. The simulations demonstrate the effectiveness of generating coordinated base-arm-object motion in the proposed A-Space, quantitatively compared to several baselines. Additionally, an object rearrangement task highlights the advantages of the AKR-based planning domain design in simplifying the task planning by reducing unnecessary action predicates of separately moving the mobile base and the arm. An ablation study on a complex, 18-step long-horizon SMMP task further examines the plan refinement algorithm.

A. Simulation Setup

The simulated mobile manipulator platform comprises a Clearpath Husky mobile base and a Universal Robot UR5e robotic manipulator equipped with a Robotiq 2-finger gripper positioned at the mobile base's rotation center. The mobile base is assumed to be omnidirectional during trajectory optimization. As the four wheels can be controlled independently, its trajectory is then processed by adjusting the orientation of the mobile base to match the direction of movement, and the shoulder joint of the manipulator is adjusted accordingly to ensure the correctness of the trajectory.

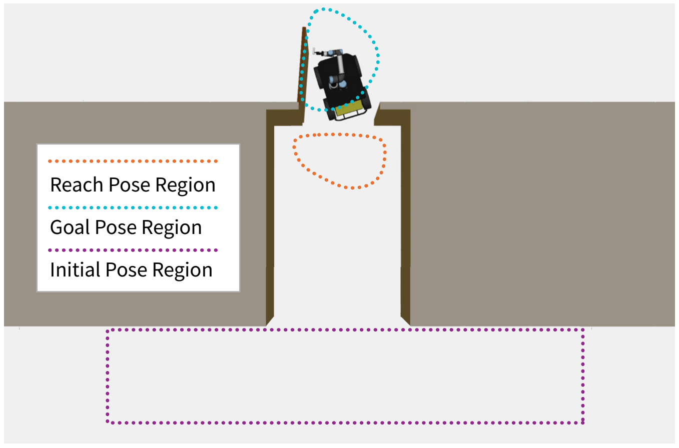
B. Comparisons with Baselines

We developed two mobile manipulation scenarios to evaluate the benefits of SMMP as compared to approaches that treat the base and arm separately. The first task, depicted in Fig. 4(a), is to approach a door and open it by pushing. The door has a single revolute joint and is located at the end of a corridor. The second task, illustrated in Fig. 4(b), involves reaching a drawer in a confined kitchen space and opening it by pulling its prismatic joint. The initial position of the robot is randomly selected from within the shaded purple region.

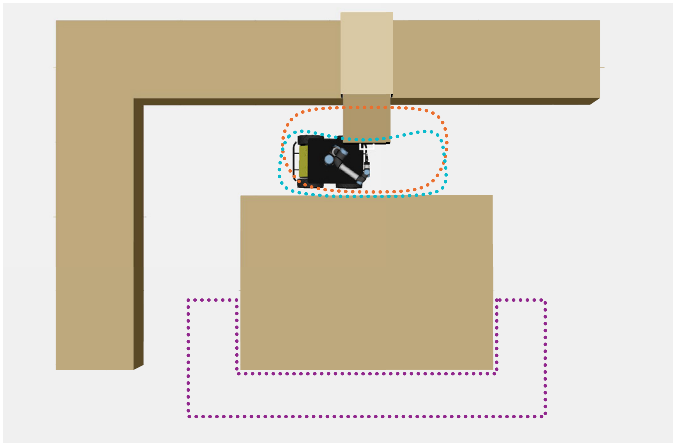
In addition to our SMMP framework, referred to as SMMP+TO (*i.e.*, trajectory optimization), we introduce three alternative setups to solve the above two mobile manipulation tasks for comparing the performance. To compare with a sampling-based constrained motion planner, we adopt the well-known RRT-Connect method [57] from the Open Motion Planning Library (OMPL) [58, 59] to solve the constrained motion planning problems formulated by SMMP, referred to as SMMP+RC. To compare the SMMP-based approaches with typical non-SMMP approaches, we introduce two additional baselines that independently compute trajectories for the mobile base and manipulator. Baseline 1 (BL1) utilizes A* to search for a feasible mobile base path and subsequently smooth through trajectory optimization. The arm pose is then determined by solving the inverse kinematics from the door handle to the mobile base at each way-point. Building upon BL1, Baseline 2 (BL2) further optimizes the poses of the manipulator and manipulated object at each way-point for collision avoidance.

Notably, our SMMP-based approaches (SMMP+TO and SMMP+RC) only needs to specify one task goal: the desired door angle or the desired drawer length to open. In contrast, non-SMMP approaches (BL1 and BL2) require specification of the pose of the mobile base when reaching the doorknob as well as after having opened the door, as the base and the manipulator are planned individually. We compute the mobile base's intermediate poses by sampling from feasible regions that are empirically determined (*i.e.*, for mobile base poses in this region, the existence of an arm pose to grasp the handle is guaranteed); see the orange areas in Fig. 4(a)(b) for reaching, and the blue areas for final poses.

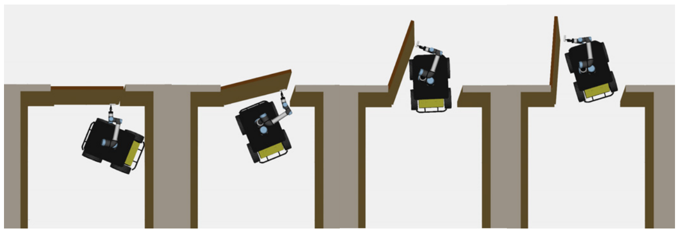
We evaluate the planning results using four criteria: (i) *success rate* as the percentage of task completion without violating constraints, (ii) the *base's effort* as the total base travel distance, (iii) the *arm's effort* as the sum of each joint's cumulative angular displacement throughout task execution, and (iv) the *planning time*. The results are summarized in Fig. 4(g). Planning the base, arm, and manipulated object separately (BL1) results in a success rate of 1% for opening doors and 36% for opening drawers. The primary cause of the failures in BL1 is collisions between the mobile manipulator and the door or drawer, as shown in Fig. 4(e)(f). Although implementing robot-object collision checks to refine motions (BL2) enhances the success rate to 51%, the proposed SMMP-based approaches still outperform the non-SMMP approaches. Failure cases of BL2 primarily arise from kinematic constraint violations, such as the end-effector disengaging from the handle, due to the absence of feasible IK solutions caused by the bulky mobile base. This highlights the necessity of



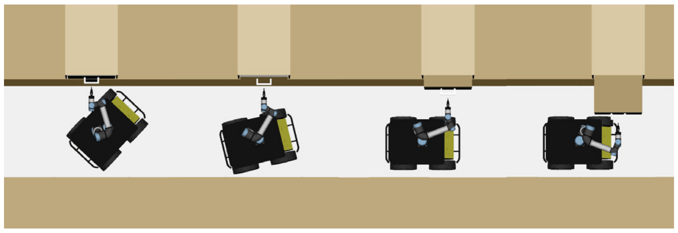
(a) Door opening task



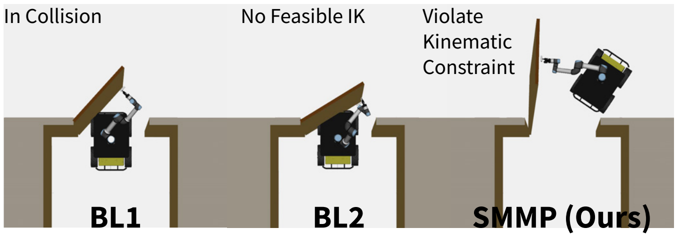
(b) Drawer opening task



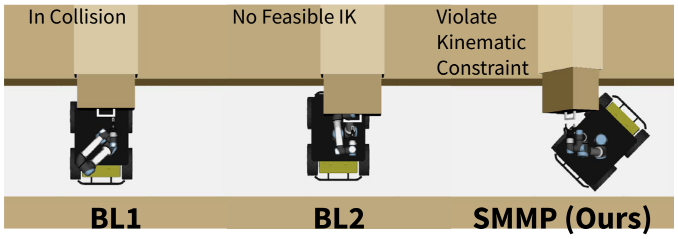
(c) Door opening task successful trail



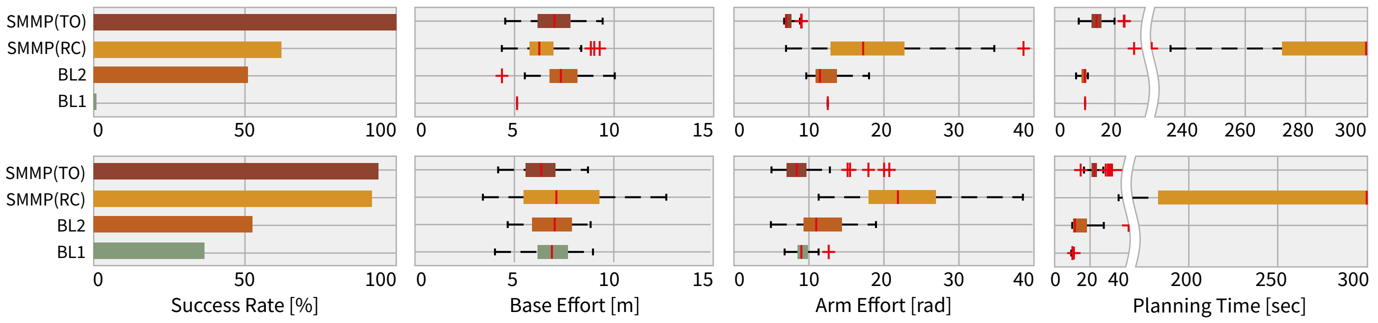
(d) Drawer opening task successful trail



(e) Door opening task failure cases



(f) Drawer opening task failure cases



(g) Experiment results. Upper: door opening task; lower: drawer opening task.

