Dynamic Planning for Sequential Whole-body Mobile Manipulation

Zhitian Li[1⋆], Yida Niu[1,2⋆], Yao Su[1], Hangxin Liu[1], Ziyuan Jiao[1†]

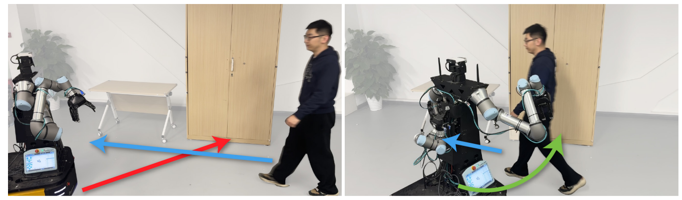
2024 IEEE 19th Conference on Industrial Electronics and Applications (ICIEA) | 979-8-3503-6086-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICIEA61579.2024.10664822

***Abstract*—The dynamic Sequential Mobile Manipulation Planning (SMMP) framework is essential for the safe and robust operation of mobile manipulators in dynamic environments. Previous research has primarily focused on either motion-level or task-level dynamic planning, with limitations in handling state changes that have long-term effects or in generating responsive motions for diverse tasks, respectively. This paper presents a holistic dynamic planning framework that extends the Virtual Kinematic Chain (VKC)-based SMMP method, automating dynamic long-term task planning and reactive whole-body motion generation for SMMP problems. The framework consists of an online task planning module designed to respond to environment changes with long-term effects, a VKC-based whole-body motion planning module for manipulating both rigid and articulated objects, alongside a reactive Model Predictive Control (MPC) module for obstacle avoidance during execution. Simulations and real-world experiments validate the framework, demonstrating its efficacy and validity across sequential mobile manipulation tasks, even in scenarios involving human interference.**

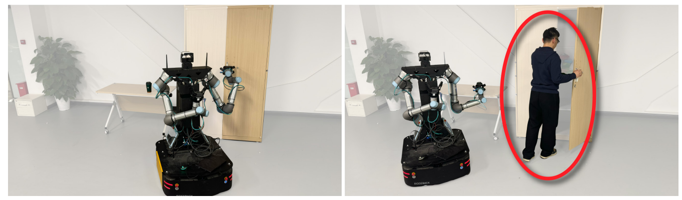
I. Introduction

Sequential Mobile Manipulation Planning (SMMP) is becoming increasingly crucial for intelligent service robots to autonomously operate and assist humans in diverse daily activities within task-rich household environments. In such scenarios, robots are required to perform a broad spectrum of manipulation tasks, navigating through expansive and confined workspaces while planning and executing sequences of distinct interactive actions. Existing research in mobile manipulation [1–3] and sequential manipulation [4–7] has made notable progress in addressing SMMP challenges. However, these methods usually assume a static environment. This assumption impedes the practical deployment of SMMP methods in service robots for household environments, as these environments pose a unique challenge: human activities can introduce unforeseen changes to environment states or even impact the robot itself. Therefore, it is crucial to develop a dynamic planning framework that can adapt to unforeseen environmental changes, and replan action sequences or motion trajectories when required, thereby ensuring the robot's reliability in the presence of unpredictable alterations.

Research on dynamic planning for service robots, particularly those capable of both navigation and manipulation, falls into two primary groups: motion-level and task-level. Motion-level dynamic planning approaches [8–14] focus on individual mobile manipulation tasks, achieving dynamic planning through rapid, reactive motion generation. These approaches formulate environment states susceptible to unexpected environment changes into objectives or constraints of the planning problem and periodically update the motion trajectory with high frequency. While excelling at responding to immediate environmental changes, such as evading humans during manipulation (see Fig. 1a), these methods struggle with state changes that have long-term effects because of the increased computational overhead. On the other hand, as Fig. 1b suggests, task-level dynamic planning approaches [15–18] involve sequences of interactive actions. Typically, they employ rapid high-level symbolic task planning for action sequencing, especially suitable for long-term operations (see Fig. 1b). Each high-level action corresponds to a low-level motion generator for execution. However, given the diverse nature of SMMP problems involving various interactive actions, designing low-level (*i.e.* motion-level) dynamic planners would demand substantial manual effort. Consequently, existing dynamic planning approaches for SMMP focus on simple manipulation tasks, and there is still considerable progress needed before they can be applied in real-world settings.



(a) **Motion-level.** Left: The human's movement interferes with the robot's approach to the door. Right: The robot must dynamically adjust its movement to avoid colliding with the human.



(b) **Task-level.** Left: The robot opens the cabinet door before placing the bottle inside. Right: The human closes the cabinet door, prompting the robot to replan its task sequence to reopen the door.

Fig. 1: Human activities can interfere with a robot's operation, necessitating dynamic planning at motion and task levels.

A noteworthy recent contribution to SMMP is presented in the work of [1, 4]. This research introduces a general SMMP planning method based on Virtual Kinematic Chain (VKC) and body schema, automating the generation of extended action sequences for SMMP problems involving diverse mobile manipulation tasks. The method showcases sophisticated whole-body coordination in confined spaces and adaptability to daily objects with diverse articulations, offering insights for deployment in service robots operating in the real world.

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

⋆ Zhitian Li and Yida Niu contributed equally to this work.

† Corresponding author. [1] State Key Laboratory of General Artificial Intelligence, BIGAI. [2] Institute for Artificial Intelligence, Peking University.

