Task-Space Decomposed Motion Planning Framework for Multi-Robot Loco-Manipulation

Xiaoyu Zhang[1], Lei Yan[2], Tin Lun Lam[1,3], Sethu Vijayakumar[1,4]

***Abstract*— This paper introduces a novel task-space decomposed motion planning framework for multi-robot simultaneous locomotion and manipulation. When several manipulators hold an object, closed-chain kinematic constraints are formed, and it will make the motion planning problems challenging by inducing lower-dimensional singularities. Unfortunately, the constrained manifold will be even more complicated when the manipulators are equipped with mobile bases. We address the problem by introducing a dual-resolution motion planning framework which utilizes a convex task region decomposition method, with each resolution tuned to efficient computation for their respective roles. Concretely, this dual-resolution approach enables a global planner to explore the low-dimensional decomposed task-space regions toward the goal, then a local planner computes a path in high-dimensional constrained configuration space. We demonstrate the proposed method in several simulations, where the robot team transports the object toward the goal in the obstacle-rich environments.**

I. Introduction

Multi-robot system (MRS) distinguishes itself by promising solutions to manipulation tasks beyond the capability of a single robot, such as large or heavy object transportation. In terms of collaboration, each robot only needs to apply a portion of effort for the manipulation task. Additionally, with the natural characteristics of MRS, parallelism and decentralization, the physical separation and the independent actions of different robots can potentially generate a group of dexterity that a single robot can hardly achieve [1]. Besides, MRS is more resilient to failures, that is the mistake of one robot has less effect on the task completion [2], [3], [4].

Multi-robot collaborative loco-manipulation tasks always involve motion planning challenges in high-dimensional constrained configuration space (C-space). Fortunately, by decomposing C-space, we can use different representations for different subsets of the entire C-space, each suited to efficient computation to reduce overall computation time [5]. This idea has been demonstrated in several works, including complex motion planning for complex system [6], [7], [8]. McConachie et al. [7] presented a framework for deformable object manipulation that interleaved planning and control towards complex manipulation. Recently, in [8], Stouraitis et al. addressed dyadic collaborative manipulation tasks by using an efficient bi-level formulation that combines graph search methods with trajectory optimization. In this work, we decompose the constrained task space to serve the corresponding subsets of C-space for the mobile base and the manipulator and develop an efficient motion planning framework for multi-robot collaborative loco-manipulation

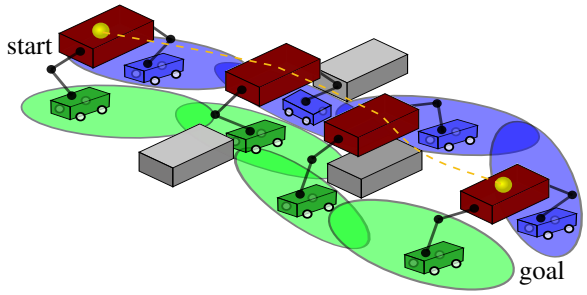

Fig. 1. A path from the start to the goal for multi-robot collaborative loco-manipulation utilizing decomposed task-space.

The space decomposition has been used in several works and showed excellent performance for MRS. In Kallem et al.'s work [9], the free workspace was decomposed into triangular regions, and the controller allows the robot to move from one cell to the next. Ayanian et al. [10] combined a triangulation of the environment with a navigation function to achieve multi-robot control. Alonso-Mora et al. [11] presented a method that allows a team of robots to navigate in formation via constrained nonlinear optimization. Their method first computes the largest obstacle-free convex polyhedron based on the work by Deits and Tedrake [12], then generates a trajectory to fit the formation in an overlapped convex region. Because of the limitation of convex optimization, the planner cannot deal with obstacles passing through the formation. We attempt to overcome this limitation by computing the decomposed convex task spaces, which are used for different subsets of the whole configuration.

In motion planning, sampling-based planning algorithms have been demonstrated their efficiency for searching high-dimensional C-space [13], [14]. Karaman et al. [15] extended their works to present asymptotically optimal motion planners, RRT* and PRM*. Moreover, some recent works achieved better performance by explicitly or implicitly guiding sampling [16], [17], [18]. However, efficient motion planning for multi-robot manipulation tasks requires not only to sample in collision-free C-space but also to comply with closed-chain kinematic constraints. As a result, C-space consists of a set of lower-dimensional points [19].

This work is supported by the Shenzhen Institute of Artificial Intelligence and Robotics for Society (2019-ICP002) and the EPSRC UK RAI Hub in Future AI and Robotics for Space (FAIR-SPACE: EP/R026092/1).

[1]Authors are with the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), Shenzhen, China.

[2]Authors are with School of Informatics, University of Edinburgh, Edinburgh, U.K.

[3]Author is with the Chinese University of Hong Kong, Shenzhen, China.

[4]Author is a visiting researcher with AIRS.