Fig. 4: **Quantitative comparisons between the SMMP and three baselines in door manipulation and drawer manipulation tasks.** The robot starts from a randomized location within the purple region and (a) opens the door to a specific angle or (b) pulls out the drawer by a specific length. The orange and blue regions indicate the feasible poses that must be given to the mobile base for baselines. The regions are empirically found to guarantee a valid IK solution for the robot. (c) and (d) are successful trials of two tasks, respectively, and (e) and (f) are typical failure cases of baseline methods and the proposed SMMP method. (g) The planning success rate and the box plots with kernel density plots of the base effort, arm effort, and planning time of the four methods in these two tasks.

simultaneous coordination of the base, arm, and object, as illustrated in the BL2 failures shown in Fig. 4(e)(f).

The SMMP+TO and SMMP+RC both generate feasible trajectories for the given task, with SMMP+TO achieving higher success rates than SMMP+RC. SMMP+TO produces more efficient trajectories in terms of shorter base and arm travel distances. Typically, sampling-based motion planners struggle to incorporate kinematic and safety constraints, necessitating extra effort to accommodate additional kinematic constraints [59]. Fig. 4(e)(f) also demonstrate failure cases of violating kinematic constraints. The typical failure mode of SMMP+RC is that the planner fails to find a feasible solution within the allowable time budget (300 seconds).

Our comparative experiments suggest that SMMP-based approaches are better suited for complex mobile manipulation tasks by jointly optimizes base–arm–object movements. Moreover, trajectory optimization proves more effective for solving SMMP-based motion planning problems due to the intricate constraints involved. However, this comes at the cost of increased planning time compared to non-SMMP baselines, as AKR introduces a higher DoF compared to robot kinematics.

C. Analysis on Efficiency Improvement in Task Planning

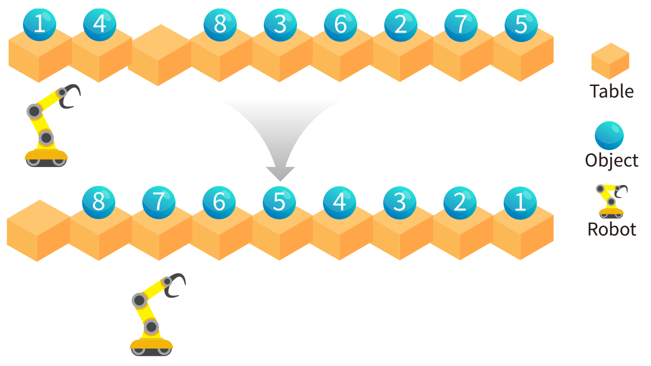
By treating the robot base, arm, and object to be manipulated as a whole, the design of the task planning domain based on the A-Space perspective can offer greater efficiency. We use an object-arrangement task as an example to quantitatively evaluate the improvement offered by the A-Space perspective, where the robot rearranges m objects on $m+1$ tables in a sorted order while satisfying the constraint that each table can support only one object. Fig. 5(a) shows a typical example of the initial and goal configuration of this task with $m = 8$ objects. Our PDDL implementation, built on top of AKR (see Appendix C for details), uses fewer predicates and supports more abstract action representations compared to domain definitions that decouple base and arm movements (*e.g.*, [48, 55]). Specifically, such separated base-arm domains typically require: (i) additional predicates to represent the mobile base's state, resulting in more state predicates; (ii) an extra action explicitly dedicated to base movement; and (iii) more action parameters for manipulation planning.

To produce a task plan, we adopt the PDDL solver from [60] which employs a hybrid strategy combining a Serialized Iterative Width (SIW) search-based planner and a Best First Search-based planner, BFS(f) [61]. We use PDDL version 2.2 throughout all task planning formulations presented in this paper. In this study, we ran 50 trials for each setup; see the results summarized in Fig. 5(b). As task complexity grows, the planning time and the number of nodes generated during the search (*i.e.*, memory usage) increase much more slowly using the AKR-based approach, as compared to separated base-arm domains, which exhibit an exponential rise. This is because non-AKR-based task planning requires more action operators to accomplish each task, resulting in greater search depths. If there are, on average, N nodes generated at each search depth level, and a solution is found at depth D, the total nodes generated is N^D. The AKR-based approach requires fewer action operators, resulting in shallower search depths and substantially reducing both the number of expanded nodes and the frequency of backtracking during task planning. In a task involving rearranging 16 objects across 50 trials, we observed an average of 76% reduction in search depth with the AKR-based task planner, leading to a significant enhancement in planning efficiency alongside a reduction in memory usage.

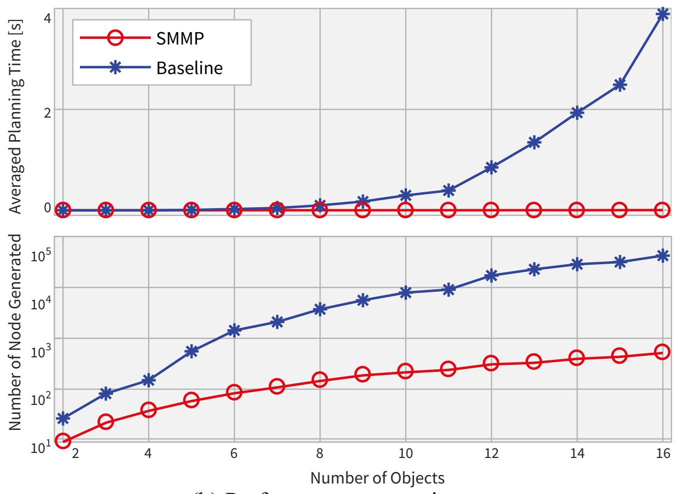
Taken together, the results of this study demonstrate that task planning based on AKR significantly reduces the need for wide and deep exploration in the search process. This improvement not only improves the efficiency and reduces memory usage of task planning, but also holds promise for reducing the number of motion planning calls required in broader TAMP frameworks.

D. Ablation Studies of Plan Refinement in SMMP

We conducted further ablation studies to evaluate how the plan refinement algorithm impacts the execution success rate in long-horizon tasks through simulation. In Fig. 6(a), the robot is assigned the task of bringing a drink from the fridge to the bedroom desk (Goal 1) and fetching trash located between the sofa and the coffee table before depositing it into the trash can (Goal 2). Notably, the robot has to temporarily place the drink on the dining table to free its gripper before opening the bedroom door ($\textcircled{a_4}$-$\textcircled{a_9}$), and must also use the broom to collect the trash because the space within which it is located is smaller than the robot's base ($\textcircled{a_{13}}$-$\textcircled{a_{17}}$). This poses significant challenges to the reliability of the generated 18-step task and motion plan. Please refer to Appendix C for details on the task planning domain designed for this problem.



(a) Experimental setup



(b) Performance comparison

Fig. 5: **AKR-based domain specification improves the task planning efficacy.** (a) An example setup of rearranging 8 objects on 9 tables; one table can only support one object. (b) The AKR-based domain specification allows a solver to search for a feasible plan for tasks involving re-arranging 2 to 16 objects in significantly less time while generating fewer nodes in search (*i.e.*, less memory).

In the study, we recorded the robot's cumulative planning success rate at each step in the sequential mobile manipulation over the 18 steps across 5 settings:

- `Non-SMMP`: The execution trajectories are generated using Baseline 2 (BL2) as described in Sec. VI-B, which plans the robot's base and arm separately without incorporating the proposed AKR.
- l=`0, w/o DS`: The end configuration for the robot's next action is randomly sampled, without employing plan refinement (as l=0) and down-sampling.
- l=`2, w/o DS`: The end configuration for the robot's next