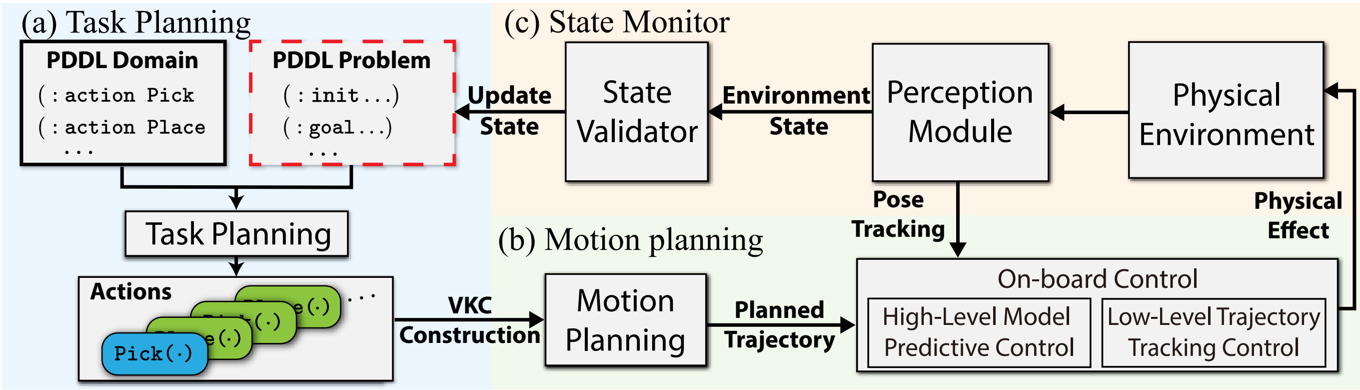


Fig. 2: **Overview of the proposed VKC-based dynamic planning framework.** (a) Our task planning module first searches for a sequence of symbolic actions [4] within the Planning Domain Definition Language (PDDL)-based planning domain. These actions indicate how VKCs are modified during interactions and can be instantiated to motion planning problems. (b) Whole-body trajectories are subsequently generated by the motion planning module, considering motion constraints on VKC structures and task goals [1]. Then, onboard controllers realize dynamic obstacle avoidance and trajectory tracking. (c) As tasks progress, the state monitor detects changes in the environment, provides feedback on robot poses to the onboard controller, and dynamically updates the PDDL problem for reactive task replanning.

Building upon the VKC-based SMMP method, we present a dynamic planning framework that is resilient to uncertainties, undesired conditions, and human interventions. To address task-level interference, we employ an online task replanning module that reacts to environmental changes with long-term effects by updating the action sequence. For motion-level interference, we utilize the VKC-based whole-body trajectory generation method for mobile manipulation planning, followed by integrating a reactive MPC module for dynamic obstacle avoidance during execution. The combined framework is depicted in Fig. 2. Simulation and experimental validations demonstrate the efficacy of our proposed dynamic planning framework in handling various household tasks, including sequential mobile manipulation with human interventions, and articulated object mobile manipulation amid moving obstacles.

A. Related Work

Sequential Mobile Manipulation Planning (SMMP): Solving SMMP problems presents challenges, mainly due to the high complexity in contact modes and the high dimensionality of the state space. Recent efforts have been directed towards addressing these challenges hierarchically, with notable approaches including Multi-Modal Motion Planning (MMMP) [19, 20], and Task and Motion Planning (TAMP) [4, 6, 21]. Despite significant progress from various perspectives, SMMP problems remain largely unsolved. The fundamental concept behind existing hierarchical strategies is similar: the symbolic task planner incrementally constructs the plan, with necessary backtracking, until the primitive manipulation skills generate a feasible motion for execution that satisfies all task constraints. Consequently, most existing works on SMMP problems simplify motion planning complexity, addressing only simple pick-and-place manipulation problems. These approaches avoid substantial efforts in designing i) intricate planning domains, and ii) specific motion planners for complex mobile manipulation tasks. However, such simplifications may not effectively tackle practical challenges in task-rich scenarios, limiting the deployment of methods in real-world settings. In contrast, the VKC-based SMMP method [1, 4] leverages the advancements of hierarchical strategies in TAMP methods. It demonstrates that the VKC perspective serves as an intermediate representation, offering a more unified approach to mobile manipulation task modeling. The method showcases its capability to adapt to a wide spectrum of mobile manipulation tasks. This paper aims to develop a dynamic planning framework for the VKC-based SMMP method, taking a step closer to addressing the complexities and challenges inherent in SMMP problems.

Reactive Planning (Online Replanning): Motion-level dynamic planning/control methods [8] typically function as fast local planners, updating the reference trajectory generated by global planners in real-time [9, 22] to avoid potential collisions in highly dynamic environments. Various online motion planning approaches for mobile manipulators have been proposed, including redundant Inverse Kinematics (IK) controller [10], obstacle trajectory prediction [8], *etc*. Among these, the MPC-based method prevails [11, 13, 14], as it converts static/dynamic obstacle avoidance into convex inequality constraints and integrates kinematics, dynamics, and physical constraints within the optimization formulation. However, in practice, state changes may cause long-term effects, making the problem too complex to solve for real-time operation due to the extended planning horizon.

Task-level dynamic planning approaches [15] utilize high-level planning utilizing formalized planning languages such as behavior trees [18], or self-defined mode switches [16, 17] to address long-horizon planning problems. High-level planning proves to be more efficient than motion planning and is more suitable for modeling sequential planning problems involving alternations in system kinematics/dynamics. However, action sequences generated by high-level planners must be interpretable into low-level trajectories for execution, necessitating significant effort in designing motion primitives. Consequently, existing dynamic planning approaches for SMMP focus on simple manipulation tasks and are far from applicable to real-world settings with emerging tasks.