Corresponding author: Lei Yan (lei.yan@ed.ac.uk)

The most common approaches for sampling closed-chain constraints fall into two categories: direct sampling and projection methods. The former uses a parameterization of the constraint to generate samples in C-space. The Random Loop Generators (RLG) [19], [20] is effective direct sampling method. However, it is difficult to integrate with other simultaneous constraints and to extend to multi-mobile manipulators. The latter moves a sample into the constrained manifold by projection. One of the earliest projection-based sampling approaches dealing with closed-chain constraints relied on the Randomized Gradient Descent (RGD) [21]. RGD iteratively moved a random sample toward an arbitrarily constrained manifold. In [22], Stilman et al. demonstrated that RGD is significantly less efficient than Jacobian pseudo-inverse projection when it is extended to more general pose constraints. Berenson et al. [23] presented a Jacobian pseudo-inverse projection method based on Task Space Region (TSR), a more general constraint representation for manipulation planning task. Our approach is similar to TSR in relying on the Jacobian pseudo-inverse projection method to generate a valid configuration. We differ in generating valid configuration in smaller space by dynamically switching the decomposed task region instead of constraining the fixed task space region.

In this work, we present a novel task-space decomposed motion planning framework for multi-robot collaborative loco-manipulation. The central question we address is how to find motion sequences in constrained C-space for a team of mobile manipulators by exploiting different configuration subsets efficiently. The main contributions are:

1) *Constrained convex task-space decomposition*: compute and decompose obstacle-free convex task spaces for the different subsets of the entire configuration through convex optimization;
2) *Configuration generation in decomposed task space*: generate valid configurations for multi-mobile manipulators in different convex sub-spaces instead of searching in the entire configuration space directly, which can avoid to be trapped in undesirable local minima and improve computation efficiency;
3) *A dual-resolution motion planning framework for loco-manipulation*: enables efficient motion planning by exploring low-resolution task-space regions and then computing a path in the high-dimensional constrained C-space.

II. Problem Statement

In this paper, we focus on motion planning for multi-robot manipulation with multiple holonomic homogeneous mobile bases. The d-dimensional workspace are given by $\mathcal{W}_d \subset SE(d)$ contains static obstacles $\mathcal{O}_d \subset \mathcal{W}_d$. For a given robot i, for $i \in \{1, ..., N\}$, the configuration for mobile base and manipulator are denoted by $\boldsymbol{q}_b^i = \{q_1^i, ..., q_{M_b}^i\}$ and $\boldsymbol{q}_m^i = \{q_{M_b+1}^i, ..., q_M^i\}$. Therefore, the entire configuration of the robot i is $\boldsymbol{q}^i = \{\boldsymbol{q}_b^i, \boldsymbol{q}_m^i\}$. The volume occupied by robot i in the team of N robots with configuration $\boldsymbol{q}^i$ is $\mathcal{A}(\boldsymbol{q}^i) \subset \mathcal{W}_3$.

The obstacle region including the volumes occupied by the robots and the obstacles is denoted by $\mathcal{C}_{obs}$. Then, the obstacle-free region can be denoted by

$$\mathcal{C}_{free} = \mathcal{W}_3 \setminus \mathcal{C}_{obs}. \tag{1}$$

The collision free region for the object is denoted by $\mathcal{C}_{obj} \subset \mathcal{C}_{free}$. And we define a projection function $\boldsymbol{f}_p : \mathcal{W}_3 \to \mathcal{W}_2$, which realizes the projection from 3-dimensional workspace to 2-dimensional workspace as

$$\widehat{\mathcal{C}}_{obj} = \boldsymbol{f}_p(\mathcal{C}_{obj}). \tag{2}$$

Then, we define another decomposition function $\boldsymbol{f}_d : \mathcal{W}_2 \to \mathcal{W}_2$, which computes the largest convex obstacle-free region for the mobile base of each robot i:

$$\widehat{\mathcal{C}}_{rob}^i = \boldsymbol{f}_d(\widehat{\mathcal{C}}_{obj}, i). \tag{3}$$

Considering forward kinematic function $\boldsymbol{f}_{FK} : \mathbb{R}^M \to \mathcal{W}_3$, the pose of the end-effector is $\boldsymbol{x}_e^i = [\boldsymbol{p}_e^i, \boldsymbol{\Omega}_e^i]^T = \boldsymbol{f}_{FK}(\boldsymbol{q}^i) \in \mathcal{C}_{obj} \subset SE(3)$. Besides, the pose of object is denoted by $\boldsymbol{x}_{obj} = [\boldsymbol{p}_{obj}, \boldsymbol{\Omega}_{obj}]^T \in \mathcal{C}_{obj} \subset SE(3)$. We have $\boldsymbol{p}_{obj}^i$ as the grasping position on the object for the end-effector of robot i. When the end-effector and the object are in contact, we have $||\boldsymbol{p}_e^i - \boldsymbol{p}_{obj}^i|| = 0$. In this problem, since all end-effectors and the object should maintain contact, we can relax the constraint $\boldsymbol{x}_{obj} \in \mathcal{C}_{obj}$.

The problem we address in this paper is how to find a sequential motion from start configuration at t_s to goal configuration at t_g such that each motion is feasible. The feasible motion is that the planner should not bring either robot or object into collision with obstacles, and should not release the object. The problem can be summarized as:

$$\begin{aligned} \text{find} \quad & \boldsymbol{q}_t^i, \quad i \in [1, N], t \in [t_s, t_g] \\ \text{s.t} \quad & \boldsymbol{q}_{b,t}^i \in \widehat{\mathcal{C}}_{obj,t}^i \\ & \boldsymbol{x}_{e,t}^i \in \mathcal{C}_{obj,t} \\ & ||\boldsymbol{p}_{e,t}^i - \boldsymbol{p}_{obj,t}^i|| = 0 \\ & \mathcal{A}(\boldsymbol{q}^i) \cap \mathcal{A}(\boldsymbol{x}_{obj}) = \emptyset. \end{aligned} \tag{4}$$

III. Decomposed constrained motion planning

The section introduces our motion planning framework for multi-robot collaborative loco-manipulation, which is composed of task-space decomposition and constrained configuration generation. The details of the main procedure and the task-space decomposition are given in this section. In the next section, we will introduce how to sample in the constrained manifold with closed-chain kinematic constraints by taking advantage of the decomposed convex regions.