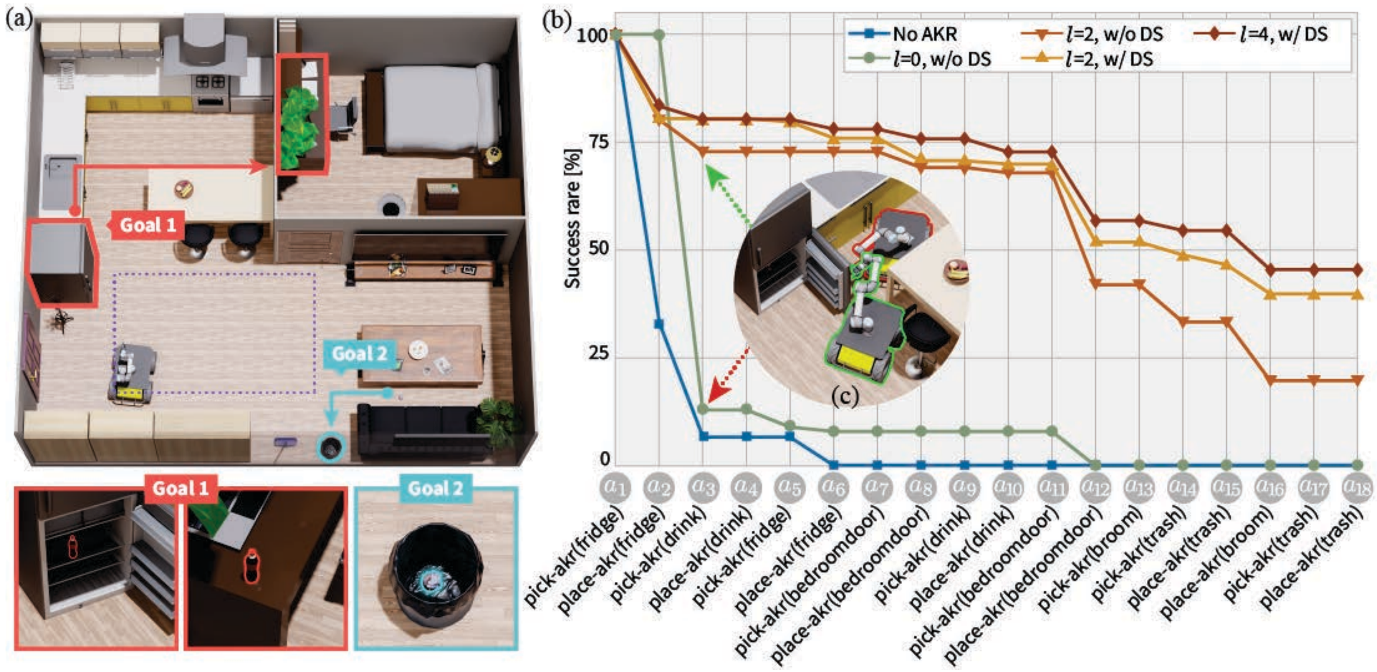


Fig. 6: **Ablation studies of plan refinement in SMMP with a simulated household environment.** (a) The robot is tasked to place a drink on the desk in the bedroom and dispose of an object in the trash can. (b) The entire task consists of 18 actions, all represented by the two symbolic actions. The proposed AKR against a baseline method and different settings of plan refinement. (c) Two possible robot start configurations that influence the feasibility of subsequent actions.

action is determined using the plan refinement algorithm *without down-sampling*. A total of 3 subsequent actions (including the current one) are considered.

- l=`2, w/ DS`: The end configuration is determined using the proposed plan refinement algorithm with *down-sampling enabled*. The same 3 subsequent actions (including the current one) are considered to allow direct comparison with the previous setup.
- l=`4, w/ DS`: A total of 5 subsequent actions (including the incoming action) are considered, with the other settings remaining the same as in the previous setting.

In each trial, the mobile manipulator is randomly positioned within the purple-dotted region depicted in Fig. 6(a); a total of 100 trials are conducted to obtain the cumulative success rate. To ensure a fair comparison, in addition to the initial and goal states of the environment, both the SMMP and non-SMMP methods received identical manually defined grasping poses for all movable objects, though not the corresponding robot configurations.

In Fig. 6(b), the cumulative success rates for each set, as the task progresses, underscore the importance of considering future actions in long-horizon tasks. Without planning in the A-Space, the motion planner faces particular challenges at (a_2), when opening the fridge door in a confined space. When no plan refinement algorithm is applied (*i.e.*, $l = 0$, w/o DS), the success rate significantly drops at a_3 because the opened fridge door and the kitchen table obstruct the robot's path to picking up the drink, as illustrated in Fig. 6(c). Without considering future actions, the robot could easily trap itself during execution in a crowded scene (*i.e.*, select a poor end configuration as in a_2). By anticipating the actions of picking up the drink and placing it in a_3 and a_4 (*i.e.*, $l = 2$, w/o DS), the robot avoids getting trapped by choosing the green end configuration instead of the red one, as in Fig. 6(c), despite this trajectory being less efficient and harder to compute at the current step, as indicated by a slight drop in motion planning success rate.

Furthermore, finding end configurations for each action using the down-sampling method improves the success rate by excluding robot configurations sharing similar geometric properties ($l = 2$, w/ DS *v.s.* $l = 2$, w/o DS), so it is more likely to find a feasible goal within the computational budget (max 5 retries are allowed for each action). Looking ahead to longer horizon tasks (*i.e.*, $l = 4$, w/ DS), one could further improve the success rate, albeit at the cost of increased computational effort. In summary, the plan refinement algorithm significantly improves the motion planning success rate by selecting action goals at each step that take into account the feasibility of future actions, making this method better suited to long-horizon tasks.

VII. Experiments

This section presents the real-world robot experiments that show that the framework can generate coordinated whole-body motions for mobile manipulation across various scenarios, solve challenging long-horizon tasks, and generalize to different robot platforms and tool-use tasks analogous to the body schema theorem.

A. Robot Platform

In this article, we evaluated the proposed SMMP method on three robot platforms with different structures.

The mobile manipulator platform consists of dual Universal Robot UR5e robotic manipulators equipped with Robotiq

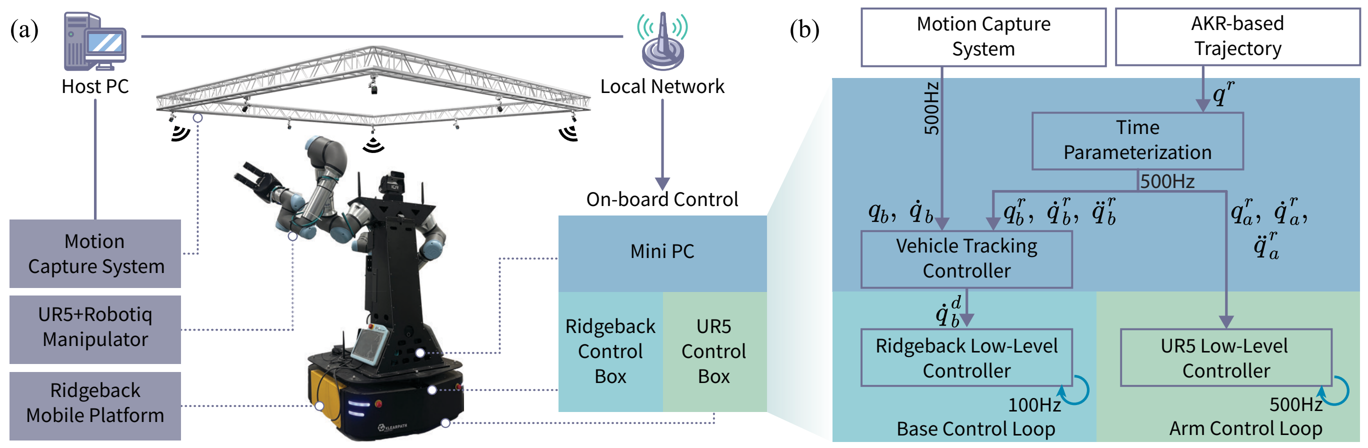


Fig. 7: **The system diagram for the mobile manipulator platform.** (a) The mobile manipulator's hardware configuration and communication diagram. (b) The control diagram of the mobile manipulator platform.