II. Dynamic VKC-based SMMP Framework

A. VKC-based Task Decomposition

The initial step for robots in solving SMMP problems is to decompose the task into a sequence of temporally feasible actions, which necessitates task planning (Fig. 2a). Conventional task planning approaches involve defining symbolic actions and states, often assuming these actions are executable. However, many symbolic actions are challenging to instantiate at the motion level. From the VKC perspective, actions are defined as modifications to the VKC structure, characterized by `pick` and `place`, transforming the SMMP problem into a sequence of VKC state and structure changes. The `pick` action moves the VKC to an object and extends its kinematics by adding a virtual attachment joint to connect the object and the end-effector. `pick` encompasses tasks where mobile manipulators interact with the environment, such as picking up an object or grasping a handle. On the other hand, the `place` action moves the object, already connected to the current VKC, to a goal pose. The object's incorporation into the VKC imposes kinematic constraints on the planner. After reaching the goal pose, `place` breaks the VKC at the virtual attachment joint, separating the mobile manipulator from the object, which is then placed at the disconnected position. `place` pertains to tasks where mobile manipulators stop interacting with the environment, like placing an object. These two actions can represent a wide range of mobile manipulation tasks, and utilizing them helps to simplify the planning domain by eliminating unnecessary actions and intermediate state predicates. Task planning on such a simplified domain has been demonstrated to be more efficient compared to conventional domains [4], thereby benefiting the dynamic task planning module in terms of computational efficiency.

B. Motion Planning on VKC

The action sequence composed of `pick` and `place` indicates how VKCs are modified during interactions and how an action is instantiated to a motion planning problem. As shown in Fig. 3, constructing a VKC involves four key inputs: the robot kinematics, the object kinematics, a virtual base, and a virtual attachment. The kinematics of the robot and the object are assumed to be known. The virtual floating base reflects the motion possibilities of the mobile platform, and a 3-DoF kinematic chain is constructed to imitate its planar motion. Virtual attachments characterize the motion constraints and spatial relations between the robot's end-effector and the attachable link (*i.e.*, links in contact) on the object.

The constructed VKC, as shown in Fig. 3, for closing a cabinet door involves inserting a virtual attachment between the robot's end-effector link and the door's handle, connecting the kinematics of the robot and the manipulated object. Notably, VKC is a malleable representation, allowing the insertion of additional virtual attachments between different components, resulting in an extended VKC. If a manipulated object is articulated, its kinematic model must be inverted for the constructed kinematic chain to remain serial. Detailed steps for VKC construction can be found in [1].

The state vector, denoted as $\boldsymbol{x} = [\boldsymbol{q}_B^\top, \boldsymbol{q}_M^\top, \boldsymbol{q}_O^\top]^\top \in \mathcal{X}_{\text{free}}$, represents the state of a VKC, where $\boldsymbol{q}_B = [x_B, y_B, \theta_B]$ is

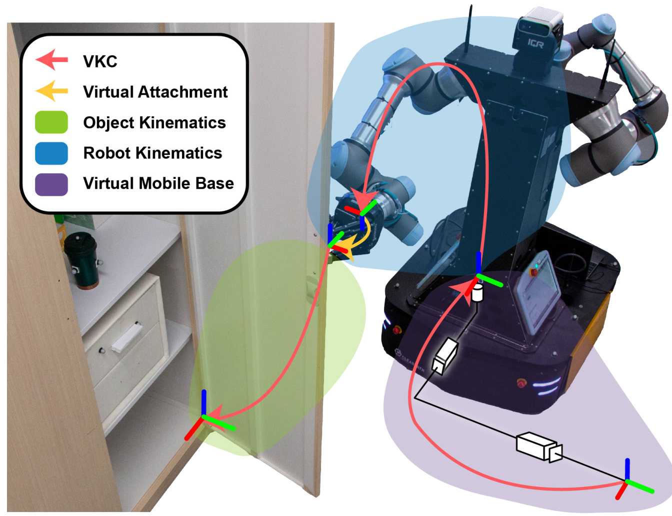


Fig. 3: **Modeling a mobile manipulation task with the VKC.** The construction of a VKC entails four essential inputs: the robot kinematics, the object kinematics, a virtual base, and a virtual attachment. Due to the articulated nature of the cabinet, its kinematic model needs inversion to preserve a serial VKC.

the vector of the omnidirectional mobile base, $\boldsymbol{q}_M \subset \mathbb{R}^6$ is the vector of manipulator joint angles, $\boldsymbol{q}_O$ is the vector of joint values for the manipulated object, and $\mathcal{X}_{\text{free}} \subset \mathbb{R}^n$ is the collision-free configuration space of VKC, where n is the total Degree of Freedom (DoF). The motion planning problem on VKCs is equivalent to finding a T-step path $\boldsymbol{x}_{[1:T]} = \langle \boldsymbol{x}_{[1]}, \boldsymbol{x}_{[2]}, \ldots, \boldsymbol{x}_{[T]} \rangle \in \mathcal{X}_{\text{free}}$, where the subscript $[k]$ describe a variable at the step k, which can be formulated and solved through trajectory optimization [1, 23]. As shown in Fig. 2b, the planned trajectory $\boldsymbol{x}_{1:T}$ is subsequently time-parameterized and then sent to the on-board controller for execution. A high-level MPC (detailed in Sec. III) is employed for movable obstacle avoidance, utilizing the VKC trajectory as the reference trajectory. Afterward, the optimized command is sent to the low-level arm/base controller for execution.