A. Method Overview

Our planner searches a valid path using a searching-based method. As shown in Fig.2, it can be summarized as:

1) The global planner searches for a valid path toward goal configuration, where the nodes of the path are presented by convex obstacle-free regions. And, these convex regions are defined as Decomposed Task-space Regions (DTRs).

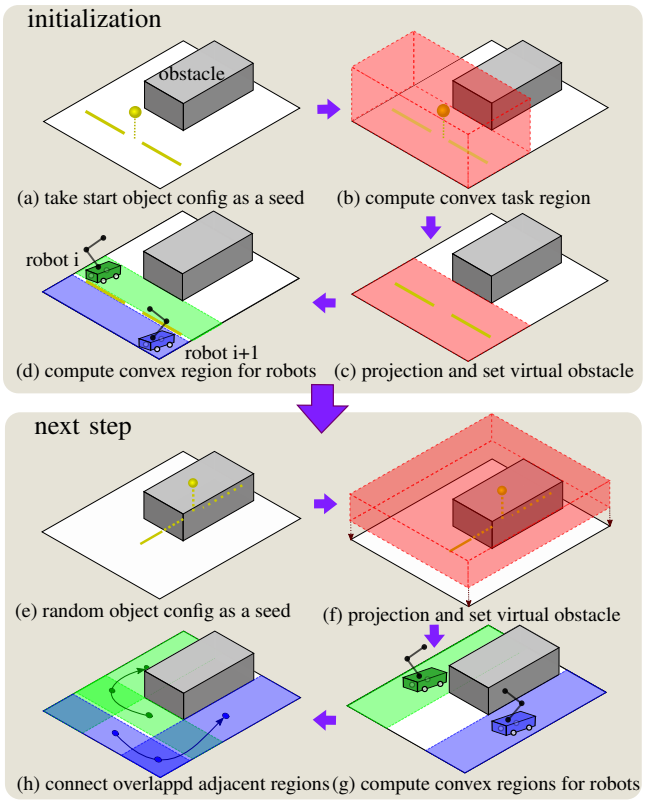

Fig. 2. A simple space decomposition for two robots with one rectangular obstacle in 3D environment. One iteration is demonstrated in the second block.

Algorithm 1 Main ($\boldsymbol{x}^s_{obj}$,$\boldsymbol{x}^g_{obj}$)

1: $\mathcal{P}^s \leftarrow SpaceDecomposition(\boldsymbol{x}^s_{obj})$
2: $\mathcal{P}^g \leftarrow SpaceDecomposition(\boldsymbol{x}^g_{obj})$
3: $\mathcal{T}^s.init(\mathcal{P}^s)$;$\mathcal{T}^g.init(\mathcal{P}^g)$
4: **while** TimeRemaining() **do**
5: $x_{rand} \leftarrow RandomTaskConfig(\mathcal{W}_3)$
6: $\mathcal{P}_{rand} \leftarrow SpaceDecomposition(x_{rand})$
7: $\mathcal{P}^s_{near}, \boldsymbol{k}^s \leftarrow NearestNeighbor(\mathcal{T}^s, \mathcal{P}_{rand})$
8: **if** $sum(\boldsymbol{k}^s) = N$ **then**
9: $\mathcal{P}^s_{reach} \leftarrow ConstrainedExtend(\mathcal{T}^s, \mathcal{P}^s_{near}, \boldsymbol{q_{rand}}, \mathcal{P}_{rand}, \boldsymbol{k}^s)$
10: $\mathcal{W}_3 \leftarrow \mathcal{W}_3 \backslash \mathcal{P}^s_{near}$
11: **else**
12: $\mathcal{P}^s_{reach} \leftarrow \mathcal{P}^s_{near}$
13: $\mathcal{P}^g_{near}, \boldsymbol{k}^g \leftarrow NearestNeighbor(\mathcal{T}^g, \mathcal{P}_{rand})$
14: **if** $sum(\boldsymbol{k}^g) = N$ **then**
15: $\mathcal{P}^g_{reach} \leftarrow ConstrainedExtend(\mathcal{T}^g, \mathcal{P}^g_{near}, \boldsymbol{q}_{rand}, \mathcal{P}_{rand}, \boldsymbol{k}^g)$
16: $\mathcal{W}_3 \leftarrow \mathcal{W}_3 \backslash \mathcal{P}^g_{near}$
17: **else**
18: $\mathcal{P}^g_{reach} \leftarrow \mathcal{P}^g_{near}$
19: **if** $\mathcal{P}^s_{reach} = \mathcal{P}^g_{reach}$ **then**
20: **return** $ExtractPath(\mathcal{T}^s, \mathcal{P}^s_{near}, \mathcal{T}^g, \mathcal{P}^g_{near})$
21: **else**
22: $Swap(\mathcal{T}^s, \mathcal{T}^g)$
23: **return** $\emptyset$

2) The local planner computes the edge between nodes by exploiting the entire C-space, including the manipulator's joint angles and the pose of mobile base. The C-space is constrained by closed-chain kinematic constraints and linear constraints resulted from DTRs.