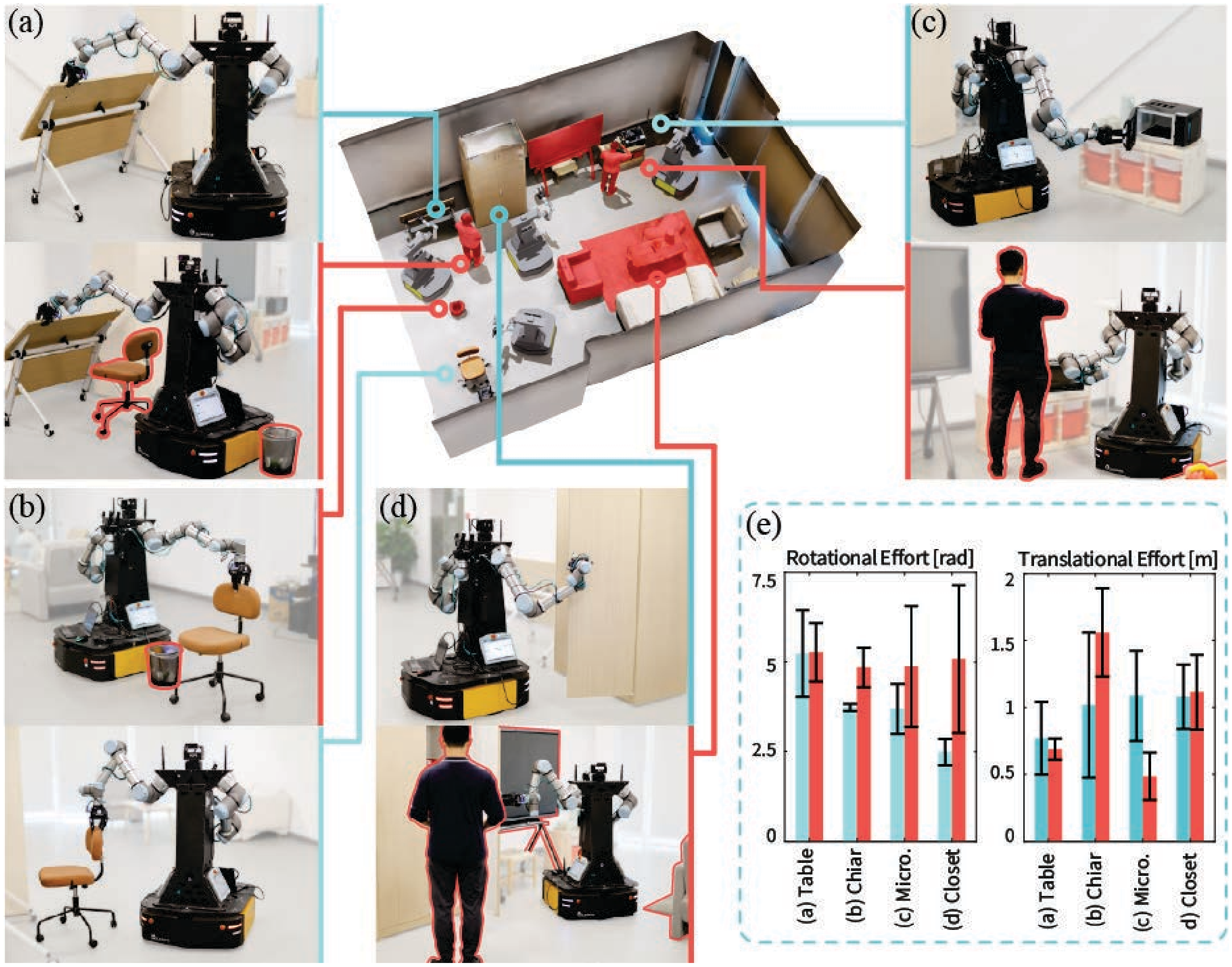


Fig. 8: **Robot performance in mobile manipulation with articulated objects.** The robot generates coordinated base-arm-object motions for (a) unfolding the table, (b) placing the chair, (c) opening the microwave, and (d) opening the closet. The robot consistently maintains coordination in manipulations, even in the face of more challenging situated constraints due to the obstacles shaded in red. (e) The translational and rotational traveling distance.

3-finger grippers installed on a Clearpath Ridgeback omnidirectional mobile platform. The on-board computational hardware is a mini PC with Intel Core i7-10700 CPU. Task and motion generation are performed on a host PC equipped with an AMD Ryzen9 5950X CPU. For perception, we utilize the Motion Capture System (MCS), operating at 500 Hz, to track object poses and the mobile base's position and orientation. MCS data is processed on its host PC and transmitted to the onboard mini PC via a local network. It is worth noting that a single UR5e manipulator serves all tasks in our setup.

Our on-board control processes involve several steps. Firstly, the MCS tracks the mobile base's 3D pose and objects' 6D poses in the physical environment, which are then transmitted to the host PC along with the manipulator configuration (UR5e joint values in this case) to update the AKR state. Next, the proposed SMMP framework updates the AKR structure according to the previous action and then generates the trajectory based on the current action goals and A-Space. Subsequently, the host PC sends the planned trajectory to the mini PC for time parameterization, adhering to hardware constraints. The time-parameterized reference trajectory is then sent to the corresponding base and arm controllers concurrently. For manipulator control, we utilize the built-in trajectory follower of the UR5e. For the mobile base control, we develop a custom PID-based velocity tracking controller to generate velocity control signals for trajectory tracking. Fig. 7 is a schematic diagram of the platform.

The aerial manipulator platform setup is similar to the mobile manipulator platform, the major difference being the flying vehicle control system. The high-level controller communicates with the MCS through Ethernet for feedback and outputs the desired attitude and thrust of each thrust generator [62]. These commands are transmitted through Crazy Radio PA antennas (2.4 GHz) to the Crazyflie 2.1 control boards, where double-loop PID controllers are implemented for 500 Hz low-level control with onboard IMU feedback. Appendix B provides more details of the system.

B. Coordinated Whole-body Trajectory Generation

In a real household environment featuring diverse everyday objects with distinct articulation, we showcase the robot's adept execution of various mobile manipulation tasks through coordinated whole-body motions generated by the proposed method. Snapshots in Fig. 8 depict the robot performing four typical household tasks: (a) unfolding a flip-top table with a horizontal revolute axis, (b) rolling a chair that can move around on the floor plane (*i.e.*, 2D displacement), (c) opening a microwave and (d) opening a closet, both involving a vertical revolute axis, with the bulky closet door requiring more sophisticated motion coordination. By formulating motion planning problems in A-Space one naturally accommodates both robot and object movements, resulting in successful and efficient task execution. This advantage is particularly evident in Fig. 8(e), in which each manipulation task is rendered increasingly complex by the presence of new obstacles (highlighted in red), and more sophisticated obstacle avoidance strategies, therefore, become necessary. Notably, for this kind of task, motion planning using in A-Space shares the same objectives and goal states, differing only in the constraint that specifies obstacle configurations in the surrounding space (see Sec. IV-A for details).

We further evaluated the motion planner's performance in terms of the efficiency of the trajectories it computes. By repeating the planning for each scenario in Fig. 8(a)-(d) five times (with random start locations) and executing the planned trajectories on physical robots, we reported statistics on base efforts and arm joint efforts measured by trajectory lengths in Fig. 8(e), to assess execution efficiency. In general, surrounding obstacles could constrain navigation, compelling the robot to compensate by increasing arm movements, and leading to notably higher joint effort when manipulating the chair, microwave, and closet.

C. Sequential Mobile Manipulation Planning

Fig. 9(a) showcases a series of the robot in a conducting complex, long-horizon sequential mobile manipulation task. The task objectives assigned to the robot are: (Goal 1) retrieving a new tissue box from a closet drawer, placing it on the tea table, and (Goal 2) disposing of the empty tissue box in the trash can. As this task involves interactions with various structures, such as the closet with a revolute joint, the drawer with a prismatic joint, and other rigid objects while navigating through confined 3D spaces, the entire task execution consists of 14 distinct actions. Our method successfully addresses this SMMP challenge, demonstrating progress at three levels.

At the task level, the difficulties in specifying the planning domain and the computational cost in solving the task are reduced, since only two action operators are required. These improvements are quantitatively evaluated in a simplified setup (Sec. VI-C). Throughout execution, the task planner correctly determines the sequence of actions, such as opening the closet before accessing the drawer, and vice versa when closing them. This suggests that our task planning setup faithfully describes the scene and the associated state transitions.

At the motion level, motion planning problems instantiated from symbolic actions are effectively solved, resulting in well-coordinated movements of the robot's base, arm, and manipulated object during mobile manipulation (see tracking performance in Fig. 10). These whole-body motions enable the robot to perform interactive tasks within confined spaces.

At the goal level, selecting a robot configuration that aligns with the motion planner's goal at the end of each action is crucial for the overall success of the task. For instance, in action $\boldsymbol{a}_2$: `place-akr(closet)`, the robot could attempt to open the closet door from either the left or right side. Our plan refinement algorithm successfully accounts for the subsequent action of pulling out the drawer ($\boldsymbol{a}_3$) and therefore selects a robot configuration such that the door is opened from the left, thus avoiding potential obstructions from the nearby carpet and the closet door that would have just been opened.

These three advancements collectively enable a robot to conduct SMMP tasks proficiently. Fig. 10 illustrates the tracking performance of the mobile manipulator. The results demonstrate that the proposed SMMP method can generate executable trajectories that are trackable by the physical robot.

D. Versatility of the Proposed SMMP Framework

The advantages of formulating SMMP from the A-Space perspective extend beyond specific robots or tasks. Since kinematic relationships can characterize various patterns of a robot's movements and a wide range of task goals, our SMMP framework can be extended to other non-traditional setups in mobile manipulation. As illustrated in Fig. 11(a),