C. State Monitor

The state monitor (Fig. 2c) tracks environment changes and updates the state changes to the task planning and motion planning modules during SMMP tasks. Specifically, the perception module detects object poses from the Motion Capture System (MCS) and tracks necessary object states (*e.g.*, mobile base's location), sending them to the onboard control module. Additionally, some object states are further sent to the state validator, which verifies the predicate states in the PDDL file. For instance, to check if an object is `On` another object, both object states are sent to the state validator, and their spatial relation is computed through predefined rules [24, 25]. The predicates that describe object relations are then updated in the PDDL problem file. At the end of each `pick` and `place` action pair, the state validator is invoked to update the environment state. Subsequently, the task planner is called to plan a new action sequence. If the newly planned action sequence is different from the previously planned sequence, our framework will discard the previous sequence and utilize the new one for motion planning.

III. MPC-based Motion Control

A. Mobile Manipulator's Model

We consider a velocity-controlled mobile manipulator with system kinematics described as:

$$^{bb}\dot{\boldsymbol{z}} = \begin{bmatrix} \cos\theta_B\,\dot{x}_B - \sin\theta_B\,\dot{y}_B \\ \sin\theta_B\,\dot{x}_B + \cos\theta_B\,\dot{y}_B \\ \dot{\theta}_B \\ \dot{\boldsymbol{q}}_M \end{bmatrix}, \tag{1}$$

where the state vector $^{bb}\boldsymbol{z}$ is the configuration of the platform in the local frame. Choosing $\boldsymbol{u} = \left[\dot{x}_B, \dot{y}_B, \dot{\theta}_B, \dot{\boldsymbol{q}}_M\right]^\top$ as input and utilizing a discretization scheme, the discrete-time nonlinear transition function f of the system can be derived in a general form:

$$\boldsymbol{z}_{[k+1]} = f\left(\boldsymbol{z}_{[k]}, \boldsymbol{u}_{[k]}\right). \tag{2}$$

B. Controller Design

The MPC controllers obtain the control input by solving an Optimal Control Problem (OCP) at each sample time [26]. That involves minimizing a given cost function over a defined prediction horizon subjected to defined constraints. In this work, the OCP is designed to generate feasible and smooth motions required for the mobile manipulator to track the reference trajectory while avoiding dynamic obstacles in the environment [23]. As a result, we formulate the OCP as:

$$\min_{\boldsymbol{z}_{[0:N]}, \boldsymbol{u}_{[0:N]}} \sum_{k=0}^{N} J\left(\boldsymbol{z}_{[k]}, \boldsymbol{u}_{[k]}\right), \tag{3a}$$

$$\text{s.t. } \boldsymbol{z}_{[k+1]} = f\left(\boldsymbol{z}_{[k]}, \boldsymbol{u}_{[k]}\right), \quad \forall k = 1, \cdots, N \tag{3b}$$

$$\boldsymbol{u}_{[k]} \in \mathcal{U}, \boldsymbol{z}_{[k]} \in \mathcal{Z}, \tag{3c}$$

$$\boldsymbol{z}_0 = \boldsymbol{z}(0), \tag{3d}$$

where N is the prediction horizon, Eq. (3a) is the cost function, Eq. (3b) is the kinematic constraints, Eq. (3c) is the constraint for admissible states ($\mathcal{Z}$) and control inputs ($\mathcal{U}$), and Eq. (3d) is the initial states constraint.

Due to the difference between `pick` and `place` actions, we formulate the cost function separately:

`pick`

1) Trajectory Tracking Cost: J^t is defined as [27]:

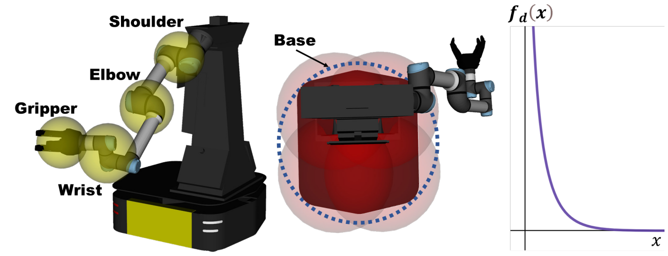
$$\begin{aligned} J^t\left(\boldsymbol{z}_{[k]}, \boldsymbol{u}_{[k]}, \boldsymbol{u}_{[k-1|k]}\right) &= \sum_{j=0}^{N} \underbrace{\left\|\boldsymbol{z}^{\text{ref}}_{[k+j]} - \boldsymbol{z}_{[k+j|k]}\right\|^2_{\boldsymbol{Q}_z}}_{\text{Tracking cost}} \\ &+ \sum_{j=0}^{N-1} \underbrace{\left\|\boldsymbol{u}_{[k+j|k]} - \boldsymbol{u}_{[k+j-1|k]}\right\|^2_{\boldsymbol{Q}_{\Delta u}}}_{\text{Input smoothness cost}} + \underbrace{\left\|\boldsymbol{z}_{[N]} - \boldsymbol{z}^{\text{ref}}_{[E]}\right\|^2_{\boldsymbol{Q}_N}}_{\text{Terminal cost}}, \end{aligned} \tag{4}$$

where the subscript $[k+j \mid k]$ denotes the predicted value of a variable at time step $k+j$, produced at the time step k; $\boldsymbol{Q}_z, \boldsymbol{Q}_{\Delta u}, \boldsymbol{Q}_N$ are positive definite weight matrices for the states, inputs and final state respectively. In this objective function, the first term penalizes deviating from the reference trajectory $\boldsymbol{z}^{\text{ref}}$, the second term encourages the smoothness of successive inputs, and the third term ensures accurate terminal state which is crucial for successful grasping.

2) Obstacle Definition and Avoidance: For collision avoidance, we approximate the contact model of the robot with a combination of spheres and utilize the barrier function (Fig. 4):

$$f_d(x) = \left(\alpha_1 + e^{-\alpha_2(x-\alpha_3)}\right)^{-1}, \tag{5}$$

to formulate soft constraint as a non-piecewise function to speed up the solver, where x is the distance from the collision sphere to the obstacles, and the parameters α_{1-3} control the safety margin and barrier function gradient.