Alg.1 shows the pseudocode of our main algorithm. Given a start and goal configuration of object $\boldsymbol{x}^s_{obj}$ and $\boldsymbol{x}^g_{obj}$, the planner is initialized by setting the root node of the two trees $\mathcal{T}^s$, $\mathcal{T}^g$ with DTRs $\mathcal{P}^s$, $\mathcal{P}^g$, which contain convex regions for mobile bases of each robot (blue and green regions in Fig.2(d)) and task region for an object (red region in Fig.2(b)), where for each DTR we have $\mathcal{P} = \{\mathcal{P}^1_{rob}, ..., \mathcal{P}^N_{rob}, \mathcal{P}_{task}\}$. More details about *SpaceDecomposition* is in Sec. IV-B. Afterward, two trees start to grow until either time elapsed or maximum iterations are reached. The *RandomTaskConfig* function generates a random sampled configuration $\boldsymbol{x}_{rand}$ in task space (yellow point in Fig.2(a)), which is utilized to compute a random DTR that is $\mathcal{P}_{rand}$. Then, the *NearestNeighbor* finds not only the neighbor with minimum $d_{\mathcal{P}}$, but also defines a $N \times 1$ vector of binary connection status $\boldsymbol{k}$ between two adjacent DTRs. $d_{\mathcal{P}}$ is the distance metric for any two adjacent DTRs:

$$d_{\mathcal{P}} = \sum_{i=1}^{N} ||\boldsymbol{p}_{\mathcal{P}^i_1} - \boldsymbol{p}_{\mathcal{P}^i_2}||, \tag{5}$$

where $\boldsymbol{p}_{\mathcal{P}^i}$ is the centre position of convex region of robot i in DTR. When $sum(\boldsymbol{k}) = N$, the robot region of $\mathcal{P}_{near}$ and $\mathcal{P}_{rand}$ are fully connected (all robot regions are connected). In this case, we can compute constrained local path to connect region $\mathcal{P}_{near}$ and $\mathcal{P}_{rand}$ by *ConstrainedExtend* method (as shown in Fig.2(h), details in Sec.V). Moreover, the planner will also shrink the workspace by subtracting $\mathcal{P}_{near}$ from $\mathcal{W}_3$. As a result, the searching space will be smaller. Otherwise, when $sum(\boldsymbol{k}) < N$, two regions cannot be connected with the current configuration, and the growing process will not be performed. Lines 13-18 in Alg.1 correspond to the conventional RRT-Connect approach [16], which grow another tree $\mathcal{T}^g$ with root of goal node to obtain $\mathcal{P}^g_{reach}$. Lastly, when two trees meet each other, we extract and connect the path from the start node to the goal node. If not, two trees are swapped, and the above process is repeated. Note that the start configuration $\boldsymbol{q}_s$ and goal configuration $\boldsymbol{q}_g$ of the robot team corresponding to $\boldsymbol{x}^s_{obj}$ and $\boldsymbol{x}^g_{obj}$ can be computed through inverse kinematics.

B. Space Decomposition

The idea central to our *space decomposition* algorithm (Alg.2) is to compute convex polyhedrons, which are represented by linear constraints, for different subsets of the entire configuration. Our method, which is built on top of the fast iterative method IRIS [12], can further reduce space complexity and avoid deadlock in the local planner (more details in Sec.V).

This method takes an arbitrary configuration $\boldsymbol{x}_{obj}$ as input (Fig.2.(a)). Thereafter, our method computes 3-dimensional (3D) task space $\mathcal{P}^3_{task}$ (red convex polyhedron in Fig.2.(b)), which is restricted by obstacle polyhedron $\mathcal{P}_{obs}$ (gray block

Algorithm 2 SpaceDecomposition ($\boldsymbol{x}_{obj}$)

$\boldsymbol{p}_{obj}, \boldsymbol{\Omega}_{obj} \leftarrow \boldsymbol{x}_{obj}; \mathcal{P} \leftarrow \emptyset$
$\mathcal{P}^3_{task} \leftarrow IRIS(\boldsymbol{p}_{obj}, \mathcal{P}_{obs}, \mathcal{P}_{bound})$
$\mathcal{P}^2_{task} \leftarrow Polytope2DProjection(\mathcal{P}^3_{task})$
$\mathcal{P}_{vir} \leftarrow SetVirtualObstacle(\boldsymbol{\Omega}_{obj}, \mathcal{P}^2_{task}, \text{N})$
for $i = 1$ to N **do**
  $\boldsymbol{p}_{rand} \leftarrow GetSeed(\boldsymbol{\Omega}_{obj}, \mathcal{P}^2_{task})$
  $\mathcal{P}[i] \leftarrow IRIS(\boldsymbol{p}_{rand}, \mathcal{P}_{Vir}, \mathcal{P}^2_{task})$
return $\mathcal{P}$

in Fig.2), and bound $\mathcal{P}_{bound}$. The bound $\mathcal{P}_{bound}$ can be limited by the maximum allowable distance regarding the robot team. Because the mobile base operates on $\mathcal{W}_2$, we compute its corresponding 2D task-space $\mathcal{P}^2_{task}$ by projecting $\mathcal{P}^3_{task}$ onto $\mathcal{W}_2$ (red convex polygon in Fig.2.(c)). Then, we divide $\mathcal{P}^2_{task}$ into N sub-spaces for each robot by setting N virtual obstacles $\mathcal{P}_{vir}$ and a arbitrary position $\boldsymbol{p}_{rand}$ as seed. Each virtual obstacle is a vector from the center of 2D task-space $\mathcal{P}^2_{task}$ to the boundary of $\mathcal{P}^2_{task}$ with a given angle, which is $i * 2\pi/N$. The seed $\boldsymbol{p}_{rand}$ is an arbitrary point in the corresponding sub-space. A simple example of this decomposition process is shown in Fig.2 for two robots.