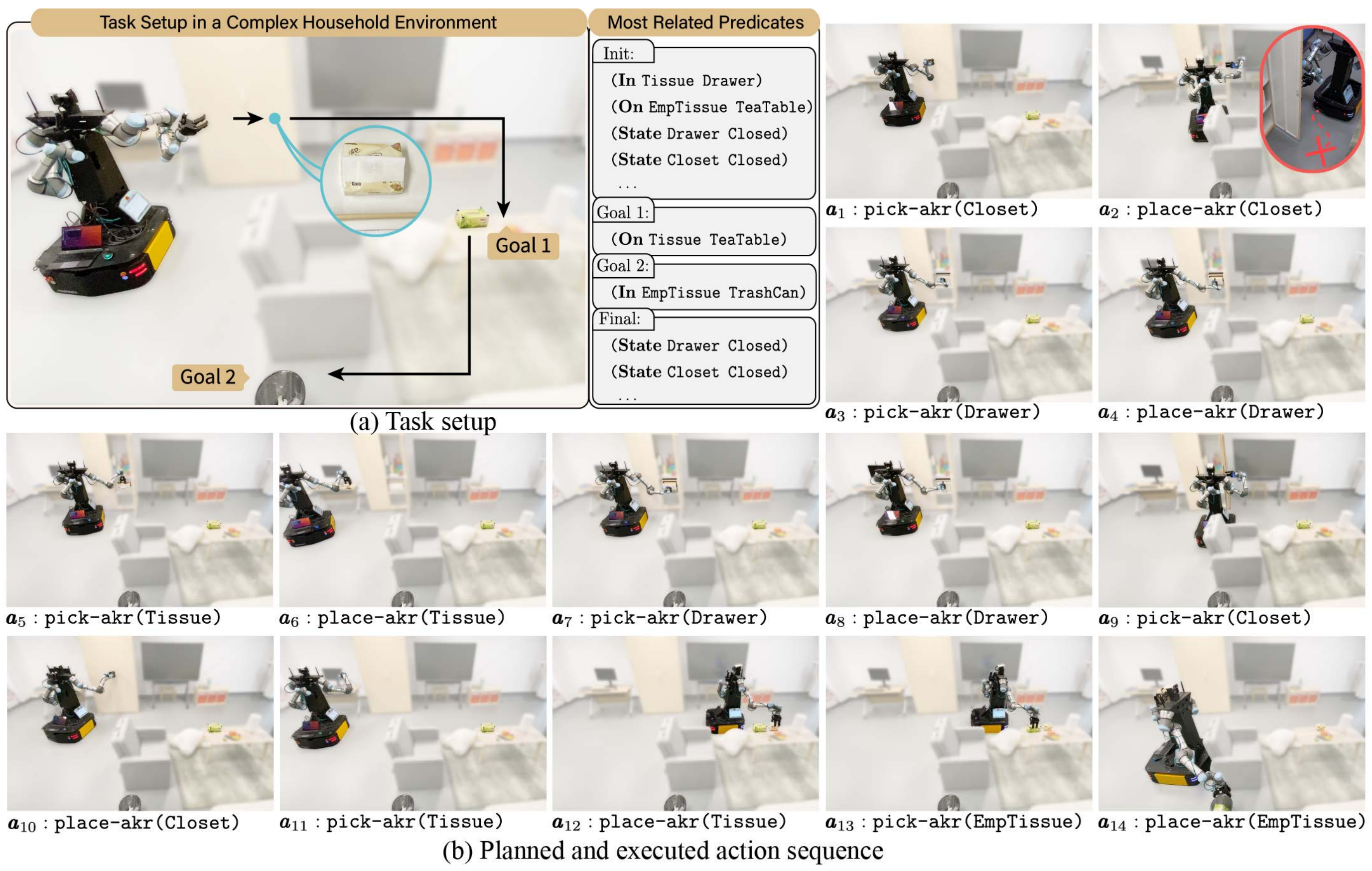


Fig. 9: **Performance of the robot in a sequential mobile manipulation task requiring multiple types of action.** (a) The robot is tasked to dispose of the empty tissue box and replace it with a new one. The task goals and environment states are defined using the PDDL. (b) The robot can solve the 14-step task using only two action operators based on the AKR-based task planner. The feasible trajectories over this long-horizon task are produced accordingly, with plan refinement based on the sequence of the actions.

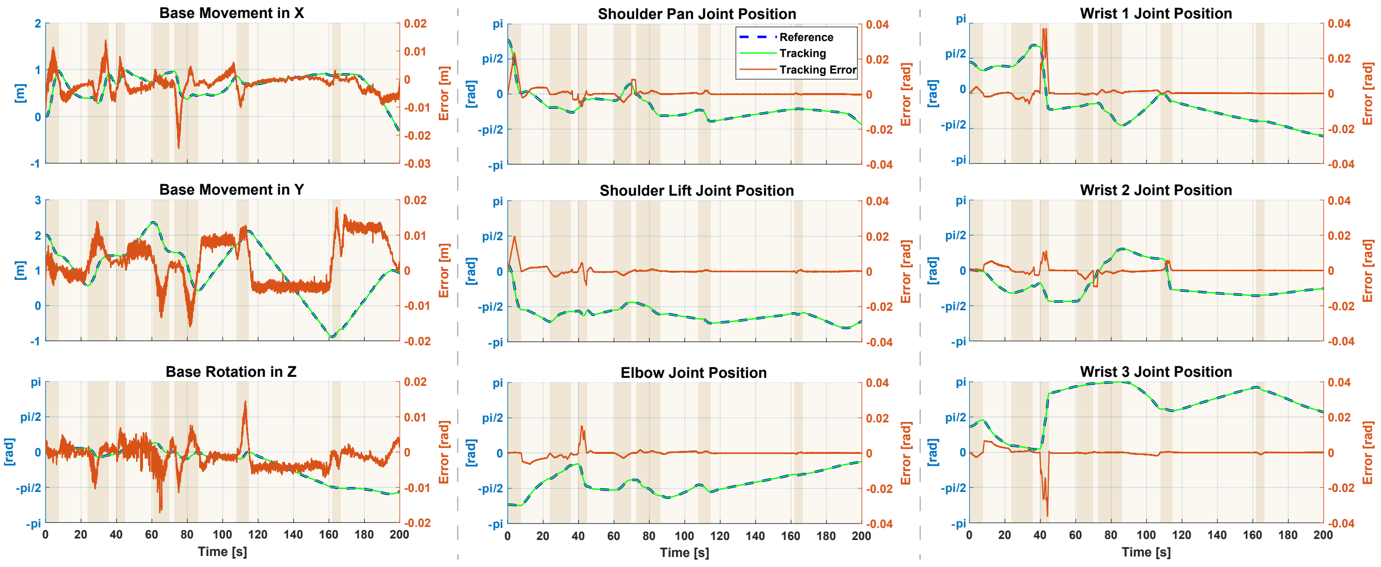


Fig. 10: **Tracking performance in a real-world SMMP task.** The figure shows the reference trajectory generated from the proposed SMMP method, alongside the actual mobile manipulator's trajectory obtained from MCS and robot joint feedback, and the tracking error. The duration of each `pick-akr` action is highlighted with a darker background.

by applying SMMP to an over-actuated UAM, which consists of an over-actuated Unmanned Aerial Vehicle (UAV) and a 3-DoF manipulator, we open up new horizons in sequential multi-step aerial manipulation. Unlike fundamentally stable ground robots, aerial robots have to prioritize their own safety - an increasingly challenging task when interacting with the surrounding environment. We, therefore, implemented a hierarchical control framework for the aerial manipulator to stabilize itself and track desired trajectories [62, 63].

Planning in the A-Space also allows a robot to incorporate

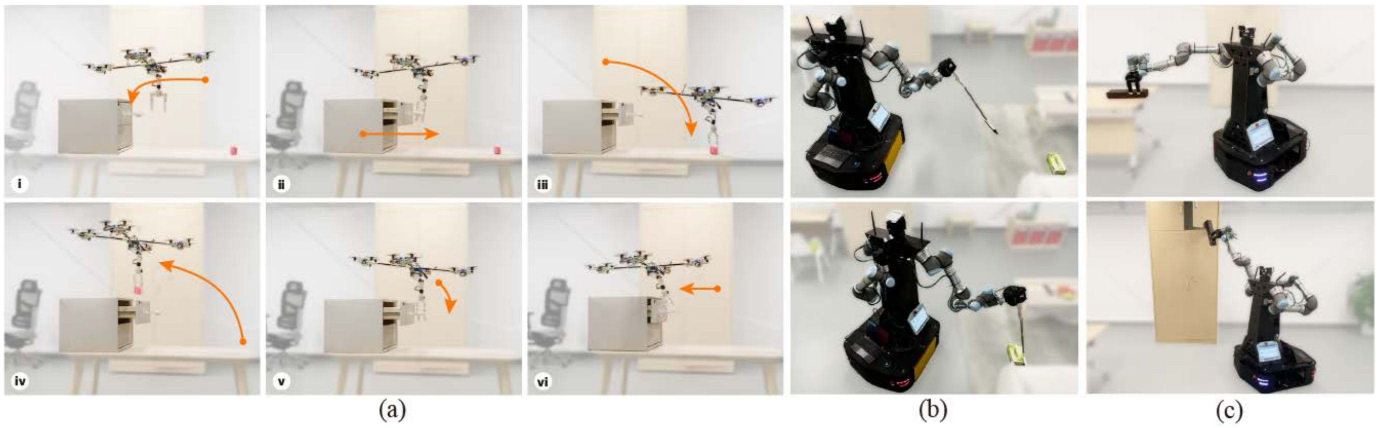


Fig. 11: **Applications of the AKR for different robots and robotic tool-use tasks.** (a) Planning sequential aerial manipulation tasks for an over-actuated Unmanned Aerial Manipulator (UAM). By abstracting the over-actuated UAV's flight as a 6-DoF floating mobile base and combining the kinematics of the 3-DoF manipulator, the proposed SMMP framework is applied to solve the task of placing an object into the drawer. (b) The AKR allows the robot to utilize manipulated objects as tools to fetch the litter with a broom and (c) to close the unreachable closet door with a stick.

external objects as body extensions for non-prehensile manipulation or tool uses. Fig. 11(b)(c) illustrate two robotic tool-use tasks modeled and computed by SMMP. In the first task, the robot utilizes a broom to sweep away litter located between the tea table and the sofa, which the robot cannot approach directly. In the second task, the robot plans to use a stick to close the closet's upper door which is unreachable by its manipulator. These results demonstrate that our framework is not limited to a specific setting; it can be applied to different robot embodiments and has the potential to significantly expand a robot's capabilities by incorporating grasped objects as tools, a crucial step forward in open-world, task-rich environments.

VIII. Discussion and Conclusion

A. Key Findings

Coordinated Robot-scene Motion in Human Environments: Through a series of single-step and multi-step mobile manipulation tasks in Sec. VI-D and Sec. VII-C, we demonstrated the effectiveness of the proposed SMMP framework in generating coordinated robot-scene motions in long-horizon tasks. This type of motion coordination in various settings is crucial for robots operating in human environments that have been primarily designed with bipedal locomotion. Indoor scenes are typically organized to meet human activities, but can be too confined and cluttered for mobile manipulators to navigate and interact with [64]. Although prior work has improved the robustness and efficiency of planning algorithms in confined and cluttered environments [29, 65, 66], the absence of coordinated whole-body motion still fundamentally limits robotic capabilities in many tasks (see supplementary video for examples). After integrating the manipulated object's kinematics with that of the robot, planning in the A-Space can facilitate a general and efficient formulation for robots needing to manipulate a variety of objects with whole-body motions, irrespective of the robot's own morphology. This is evidenced by our experimental results in Fig. 8 and Fig. 11, where the proposed approach successfully generates coordinated base-arm-object trajectories across a variety of scenarios involving a ground mobile manipulator and an aerial manipulator. The robots effectively handle interactions with a range of articulated furniture, like doors and drawers, and achieve significantly higher success rates compared to non-SMMP methods, as quantified in Sec. VI-B.