(a) (b)

Fig. 4: **Contact model design for collision avoidance.** (a) Contact model approximation. (b) Non-piecewise barrier function.

For the dynamic pedestrian, we parameterize the corresponding collision model as a cylinder and assume a constant velocity model as:

$$\boldsymbol{r}^o_{[k]} = \boldsymbol{r}^o_{[k-1]} + \dot{\boldsymbol{r}}^o_{[k-1]} \cdot \Delta t, \tag{6}$$

where $\boldsymbol{r}^o_{[k]}$ is the center position of the cylinder in the world frame at step k.

The obstacle constraint is defined as:

$$J^{\text{obs}}\left(\boldsymbol{p}_{[k]}, \boldsymbol{r}^o_{[k]}\right) = \sum_{i=0}^{n_o} \left(f_d\left(\left\|\boldsymbol{p}_{i[k]} - \boldsymbol{r}^o_{[k]}\right\|\right)\right)_{\boldsymbol{Q}_p}, \tag{7}$$

where $\boldsymbol{p}_i$ is the position of i-th contact sphere's center, n_o is the number of contact spheres, $\boldsymbol{Q}_p$ is the weighting matrix.

Boundary obstacles, such as the wall or other furniture in the in-room environment, are also considered analogously as soft planar constraints based on collision primitive position $\boldsymbol{p}$ and normal vector $\boldsymbol{\lambda}_i$:

$$J^{\pi}\left(\boldsymbol{p}_{[k]}\right) = \sum_{i=0}^{n_o} \sum_{j=0}^{n_\pi} \left(f_d\left(\pi\left(\boldsymbol{p}_{i[k]}, \boldsymbol{r}^{\text{obs}}_{j[k]}\right)\right)\right)_{\boldsymbol{Q}_\pi} \tag{8}$$

where $\pi\left(\boldsymbol{p}_i, \boldsymbol{r}^{\text{obs}}_j\right) = \boldsymbol{\lambda}_j \cdot \left(\boldsymbol{p}_i - \boldsymbol{r}^{\text{obs}}_j\right)$, n_π is the number of static obstacles, $\boldsymbol{Q}_\pi$ is the related weighting matrix.

For `pick`, we define the cost function J in Eq. (3) as $J = J^t + J^{\text{obs}} + J^{\pi}$, and solve the velocity command of the mobile base and manipulator jointly. However, for `place` when manipulating articulated objects, the VKC-based framework plans the trajectory of the whole kinematic chain $\boldsymbol{x}$ as the reference trajectory, including the motion of the passive DoF $\boldsymbol{q}^\top_O$ on the object. This design requires the attachment location on the articulated object maintained within the workspace of the mobile manipulator during MPC-based tracking control.

`place`

3) Attached Constraint: An attachment constraint is employed to keep the attachment location reachable by the arm:

$$J^a = \left(f_d\left(d_{\text{up}} - d_a\right) + f_d\left(d_a - d_{\text{low}}\right)\right)_{\boldsymbol{Q}_a} \tag{9}$$

Algorithm 1: Control Command Compulation

```
input : Mobile Base Reference z^ref_B[0:E]
        Articulated Joint Reference q^ref_O[0:E]
output: Mobile Manipulator Velocity u
for i ← 0 to n_l do
    // get the predictive horizon
    z_B[1:N], u_B[1:N] ← MPC(z^ref_B[i:i+N]);
    for j ← 1 to N do
        // previous time step q_M[j−1] as initial
          guess;
        T_base ← Transform(z_B[j]);
        T_attach ← Transform(q^ref_O[i+j]);
        q_M[j] ← IK(T_base, T_handle);
        q̇_M[j] ←FirstOrderDiff(q_M[j])
    end
    u ← [u_B; q̇_M]
end
```

where d_a is the distance between the mobile base and the attachment location point on the target articulated object. d_{up} and d_{low} is the upper and lower bounds of the distance, $\boldsymbol{Q}_a$ is the related weighting matrix.

Furthermore, we separate the control command calculation for the mobile base and the manipulator in a lead-follower manner as detailed in Alg. 1. The cost function J in Eq. (3) is defined as $J = J^t + J^{\text{obs}} + J^{\pi} + J^a$, and only the mobile base $\boldsymbol{z}_B$ (*i.e.* first three elements in $\boldsymbol{z}$) is considered by the MPC framework. Then, the velocity command for the manipulator is computed by solving IK given the predicted base trajectory $\boldsymbol{z}_B$ and the planned object trajectory $\boldsymbol{q}_O^{\text{ref}}$.

IV. Simulation & Experiment

A. Simulation Setup

Our approach aims to provide safe planning and control for mobile manipulators to accomplish physical tasks in dynamic environments. To validate it, we set a simulation scenario in RVIZ of ROS to evaluate its performance. A mobile manipulator consisting of dual UR5e 6-DoF manipulators equipped with Robotiq 3-finger grippers as end-effectors and a Clearpath Ridgeback omnidirectional mobile base is utilized as the robot platform in both simulation and experiment. Of note, we only utilize one manipulator in this work, resulting in a 9-DoF system (3 for the base, 6 for the arm).