IV. Constrained Configuration Generation

To achieve the manipulation task, a local planner is designed to connect each region through C-space. In addition to the convex polyhedron represented by its equivalent linear constraints, there is also a closed-chain kinematic constraint, which induces parts of varying dimensionality in C-space. Therefore, we consider the sample projection approach [21], which has already demonstrated efficiency to generate the samples in the closed-chain constrained manifold. Additionally, we extend the idea of constraining task region by the fixed boundary to dynamically switching bounds according to given $\mathcal{P}^2_{task}$ and $\mathcal{P}^3_{task}$ [23]. Then, we introduce a projection-based sampling approach for multi-mobile manipulators under closed-chain kinematic constraints.

A. Notations

Throughout this section, $\mathcal{F}^t_i$ represents the *task frame* of robot i, which is also its end-effector frame e. The transformation matrix $\boldsymbol{T}^1_2$ consists of a 3×3 rotation matrix $\boldsymbol{R}^1_2$ and a 3×1 displacement vector $\boldsymbol{p}^1_2$. The following three transformation matrices are defined for robot i:

1) $\boldsymbol{T}^0_{e,i}$: transformation matrix of the end-effector frame with respect to the origin;
2) $\boldsymbol{T}^{b,i}_{e,i}$: transformation matrix of the end-effector frame with respect to the base of manipulator;
3) $\boldsymbol{T}^{e,i}_{e',i}$: offset in the end-effector frame e.

We assume the base frame b is where the connection between the manipulator and mobile base.

B. Satisfaction of closed-chain constraints

In this section, we demonstrate how to generate a valid random configuration $\boldsymbol{q}_{rand}$ using a projection strategy. This process is similar to the TSR framework [23], where the closed-chain constraints are encoded as pose constraints.

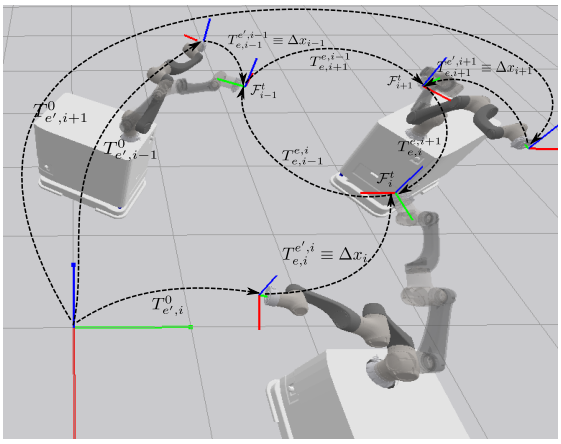

Fig. 3. Transform and frames for three adjacent end-effectors. The transparent manipulators are with desired pose.

Considering two adjacent manipulators in Fig.3, the frames of their end-effectors are denoted as $\mathcal{F}^t_i$ and $\mathcal{F}^t_{i+1}$. Then, we define $\boldsymbol{T}^{e,i+1}_{e,i}$, which represents the offset transformation from the frame $\mathcal{F}^t_{i+1}$ to $\mathcal{F}^t_i$. When robots are manipulating an object, the end-effector should be always connected with the object. Therefore, $\boldsymbol{T}^{e,i+1}_{e,i}$ will be constant, and it can be easily computed from initial configuration when two manipulators formed a closed-chain constraint. We assume $\boldsymbol{T}^0_{e,i+1}$ is fixed. Then, given a random configuration $\boldsymbol{q}^i_{a,rand}$, we use forward kinematics to get the current pose of end-effector $\boldsymbol{T}^0_{e',i}$. The desired pose $\boldsymbol{T}^0_{e,i}$ of end-effector $\mathcal{F}^t_i$ can be obtained from $\boldsymbol{T}^{e,i+1}_{e,i}$ and $\boldsymbol{T}^0_{e,i+1}$ as

$$\boldsymbol{T}^0_{e,i} = \boldsymbol{T}^0_{e,i+1} \boldsymbol{T}^{e,i+1}_{e,i}. \tag{6}$$

The error transformation matrix is obtained as:

$$\boldsymbol{T}^{e',i}_{e,i} = (\boldsymbol{T}^0_{e',i})^{-1} \boldsymbol{T}^0_{e,i}. \tag{7}$$

Next, we calculate the orientation error from the rotation matrix $\boldsymbol{R}^{e',i}_{e,i}$ as the following RPY representation:

$$\boldsymbol{\Omega}^{e',i}_{e,i} = \begin{bmatrix} atan2(\boldsymbol{R}^{e',i}_{e,i_{32}}, \boldsymbol{R}^{e',i}_{e,i_{33}}) \\ -atan2(\boldsymbol{R}^{e',i}_{e,i_{31}}) \\ atan2(\boldsymbol{R}^{e',i}_{e,i_{21}}, \boldsymbol{R}^{e',i}_{e,i_{11}}) \end{bmatrix}. \tag{8}$$