Integrated Representation for Sequential Tasks: Success in solving SMMP tasks relies heavily on the fluent execution of each single-step action. One notable advantage of framing motion planning problems based on the integrated robot-scene representation, AKR, is the clarity of directly defined goal configurations (*i.e.*, the action parameters) at each step in terms of object states, eliminating the need to specify robot states as part of the goal. For instance, in the task of opening a microwave in Fig. 8(c), the goal is to set the microwave door to a particular angle. The robot's pose and mobile base location are less critical, as they are optimized to adhere to the situational constraints (*i.e.*, the human's location) during motion planning, which have been incorporated into the A-Space. Consequently, in Fig. 8(e), when the level of confinement for the same task increases due to obstacles, the AKR-based planner readily adapts base-arm coordination and produces different trajectories to achieve the same task goal. In contrast, planning methods that treated robot base and arm movements separately require separate goal specifications and action predicates for each component; see Fig. 5. While this approach may be computationally efficient in motion planning, it presents challenges in coordinating movements, especially when dealing with environmental constraints imposed by external objects. As shown in Fig. 4(g), baselines that separate base and arm planning suffer a significant drop in task success rates due to the lack of coordination between their respective end configurations. This misalignment becomes particularly problematic in complex SMMP setups, where the iterative nature of TAMP causes failed motion planning attempts to trigger frequent backtracking and replanning, ultimately leading to failures.

B. Limitations and Future Directions

Efficient Planning for Responsive Operation: As reported in Sec. VI-B, A-Space planning times, produced via trajectory optimization, are notably faster than sampling-based methods but still exceed those of baselines due to the incorporation of additional DoFs. While the proposed SMMP framework can generate flexible and coordinated trajectories within confined spaces, it may be less suitable for applications that require responsive operation. A recent study by Sundaralingam *et al*. introduces a potential solution by parallelizing trajectory optimization computations on GPUs [67]. Their approach demonstrates promising results, speeding up times by a factor of 60. By integrating this GPU-accelerated motion generation library with our SMMP framework, we achieve high-DoF dexterity and responsive operation [68]. Furthermore, our approach demonstrates its applicability across a series of realistic scenes adapted from iThor [69] (see Appendix C for details), reinforcing its potential for real-world deployment.

Obtaining Scene Kinematics: We also acknowledge that the success of the proposed SMMP framework relies heavily on precise knowledge of scene kinematics, which may not always be available in unstructured environments. Recent advancements in computer vision have enabled the reconstruction and inference of part-level relations among objects with articulation [70–72], offering the potential to acquire object kinematics from vision alone [24, 73]. Still, the precision required for manipulation exceeds the current state-of-the-art in computer vision techniques. Integrating tactile feedback at the robot's end-effectors (*e.g*., vision-based tactile sensors) and employing advanced adaptive controllers could enhance robot execution in scenarios with uncertain object kinematics due to perception noise. Moreover, by leveraging readily available environment datasets with known kinematics such as [69], the proposed SMMP framework can serve as an effective data generation platform, addressing the persistent challenge of high-quality data collection in learning-based manipulation research [4].

Interacting with Scenes: Perceiving human-made scenes and the objects within them naturally guides the actions of agents [74, 75], forming the foundations for accomplishing complex tasks. However, existing approaches typically focus on capturing 2D or 3D occupancy information for obstacle avoidance during navigation or pick-and-place manipulation. To tackle longer-horizon tasks, it is crucial to incorporate *actionable* information, such as the actions that entities in the scene can perform and the physical constraints they impose, into robot planning [14, 24]. Identifying what information can be considered actionable and beneficial for subsequent manipulation tasks is a fundamental challenge addressed in this article. Our investigation into SMMP suggests that kinematics could serve as a key bridging stone between perception-based scene understanding and control-based manipulative robot actions.

C. Conclusion

In this article, we introduced the concept of the Augmented Kinematic Representation (AKR), which integrates scene kinematics into the robot's own model to construct a unified Augmented Configuration Space (A-Space) for solving sequential mobile manipulation tasks. We developed a tri-level planning framework that combines PDDL-based task planning, trajectory optimization, and plan refinement, and validated it extensively through both simulation and real-world experiments. Our results demonstrate the framework's effectiveness in generating coordinated whole-body motions, even in confined spaces with articulated objects, and its ability to execute complex tasks involving up to 14 sequential actions without interruption. As kinematics offers a general representation of constrained motion beyond robotic morphology alone, the proposed AKR and A-Space framework holds strong promise for broad application across diverse robot platforms and challenging manipulation scenarios.

Appendix

A. Trajectory Initialization

We implement two trajectory initialization baselines [76]:

1) **Stationary:** The trajectory $\boldsymbol{q}_{1:T}$ is initialized by way-points $\boldsymbol{q}_t$ that are the same as the initial pose $\boldsymbol{q}_1$.
2) **Interpolated:** The trajectory $\boldsymbol{q}_{1:T}$ is initialized by way-points that are linearly interpolated between the initial pose $\boldsymbol{q}_1$ and the goal pose $\boldsymbol{q}_T$.

Next, we investigate how different trajectory initialization methods affect the planning results in three scenarios; see Fig. 12(a)-(c). The robot's task is to pick up the rigid stick and use it to reach a target indicated by the red cube. This task consists of two steps: i) navigate to the stick and pick it up, ii) navigate to and reach the target with the stick. The three scenarios designed for evaluation are increasing in complexity: no obstacle (Fig. 12(a)), two small obstacles (Fig. 12(b)), or a much larger one (Fig. 12(c)). Experimental results reported below are the average of 50 different initial poses, each with 10 times.

A successfully optimized trajectory is a converged result without violating any constraints (*e.g*., collisions). Fig. 12(d) compares success rates. When the environment is clean (Scenario 1), even the simplest *Stationary* trajectory initialization method performs well. When there is additional complexity introduced by the obstacles (Scenario 2), the *Stationary* method deteriorates, whereas the *Interpolated* method still maintains a high success rate. When the navigable space is significantly reduced (Scenario 3), only the proposed $A^{\star}$-based initialization method can consistently perform well to generate feasible plans. Taken together, experimental results indicate that combining the proposed $A^{\star}$-based initialization with the optimization-based motion planner can well handle the challenging motions that require combining navigation and manipulation in cluttered space with obstacle avoidance.

B. Unmanned Aerial Manipulator Platform

The Unmanned Aerial Manipulator (UAM) platform consists of an over-actuated omnidirectional flying vehicle and a 3-DoF robotic manipulator [63]. The flying vehicle integrates four omnidirectional thrust generators, each built with a generic quadcopter (Crazyflie 2.1 control board) and a 2-DoF passive gimbal mechanism [77], enabling independent position and attitude tracking capability. The robotic manipulator comprises three serial rotational DoFs and a parallel gripper. Four Dynamixel XC330-M228-T motors actuate the manipulator, while a Raspberry Pi Zero and a Dynamixel U2D2 converter are fitted to the flying vehicles to receive wireless control commands. Fig. 13 is a schematic diagram of the platform.

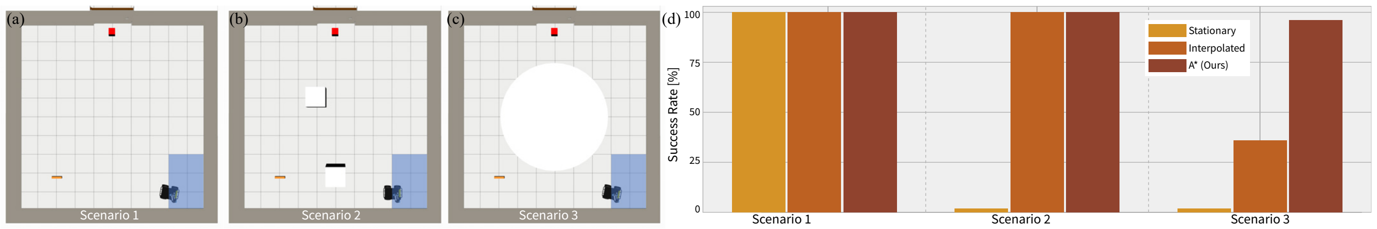


Fig. 12: **Comparisons of motion planning on AKRs by different trajectory initialization methods.** (a)-(c) The experimental scenarios in increasing complexity. The robot's initial pose is uniformly sampled within the blue region; it is tasked to pick up the stick and use it to reach the red cube. (d) The proposed A*-based trajectory initialization has the highest success rates (almost always) in generating a feasible plan. In comparison, the *Stationary* method fails to generate feasible plans in Scenarios 2 and 3. Similarly, the *Interpolated* method struggles in Scenario 3 (30% success rate).