We consider a scenario where the robot is tasked to approach and open the closet door while a pedestrian is walking nearby. We simulate the pedestrian as a cylinder with the dimension of $(r, h) = (0.25, 2.0)\ m$. The robot's collision model is approximated with a series of spheres with dimensions of $0.4\ m$ for the base and $0.15\ m$ for the arm, as illustrated in Fig. 4. To implement the optimal control problem defined in Eq. (3a), the CppAD Ipopt Toolkit was utilized as the solver for the NMPC controller. The implementation parameters of the MPC algorithm are shown in Tab. I. In our simulation, we select a long prediction horizon ($5s$) to enhance long-term optimal control policy, providing heightened sensitivity towards dynamic cylinder obstacles.

TABLE I: MPC Implementation Parameters

Solver Parameter	Value	Weighting Parameter	Value
Prediction horzion N	$5\ s$	Base Position $\boldsymbol{Q}_z(1,2)$	10
Discretization timestep Δt	$0.05\ s$	Base Orientation $\boldsymbol{Q}_z(3)$	15
Jacobian matrix	Sparse	Arm Joint Angles $\boldsymbol{Q}_z(4:9)$	20
Maximum iteration	30	Linear Velocity $\boldsymbol{Q}_{\Delta u}(1,2)$	2
Maximum linear velocity (Pick)	$0.8\ m/s$	Angular Velocity $\boldsymbol{Q}_{\Delta u}(3:9)$	5
Maximum linear velocity (Place)	$0.5\ m/s$	Terminal Term $\boldsymbol{Q}_N$	$1e5$
Maximum angular velocity	$0.5\ rad/s$	Collision Pedestrian $\boldsymbol{Q}_p$	15
Control Rate	$20\ Hz$	Collision Planar $\boldsymbol{Q}_\pi$	9
		Attachment Term $\boldsymbol{Q}_a$	7.5

B. Simulation Results

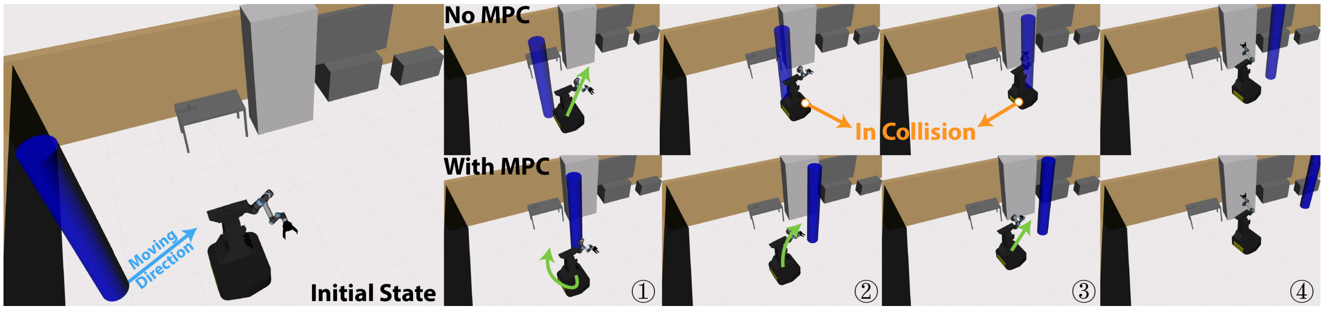
To prove the effectiveness of the proposed NMPC algorithm in avoiding possible collision with pedestrians, we design two challenging mobile manipulation tasks in conducting `pick` and `place`, and the results are summarized in Fig. 5.

`pick`: In Fig. 5a, the mobile manipulator is tasked to approach and grasp the handle of the cabinet while a pedestrian moves from $\boldsymbol{r}^o_{[0]} = (-0.8, 4.2)\ m$ with the velocity $\dot{\boldsymbol{r}}^o = (0.1, 0.35)\ m/s$. As shown in the first row of Fig. 5a, tracking the reference trajectory from the VKC-base planner without replanning, the pedestrian coincides with the mobile manipulator. Utilizing the proposed NMPC algorithm (second row of Fig. 5a), the robot successfully avoids the potential collision and accurately grasps the target position on the handle. The trajectory and velocity command generated by the NMPC controller is shown in Fig. 5c, while the distance from the robot and possible collision is shown in Fig. 5e. As we can see, there is a substantial change in velocity at $t = 12\ s$ (Fig. 5c) when the mobile manipulator accelerates to avoid the pedestrian. Despite aggressive behavior at this point, the velocity curve remains smooth and within the physical limits. Although the mobile manipulator violates the soft constraints, a safety distance is maintained between each link of the mobile manipulator and the pedestrian along the whole trajectory (Fig. 5e). The gripper's proximity to collision is registered at the end as the `pick` requires grasping the handle.

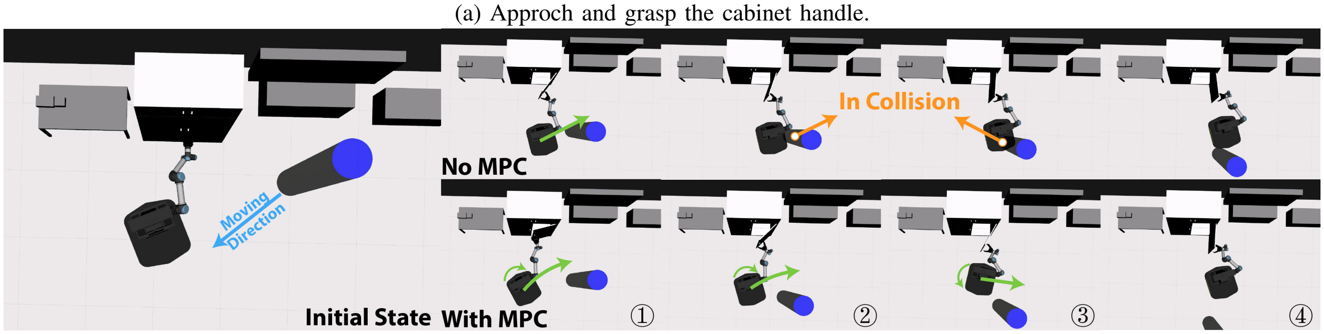
`place`: In Fig. 5b, a pedestrian starts moving from the position $\boldsymbol{r}^o_{[0]} = (0.9, -1.1)\ m$ with the velocity of $\dot{\boldsymbol{r}}^o = (-0.2, 0.1)\ m/s$. Similar to `pick`, without the proposed NMPC, the mobile manipulator would collide with the pedestrian (first row). The NMPC first predicts the trajectory of the mobile base, then solves the control inputs of the manipulator using IK, taking into account the door joint trajectory on the VKC (second row). Fig. 5f demonstrates that this method successfully achieves a collision-free execution, and both the velocity and position curves are smooth and within the hardware limits of the physical robot platform (Fig. 5d).