Then, the distance between frame $\mathcal{F}^t_i$ and $\mathcal{F}^t_{i'}$ is:

$$\Delta \boldsymbol{x}_i \equiv \boldsymbol{T}^{e',i}_{e,i} \equiv \begin{bmatrix} \boldsymbol{p}^{e',i}_{e,i} \\ \boldsymbol{\Omega}^{e',i}_{e,i} \end{bmatrix}. \tag{9}$$

Finally, once $\Delta \boldsymbol{x}_i$ is obtained, the end-effector of robot i will move to obey the closed-chain constraint by using the pseudo-inverse of Jacobian $\boldsymbol{J}^\dagger_i$ which can map the task-space error $\Delta \boldsymbol{x}_i$ to the joint velocity:

$$\dot{\boldsymbol{q}}^i_m = \boldsymbol{J}^\dagger_i \Delta \boldsymbol{x}_i. \tag{10}$$

C. Closed-chain constraints for multi-mobile manipulators

Next, we extend the sample projection approach to multi-robot system with N mobile manipulators, that is *GenerateValidConfig* method in Alg.3. By randomly sampling a configuration $\boldsymbol{q}^i_{m,rand}$ for each manipulator, the closed-chain constraint will not be naturally satisfied. We define the error function as:

$$d_e = \sum_{i=1}^{N} ||\Delta \boldsymbol{x}_i||. \tag{11}$$

Then, our method iteratively move the pose to the constraint manifold (as shown in previous section) until $d_e \simeq 0$. However, robots may reach singularity if two robots are too far away from each other. Therefore, another joint velocity is defined for mobile base as:

$$\dot{\boldsymbol{q}}^i_b = \left[\boldsymbol{p}^{e,i}_{e',i}[1],\ \boldsymbol{p}^{e,i}_{e',i}[2],\ \boldsymbol{\Omega}^{e',i}_{e,i}[3]\right]^T, \tag{12}$$

where $\boldsymbol{p}^{e,i}_{e',i}[1:2]$ is the relative position between two task frames in x-y plane, and $\boldsymbol{\Omega}^{e',i}_{e,i}[3]$ is the relative orientation about z-axis. $\dot{\boldsymbol{q}}^i_b$ is the velocity of the mobile base to escape singularity. Then, by combining (10) and (12), we have the projected velocity for the whole configuration:

$$\dot{\boldsymbol{q}}^i = \left[\dot{\boldsymbol{q}}^{i\,T}_b\ \dot{\boldsymbol{q}}^{i\,T}_m\right]^T. \tag{13}$$

Note that if the region $\mathcal{P}^2_{task}$ is a non-convex region, there will be either deadlock or require a long time to find a constrained configuration because of local minimum. Since we have already obtained an obstacle-free convex region for each mobile base by the proposed task-space decomposition algorithm, the convergence time to get a valid configuration is significantly decreased.

D. Constrained Extend

Alg.3 tries to establish a connection between the two configurations, which can be done by incrementally stepping from $\boldsymbol{q}_{near}$ to $\boldsymbol{q}_{rand}$. The $ConstrinedConfig$ function use the projection method introduced in Sec.V-C to project $\boldsymbol{q}_{ext'}$ onto the constrained manifold. As a result, Alg.3 returns a list of valid configurations. When every two adjacent regions are fully connected, the sum of $\boldsymbol{k}$ equals to the robot team's size N, we can simply get a valid configuration sequences in given convex obstacle-free regions. When the adjacent regions are not fully connected, where $sum(\boldsymbol{k}) < N$, we can use a search-based algorithm to get the corresponding configuration for remaining robots, i.e., RRT, PRM, or A*. In practice, as long as we keep running a global planner, the connected regions can always be found.

Algorithm 3 ConstrainedExtend($\mathcal{T}, \mathcal{P}_{near}, \boldsymbol{q}_{b,rand}, \mathcal{P}_{rand}, \boldsymbol{k}$)

$\mathcal{V} \leftarrow \emptyset$; $valid\_config \leftarrow TRUE$
$\boldsymbol{q}_{near} \leftarrow GenerateValidConfig(\boldsymbol{q}_{b,rand}, \mathcal{P}_{near})$
while $valid\_config$ **do**
  $\boldsymbol{q}_{ext'} \leftarrow StepToward(\boldsymbol{q}_{rand}, \boldsymbol{q}_{near})$
  $\boldsymbol{q}_{ext} \leftarrow ConstrinedConfig(\boldsymbol{q}_{ext'}, \mathcal{P}_{rand}, \mathcal{P}_{near}, \boldsymbol{k})$
  if $\boldsymbol{q}_{ext} = None$ **then**
    $valid\_config \leftarrow FALSE$
  $\mathcal{V} \leftarrow \boldsymbol{q}_{ext}$
  if $\boldsymbol{q}_{ext} = \boldsymbol{q}_{rand}$ **then**
    $break$
return $\mathcal{V}$

V. Experiments

We validate our method through several different simulations with the purpose of testing the performance of collaborative object transportation. These simulations include the scenario that MRS needs to pass through the interior door, long-narrow passage and obstacle-rich environment, respectively. All the simulations are carried out in the Bullet physics simulation environment with a team of nine DOFs (three DOFs mobile base and six DOFs manipulator) omni-directional AIRS lab mobile platforms. All experiments were executed within five minutes on a desktop with an Intel Core i7-9700 3.0GHz and 32GB 3000MHz RAM.