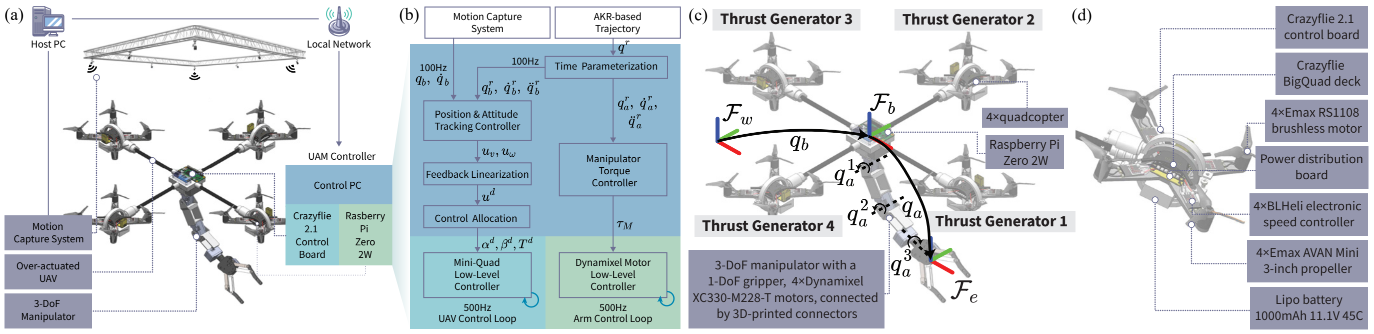


Fig. 13: **The system diagram for the aerial manipulator platform.** (a) The aerial manipulator's communication diagram. (b) The control diagram of the platform. (c) The design of the aerial manipulator platform, and (d) the omnidirectional thrust generator.

C. Additional Materials

The project page[1] for this article hosts supplementary materials that could not be included in the main manuscript due to space constraints. These materials include:

PDDL files: Domain files for the simulation tasks described in Secs. VI-C, VI-D and VII-C, along with the corresponding problem files specifying the initial and goal states, are provided. These are released together with the off-the-shelf task planner to support reproducibility and further research.

Extended simulation results: We tested our AKR-based mobile manipulation planner in realistic, cluttered household scenes adapted from iThor [69]. Each scene includes articulated objects selected from the PartNet-Mobility dataset [78], replacing existing objects of the same category to preserve contextual realism. Grasp poses were generated using AO-Grasp [54]. These results are available on the project page. To handle the scale and complexity of this evaluation, we ported our AKR-based motion planning framework to the Curobo platform and developed a fully automated toolchain for environment setup.

Codebase: The codebase used in our experiments, including that for the extended simulations, is released via the project page.

References

[1] Z. Jiao, Z. Zhang, X. Jiang, D. Han, S.-C. Zhu, Y. Zhu, and H. Liu, "Consolidating kinematic models to promote coordinated mobile manipulations," in *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[2] Z. Jiao, Z. Zhang, W. Wang, D. Han, S.-C. Zhu, Y. Zhu, and H. Liu, "Efficient task planning for mobile manipulation: a virtual kinematic chain perspective," in *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[1]https://aug-kin-rep.github.io

[3] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[4] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *Journal of Machine Learning Research (JMLR)*, vol. 22, no. 30, pp. 1–82, 2021.

[5] M. Diffler, F. Huber, C. Culbert, R. Ambrose, and W. Bluethmann, "Human-robot control strategies for the nasa/darpa robonaut," in *IEEE Aerospace Conference Proceedings*, vol. 8, pp. 3939–3947, IEEE, 2003.

[6] Z. Jiang, X. Cao, X. Huang, H. Li, and M. Ceccarelli, "Progress and development trend of space intelligent robot technology," *Space: Science & Technology*, 2022.

[7] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science Robotics*, vol. 8, no. 84, p. eadf7843, 2023.

[8] O. Khatib, K. Yokoi, O. Brock, K. Chang, and A. Casal, "Robots in human environments: Basic autonomous capabilities," *International Journal of Robotics Research (IJRR)*, vol. 18, no. 7, pp. 684–696, 1999.

[9] O. Brock, J. Park, and M. Toussaint, "Mobility and manipulation," *Springer Handbook of Robotics*, pp. 1007–1036, 2016.

[10] Y. Karayiannidis, C. Smith, F. E. V. Barrientos, P. Ögren, and D. Kragic, "An adaptive control approach for opening doors and drawers under uncertainties," *Transactions on Robotics (T-RO)*, vol. 32, no. 1, pp. 161–175, 2016.

[11] R. Martín-Martín and O. Brock, "Coupled recursive estimation for online interactive perception of articulated objects," *International Journal of Robotics Research (IJRR)*, pp. 1–37, 2019.

[12] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *International Conference on Robotics and Automation (ICRA)*, 2011.

[13] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning.," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[14] Z. Jiao, Y. Niu, Z. Zhang, S.-C. Zhu, Y. Zhu, and H. Liu, "Sequential manipulation planning on scene graph," in *International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[15] J.-P. Sleiman, F. Farshidian, and M. Hutter, "Versatile multicontact planning and control for legged loco-manipulation," *Science Robotics*, vol. 8, no. 81, p. eadg5014, 2023.

[16] T. Lozano-Pérez and L. P. Kaelbling, "A constraint-based method for solving sequential manipulation planning problems," in *International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[17] N. T. Dantam, S. Chaudhuri, and L. E. Kavraki, "The task-motion kit: An open source, general-purpose task and motion-planning framework," *IEEE Robotics and Automation Magazine (RA-M)*, vol. 25, no. 3, pp. 61–70, 2018.

[18] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual Review of Control, Robotics, and Autonomous Systems*, 2021.

[19] B. Kim, L. Shimanuki, L. P. Kaelbling, and T. Lozano-Pérez, "Representation, learning, and planning algorithms for geometric task and motion planning," *International Journal of Robotics Research (IJRR)*, vol. 41, no. 2, pp. 210–231, 2022.

[20] S. Gallagher, *How the body shapes the mind*. Clarendon Press, 2006.

[21] N. P. Holmes and C. Spence, "Beyond the body schema: Visual, prosthetic, and technological contributions to bodily perception and awareness," *Human body perception from the inside out: Advances in visual cognition*, pp. 15–64, 2006.

[22] A. Clark and R. Grush, "Towards a cognitive robotics," *Adaptive Behavior*, vol. 7, no. 1, pp. 5–16, 1999.

[23] M. Wilson, "Six views of embodied cognition," *Psychonomic bulletin & review*, vol. 9, pp. 625–636, 2002.

[24] M. Han, Z. Zhang, Z. Jiao, X. Xie, Y. Zhu, S.-C. Zhu, and H. Liu, "Scene reconstruction with functional objects for robot autonomy," *International Journal of Computer Vision (IJCV)*, vol. 130, no. 12, pp. 2940–2961, 2022.

[25] S. Chitta, B. Cohen, and M. Likhachev, "Planning for autonomous door opening with a mobile manipulator," in *International Conference on Robotics and Automation (ICRA)*, 2010.

[26] A. Jain and C. C. Kemp, "Pulling open doors and drawers: Coordinating an omni-directional base and a compliant arm with equilibrium point control," in *International Conference on Robotics and Automation (ICRA)*, 2010.

[27] M. Stuede, K. Nuelle, S. Tappe, and T. Ortmaier, "Door opening and traversal with an industrial cartesian impedance controlled mobile robot," in *International Conference on Robotics and Automation (ICRA)*, 2019.

[28] M. V. Minniti, F. Farshidian, R. Grandia, and M. Hutter, "Whole-body mpc for a dynamically stable mobile manipulator," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 4, pp. 3687–3694, 2019.

[29] D. Berenson, J. Kuffner, and H. Choset, "An optimization approach to planning for mobile manipulation," in *International Conference on Robotics and Automation (ICRA)*, 2008.

[30] K. Gochev, A. Safonova, and M. Likhachev, "Planning with adaptive dimensionality for mobile manipulation," in *International Conference on Robotics and Automation (ICRA)*, 2012.

[31] D. M. Bodily, T. F. Allen, and M. D. Killpack, "Motion planning for mobile robots using inverse kinematics branching," in *International Conference on Robotics and Automation (ICRA)*, 2017.

[32] J. Haviland, N. Sünderhauf, and P. Corke, "A holistic approach to reactive mobile manipulation," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 3122–3129, 2022.

[33] C. Wang, Q. Zhang, Q. Tian, S. Li, X. Wang, D. Lane, Y. Petillot, and S. Wang, "Learning mobile manipulation through deep reinforcement learning," *Sensors*, vol. 20, no. 3, p. 939, 2020.

[34] H. Ito, K. Yamamoto, H. Mori, and T. Ogata, "Efficient multitask learning with an embodied predictive model for door opening and entry with whole-body control," *Science Robotics*, vol. 7, no. 65, p. eaax8177, 2022.

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

[36] R. Alami, J.-P. Laumond, and T. Siméon, "Two manipulation planning algorithms," in *Proceedings of the Workshop on Algorithmic Foundations of Robotics (WAFR)*, pp. 109–125, AK Peters, Ltd. Natick, MA, USA, 1994.

[37] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," *International Journal of Robotics Research (IJRR)*, vol. 28, no. 1, pp. 104–126, 2009.

[38] K. Hauser and V. Ng-Thow-Hing, "Randomized multi-modal motion planning for a humanoid robot manipulation task," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 6, pp. 678–698, 2011.

[39] J. Barry, L. P. Kaelbling, and T. Lozano-Pérez, "A hierarchical approach to manipulation with diverse actions," in *International Conference on Robotics and Automation (ICRA)*, 2013.

[40] M. Toussaint, K. Allen, K. A. Smith, and J. B. Tenenbaum, "Differentiable physics and stable modes for tool-use and manipulation planning," in *Robotics: Science and Systems (RSS)*, 2018.

[41] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "Pddl-the planning domain definition language," tech. rep., Yale Center for Computational Vision and Control, 1998.