C. Experiment Setup

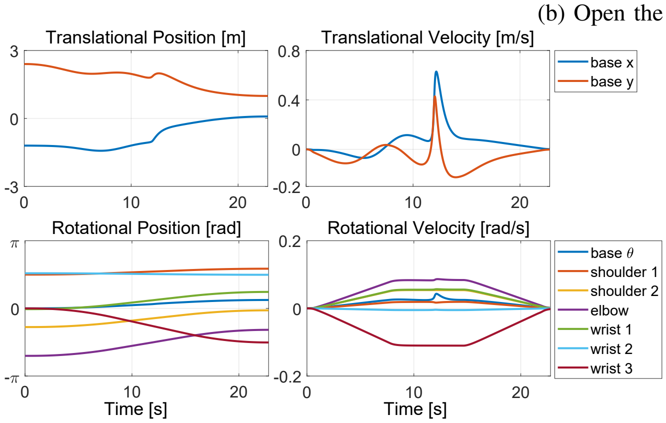
The same mobile manipulator platform was adopted for the experiment, where a Zotac Zbox PC with an Intel Core i7-10700 CPU is utilized as the local computing hardware. The Vicon MCS serves as the perception module for tracking object poses and the mobile base's position at $200\ Hz$. The mobile base is controlled with a PID controller at $100\ Hz$ to track the planned trajectory and the arm is controlled by its built-in controller. After planning with the proposed method, the planned trajectories are time-parameterized to add timestamps and velocity/acceleration values for robot control.



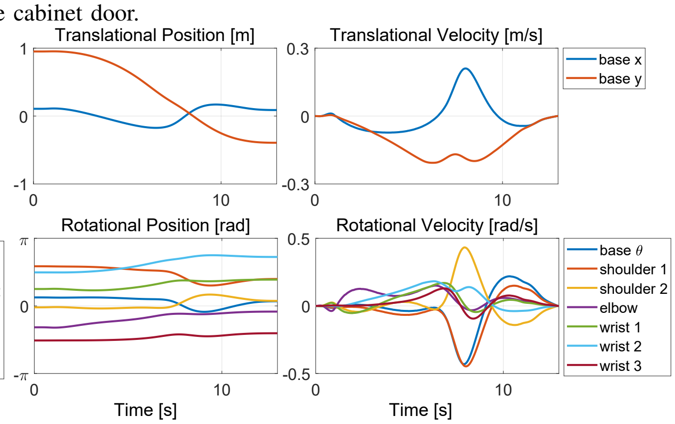
(a) Approch and grasp the cabinet handle.



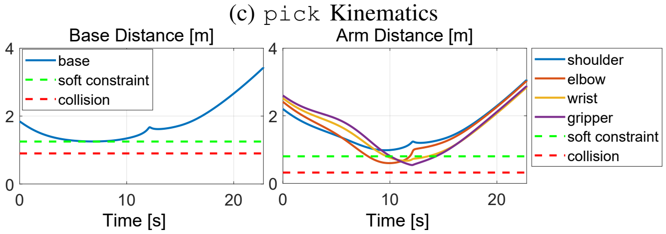
(b) Open the cabinet door.



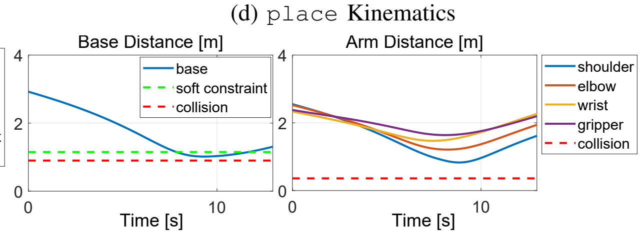
(c) `pick` Kinematics



(d) `place` Kinematics



(e) Distance to Pedestrian in `pick`



(f) Distance to Pedestrian in `place`

Fig. 5: Dynamic Obstacle Avoidance Results

As illustrated in the task setup in Fig. 6, our experiment scenario includes a cabinet, a tea table, a desk, and a tea can. At the initial environment state, the tea can is inside the closed cabinet, where the task goal given to the robot is placing the tea can on the tea table while the cabinet remains closed. We qualitatively evaluate the dynamic planning capability of the proposed framework by allowing human intervention during robot execution, such as moving the tea can away.

D. Experiment Results

The initial action plan generated by our framework is to first open the cabinet door, move the tea can to the tea table, and finally close the cabinet. During execution, the human relocates the tea can from the tea table to the nearby desk when the robot is closing the cabinet, as demonstrated in Fig. 6-③. The state monitor in our framework detects the environmental change after closing the cabinet and decides that a request for a new action sequence is required. The new sequence instructs the robot to move the tea can back to the tea table, and the second human intervention occurs when the robot is placing the tea can, with the cabinet being opened by a human, as shown in Fig. 6-⑤. The robot again successfully addresses the intervention and executes the new action sequence to fulfill the final task requirement at Fig. 6-⑦. Our experiment results demonstrate the capability of our dynamic SMMP framework to effectively respond to human intervention.