In the first simulation, we demonstrate that two mobile manipulators pass through an interior door and a long-narrow passage while maintaining contact with the object in order to transport it from an initial position to the desired position. As shown in Fig.4, to pass through the interior door, the robot team needs inevitability to reconfigure to avoid self-collision and stay in their convex region to avoid collision with obstacles. The long-narrow passage is used for two purposes. First, it demonstrates that the robot team can find a collision-free passage over the obstacle. Second, it can also evaluate the proposed method's effectiveness and efficiency of passing through the long and narrow passage as most of the sampling-based motion planning methods might have less performance in this scenario. Fig.4.(b)-(e) are four snapshots that show the configurations of the robot team satisfy the constraints while moving toward the goal. As shown in Fig.4.(c) and (d), the proposed method can enable MRS to change its base formation and arm configuration to pass through the interior door and the long-narrow passage while avoiding collision with the obstacle-rich environment.

Furthermore, we demonstrated that our method could also work in a team of different size. As shown in Fig.5, three mobile manipulators are adopted to transport the object from start to goal while avoiding obstacles and maintaining closed-chain constraint in the obstacle-rich environment. The obstacle-rich environment can also evaluate the computation performance of convex region. As shown in Fig.5.(b)-(d), our planner can enable the robot team to manipulate the ball without collision with several obstacles by reconfiguration and exploiting different convex regions.

In all these experiments, the robot team successfully achieve simultaneous locomotion and manipulation by re-configuration according to different scenarios. In TABLE I, the average planning time with standard deviation over 100 trials is highlighted and the highest and lowest ten trials are removed. Each trial is collected within 3000 iterations and the final path is smoothed using the short-cut-smooth method within 300 iterations. We then compared the execu-

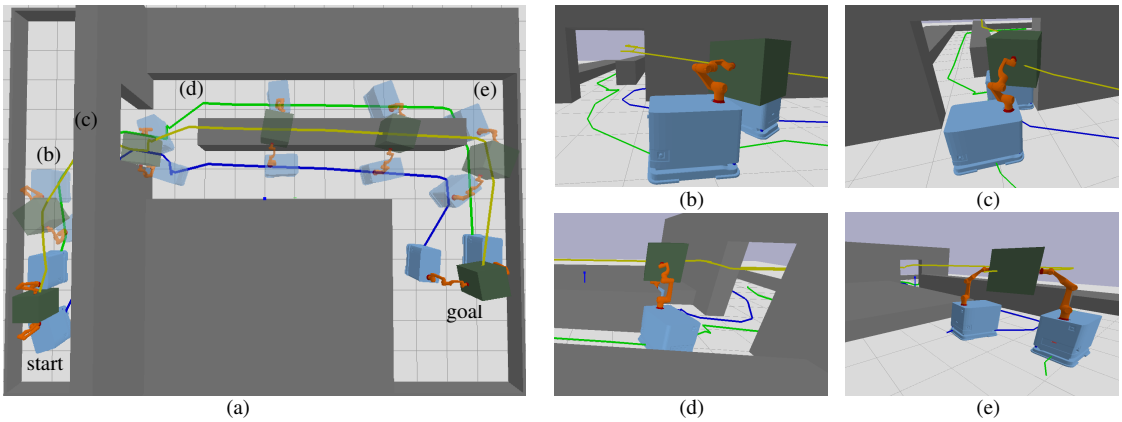

Fig. 4. Experiment 1: (a) Keyframes of multi-robot collaborative transportation, (b) and (c) are configurations that pass through the interior door, (d) and (e) are configurations that pass through narrow passage and allow obstacle through robot team.

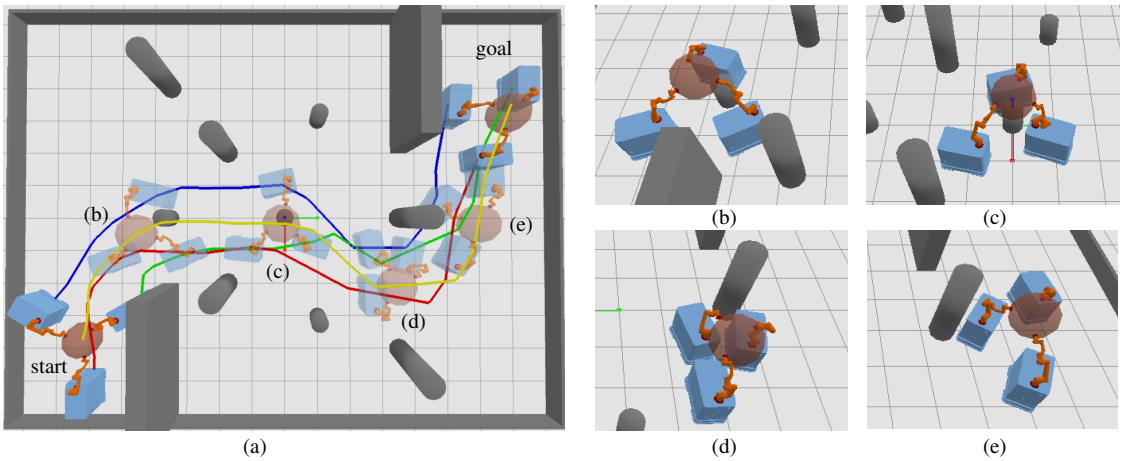

Fig. 5. Experiment 2: (a) Keyframes of multi-robot collaborative transportation. (b-e) are configurations that pass through obstacles of different heights.