[42] M. Helmert, "The fast downward planning system," *Journal of Artificial Intelligence Research*, vol. 26, pp. 191–246, 2006.

[43] E. Karpas and D. Magazzeni, "Automated planning for robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 417–439, 2020.

[44] E. Erdem, K. Haspalamutgil, C. Palaz, V. Patoglu, and T. Uras, "Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation," in *International Conference on Robotics and Automation (ICRA)*, 2011.

[45] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *International Conference on Robotics and Automation (ICRA)*, 2014.

[46] C. R. Garrett, T. Lozano-Perez, and L. P. Kaelbling, "Ffrob: Leveraging symbolic planning for efficient task and motion planning," *International Journal of Robotics Research (IJRR)*, vol. 37, no. 1, pp. 104–136, 2018.

[47] X. Zhang, Y. Zhu, Y. Ding, Y. Jiang, Y. Zhu, P. Stone, and S. Zhang, "Symbolic state space optimization for long horizon mobile manipulation planning," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 866–872, IEEE, 2023.

[48] Z. Yang, C. Garrett, D. Fox, T. Lozano-Pérez, and L. P. Kaelbling, "Guiding long-horizon task and motion planning with vision language models," *arXiv preprint arXiv:2410.02193*, 2024.

[49] Y. Sung, Z. Wang, and P. Stone, "Learning to correct mistakes: Backjumping in long-horizon task and motion planning," in *Conference on Robot Learning (CoRL)*, pp. 2115–2124, PMLR, 2023.

[50] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *Transactions on Robotics (T-RO)*, vol. 26, no. 3, pp. 576–584, 2010.

[51] C. Rösmann, F. Hoffmann, and T. Bertram, "Kinodynamic trajectory optimization and control for car-like robots," in *International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[52] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *International Journal of Robotics Research (IJRR)*, vol. 33, no. 9, pp. 1251–1270, 2014.

[53] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: gradient optimization techniques for efficient motion planning," in *International Conference on Robotics and Automation (ICRA)*, 2009.

[54] C. P. Morlans, C. Chen, Y. Weng, M. Yi, Y. Huang, N. Heppert, L. Zhou, L. Guibas, and J. Bohg, "Ao-grasp: Articulated object grasp generation," in *International Conference on Intelligent Robots and Systems (IROS)*, pp. 13096–13103, IEEE, 2024.

[55] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning," in *Proceedings of the International Conference on Automated Planning and Scheduling*, 2020.

[56] D. Arthur and S. Vassilvitskii, "K-means++ the advantages of careful seeding," in *Proceedings of ACM-SIAM Symposium on Discrete algorithms*, 2007.

[57] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *International Conference on Robotics and Automation (ICRA)*, 2000.

[58] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics and Automation Magazine (RA-M)*, vol. 19, no. 4, pp. 72–82, 2012.

[59] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *International Journal of Robotics Research (IJRR)*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[60] C. Muise, K. Taylor-Muise, and A. Coles, "Planning.domains." http://planning.domains/.

[61] N. Lipovetzky and H. Geffner, "Width and serialization of classical planning problems," in *Proceedings of the 20th European Conference on Artificial Intelligence*, ECAI'12, p. 540–545, IOS Press, 2012.

[62] Y. Su, P. Yu, M. Gerber, L. Ruan, and T.-C. Tsao, "Nullspace-based control allocation of overactuated uav platforms," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 8094–8101, 2021.

[63] Y. Su, J. Li, Z. Jiao, M. Wang, C. Chu, H. Li, Y. Zhu, and H. Liu, "Sequential manipulation planning for over-actuated unmanned aerial manipulators," in *International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[64] W. Wang, Z. Zhao, Z. Jiao, Y. Zhu, S.-C. Zhu, and H. Liu, "Rearrange indoor scenes for human-robot co-activity," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[65] Z. Han, J. Allspaw, G. LeMasurier, J. Parrillo, D. Giger, S. R. Ahmadzadeh, and H. A. Yanco, "Towards mobile multi-task manipulation in a confined and integrated environment with irregular objects," in *International Conference on Robotics and Automation (ICRA)*, 2020.

[66] C. Nam, S. H. Cheong, J. Lee, D. H. Kim, and C. Kim, "Fast and resilient manipulation planning for object retrieval in cluttered and confined environments," *Transactions on Robotics (T-RO)*, vol. 37, no. 5, pp. 1539–1552, 2021.

[67] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. Van Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos, *et al.*,

"Curobo: Parallelized collision-free robot motion generation," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[68] Z. Li, Y. Niu, Y. Su, H. Liu, and Z. Jiao, "Dynamic planning for sequential whole-body mobile manipulation," in *IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2024.

[69] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, *et al.*, "Ai2-thor: An interactive 3d environment for visual ai," *arXiv preprint arXiv:1712.05474*, 2017.

[70] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[71] A. Bokhovkin, V. Ishimtsev, E. Bogomolov, D. Zorin, A. Artemov, E. Burnaev, and A. Dai, "Towards part-based understanding of rgb-d scans," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[72] Z. Zhang, L. Zhang, Z. Wang, Z. Jiao, M. Han, Y. Zhu, S.-C. Zhu, and H. Liu, "Part-level scene reconstruction affords robot interaction," in *International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[73] M. Han, Z. Zhang, Z. Jiao, X. Xie, Y. Zhu, S.-C. Zhu, and H. Liu, "Reconstructing interactive 3d scenes by panoptic mapping and cad model alignments," in *International Conference on Robotics and Automation (ICRA)*, 2021.

[74] J. J. Gibson, *The perception of the visual world.* Houghton Mifflin, 1950.

[75] J. J. Gibson, *The senses considered as perceptual systems.* Houghton Mifflin, 1966.

[76] B. Magyar, N. Tsiogkas, J. Deray, S. Pfeiffer, and D. Lane, "Timed-elastic bands for manipulation motion planning," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 4, pp. 3513–3520, 2019.

[77] P. Yu, Y. Su, M. J. Gerber, L. Ruan, and T.-C. Tsao, "An over-actuated multi-rotor aerial vehicle with unconstrained attitude angles and high thrust efficiencies," *IEEE Robotics and Automation Letters (RA-L)*, vol. 6, no. 4, pp. 6828–6835, 2021.

[78] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, *et al.*, "Sapien: A simulated part-based interactive environment," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11097–11107, 2020.

Ziyuan Jiao (Member, IEEE) received his Ph.D. degree in Mechanical Engineering from the University of California, Los Angeles (UCLA) in 2022. He is now a research scientist at the State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI). Dr. Jiao's research focuses on robotic manipulation planning, task and motion planning, mobile manipulation, and optimization. He is a recipient of the NSFC Youth Program.

Yida Niu (Student Member, IEEE) received his M.S. degree in Robotics from Northeastern University (NEU) in 2021. He is currently a Ph.D. student at the Institute for Artificial Intelligence, Peking University (PKU), and an intern research scientist at the State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI). His research focuses on robotic manipulation, task and motion planning, and mobile manipulation.

Zeyu Zhang (Member, IEEE) received his Ph.D. degree in Computer Science from the University of California, Los Angeles (UCLA) in 2023. He is a research scientist at State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI). He received an M.S. degree in Computer Science from UCLA in 2019 and a B.S. degree in Computer Science from Hunan University in 2017. His research interests focus on robot perception, learning, and cognitive robotics.

YangYang Wu is a Research Engineer at the State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI). He received his M.S. degree from Columbia University in 2021. Prior to joining BIGAI, he was a Research Assistant at Tsinghua University, working on robot navigation and manipulation. His research interests include mobile manipulation, task and motion planning, robot learning, and human-robot interaction.

Yao Su (Member, IEEE) received the M.S. and Ph.D. degrees from the Department of Mechanical and Aerospace Engineering, University of California, Los Angeles (UCLA) in 2017 and 2021. He is now a research scientist at State Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence (BIGAI). His research interests include robotics, control, optimization, trajectory planning, and mechatronics.

Yixin Zhu is an Assistant Professor at the School of Psychological and Cognitive Sciences and an Assistant Dean at the Institute for Artificial Intelligence, both at Peking University. His research builds interactive AI by integrating high-level common sense (functionality, affordance, physics, causality, intent) with raw sensory inputs (pixels and haptic signals) to enable richer representation and cognitive reasoning on objects, scenes, shapes, numbers, and agents.

Hangxin Liu (Member, IEEE) is a Research Scientist at the Beijing Institute for General Artificial Intelligence (BIGAI), where he serves as Deputy Director of the State Key Laboratory of General Artificial Intelligence and Director of the Robotics Lab. He received his Ph.D. in Computer Science and M.S. in Mechanical Engineering from UCLA. His research spans perception, reasoning, planning, and learning for robotic systems and embodied AI.

Song-Chun Zhu (Fellow, IEEE) is the Director of the Beijing Institute for General Artificial Intelligence (BIGAI) and a Chair Professor at Peking University and Tsinghua University. He received his Ph.D. from Harvard University in 1996 and was a professor at UCLA, where he founded the Center for Vision, Cognition, Learning, and Autonomy (VCLA) before returning to China in 2020. His honors include the Marr Prize, Sloan Fellowship, and Helmholtz Test-of-Time Award. He has served as General Chair for CVPR and led major U.S. research initiatives (MURI). With over 400 publications, his research spans computer vision, cognition, robotics, and machine learning, driven by the goal of a unified theory of intelligence.