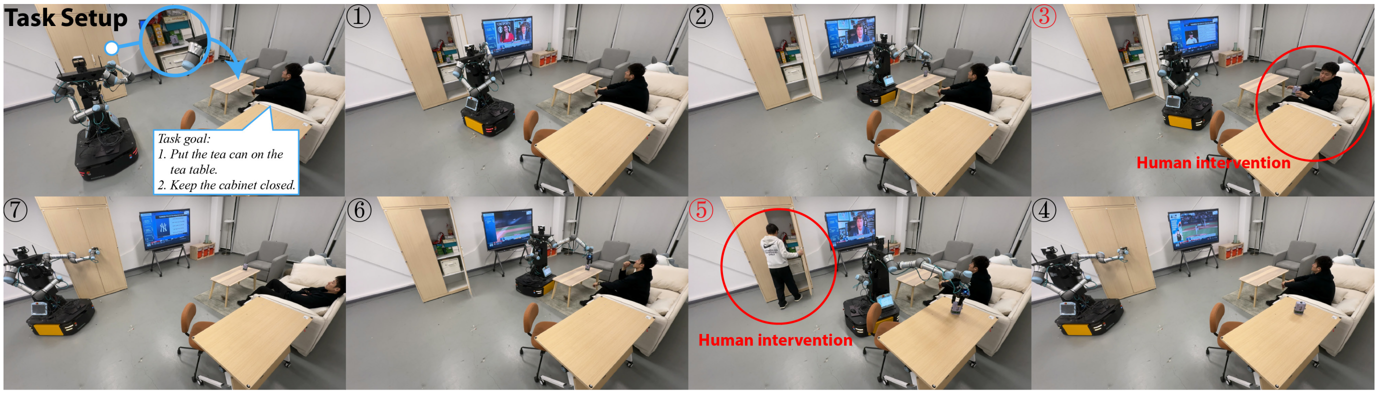


Fig. 6: Experiment setup and execution results.

V. Conclusion

In this paper, we present a holistic VKC-based dynamic SMMP framework achieving both task-level and motion-level dynamic planning under human activities. For the task level, we design the planning domain from the VKC perspective, implementing dynamic task replanning on a physical robot platform, and qualitatively verifying its efficacy through real-world experiments. For the motion level, we develop a new MPC approach for high-dimensional mobile manipulator dynamic collision avoidance in both interactive and non-interactive tasks. Additionally, we perform a quantitative study in simulations for pick and place actions to validate the effectiveness of the presented approach. In the future, we aim to fully deploy the proposed framework on a physical mobile manipulator platform, along with the non-MCS perception and localization module.

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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[2] 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.

[3] 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.

[4] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

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

[6] 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 *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[7] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, *et al.*, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.

[8] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-free mpc for legged robots in static and dynamic scenes," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[9] R. Luna, M. Moll, J. Badger, and L. E. Kavraki, "A scalable motion planner for high-dimensional kinematic systems," *International Journal of Robotics Research (IJRR)*, vol. 39, no. 4, pp. 361–388, 2020.

[10] B. Burgess-Limerick, J. Haviland, C. Lehnert, and P. Corke, "Reactive base control for on-the-move mobile manipulation in dynamic environments," *Robotics and Automation Letters (RA-L)*, 2024.

[11] J. Pankert and M. Hutter, "Perceptive model predictive control for continuous mobile manipulation," *Robotics and Automation Letters (RA-L)*, vol. 5, no. 4, pp. 6177–6184, 2020.

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

[13] M. Spahn, B. Brito, and J. Alonso-Mora, "Coupled mobile manipulation via trajectory optimization with free space decomposition," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[14] S. Zimmermann, R. Poranne, and S. Coros, "Go fetch!-dynamic grasps using boston dynamics spot with external robotic arm," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[15] M. P. Georgeff and A. L. Lansky, "Reactive reasoning and planning.," in *AAAI Conference on Artificial Intelligence (AAAI)*, vol. 87, pp. 677–682, 1987.

[16] M. Toussaint, J. Harris, J.-S. Ha, D. Driess, and W. Hönig, "Sequence-of-constraints mpc: Reactive timing-optimal control of sequential manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[17] P. S. Schmitt, F. Wirnshofer, K. M. Wurm, G. v. Wichert, and W. Burgard, "Planning reactive manipulation in dynamic environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[18] S. Li, D. Park, Y. Sung, J. A. Shah, and N. Roy, "Reactive task and motion planning under temporal logic specifications," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[19] 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.

[20] 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.

[21] 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.

[22] Z. He, J. Wu, J. Zhang, S. Zhang, Y. Shi, H. Liu, L. Sun, Y. Su, and X. Leng, "Cdm-mpc: An integrated dynamic planning and control framework for bipedal robots jumping," *Robotics and Automation Letters (RA-L)*, vol. 9, no. 7, pp. 6672–6679, 2024.

[23] Y. Su, Y. Jiang, Y. Zhu, and H. Liu, "Object gathering with a tethered robot duo," *Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 2132–2139, 2022.

[24] 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 *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[25] 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.

[26] K. Zhou, P. Wu, Y. Su, H. Gao, J. Ma, H. Liu, and C. Liu, "Aspire: An informative trajectory planner with mutual information approximation for target search and tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[27] Y. Su, J. Zhang, Z. Jiao, H. Li, M. Wang, and H. Liu, "Real-time dynamic-consistent motion planning for over-actuated uavs," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.