TABLE I

THE TIME COST WITH STANDARD DEVIATIONS (IN SECOND). THE RESULT IS AVERAGED OVER 100 RUNS FOR EACH SCENARIO.

Scenario	Method	Space Decomposition	planning time	Total
1: Fig.4	RLG	-	195.3±169.1	195.3
1: Fig.4	TSR	-	137.0±92.6	137.0
1: Fig.4	Ours	26.1±17.4	37.5±14.9	63.6
2: Fig.5	RLG	-	171.4±94.1	171.4
2: Fig.5	TSR	-	118.12±67.7	118.12
2: Fig.5	Ours	32.9±22.5	47.2±25.1	80.1

tion time of our planner against the RRT-connect algorithm using RLG and TSR. We used the same parameter for the RRT-connect algorithm. Our planner exhibits faster planning time compared with the other methods in both experiments. The improvements over the closest competitor TSR are significant, which is approximately decreased by 54% and 32%. Furthermore, the results also show that our planner is more stable as we have smaller standard deviations. Therefore, this result indicates that with the local boundary of decomposed convex regions, dynamic workspace shrinking, and local minimum avoidance for the local planner, our planner is faster and more stable. Our method might be less efficient in the obstacle-rich environment, as it weakens the exploring ability of *SpaceDecomposition*, and more computation time for more robots. Note that the exploring process of our method is done by both searching-based frameworks explicitly and *SpaceDecomposition* implicitly. Besides, using parallel programming might further improve our planner's performance. According to [11], their approach cannot accomplish such dexterous manipulation in $SE(3)$. Our method's overall result is effective and efficient for performing simultaneous locomotion and manipulation tasks in an obstacle-rich environment.

VI. CONCLUSION

This paper presented a novel task-space decomposed motion planning framework for multi-robot collaborative loco-manipulation under closed-chain constraint. By decomposing the constrained task-space into different convex regions in a global planner, the local planner can efficiently generate a valid configuration for a multi-mobile manipulator system without deadlock. We evaluated the method in two obstacle-rich simulation environments, including narrow passages and plenty of obstacles. The results demonstrate that the proposed method enables a team of mobile manipulators to reconfigure their configurations for obstacle avoidance while carrying an object toward the goal. In future work, we plan to extend our method for articulated object loco-manipulation with a heterogeneous multi-mobile manipulator system.

REFERENCES

[1] R. G. Brown and J. S. Jennings, "A pusher/steerer model for strongly cooperative mobile robot manipulation," in *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 1995, pp. 562–568.

[2] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 220–240, 1998.

[3] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[4] L. Yan, T. Stouraitis, and S. Vijayakumar, "Decentralized ability-aware adaptive control for multi-robot collaborative manipulation," *IEEE Robotics and Automation Letters*, 2021.

[5] O. Brock and L. E. Kavraki, "Decomposition-based motion planning: A framework for real-time motion planning in high-dimensional configuration spaces," in *2001 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, 2001, pp. 1469–1474.

[6] V. Ivan, D. Zarubin, M. Toussaint, T. Komura, and S. Vijayakumar, "Topology-based representations for motion planning and generalization in dynamic environments with interactions," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1151–1163, 2013.

[7] D. McConachie, A. Dobson, M. Ruan, and D. Berenson, "Manipulating deformable objects by interleaving prediction, planning, and control," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 957–982, 2020.

[8] T. Stouraitis, I. Chatzinikolaidis, M. Gienger, and S. Vijayakumar, "Online hybrid motion planning for dyadic collaborative manipulation via bilevel optimization," *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1452–1471, 2020.

[9] V. Kallem, A. T. Komoroski, and V. Kumar, "Sequential composition for navigating a nonholonomic cart in the presence of obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1152–1159, 2011.

[10] N. Ayanian, V. Kallem, and V. Kumar, "Synthesis of feedback controllers for multiple aerial robots with geometric constraints," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 3126–3131.

[11] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[12] R. Deits and R. Tedrake, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic foundations of robotics XI*, 2015, pp. 109–124.

[13] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[14] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[15] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[16] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *2000 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, 2000, pp. 995–1001.

[17] J. Denny, R. Sandström, A. Bregger, and N. M. Amato, "Dynamic region-biased rapidly-exploring random trees," in *Algorithmic Foundations of Robotics XII*, 2020, pp. 640–655.

[18] B. Ichter, J. Harrison, and M. Pavone, "Learning sampling distributions for robot motion planning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7087–7094.

[19] M. Gharbi, J. Cortés, and T. Simeon, "A sampling-based path planner for dual-arm manipulation," in *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, 2008, pp. 383–388.

[20] J. Cortés and T. Siméon, "Sampling-based motion planning under kinematic loop-closure constraints," in *Algorithmic Foundations of Robotics VI*, 2004, pp. 75–90.

[21] J. H. Yakey, S. M. LaValle, and L. E. Kavraki, "Randomized path planning for linkages with closed kinematic chains," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 951–958, 2001.

[22] M. Stilman, "Task constrained motion planning in robot joint space," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 3074–3081.

[23] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *The International Journal of Robotics Research*, vol. 30, no. 12, pp. 1435–1460, 2011.