Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

Collaborative Planar Pushing of Polytopic Objects with Multiple Robots in Complex Scenes

Zili Tang, Yuming Feng and Meng Guo*
College of Engineering, Peking University

***Abstract*—Pushing is a simple yet effective skill for robots to interact with and further change the environment. Related work has been mostly focused on utilizing it as a non-prehensile manipulation primitive for a robotic manipulator. However, it can also be beneficial for low-cost mobile robots that are not equipped with a manipulator. This work tackles the general problem of controlling a team of mobile robots to push collaboratively polytopic objects within complex obstacle-cluttered environments. It incorporates several characteristic challenges for contact-rich tasks such as the hybrid switching among different contact modes and under-actuation due to constrained contact forces. The proposed method is based on hybrid optimization over a sequence of possible modes and the associated pushing forces, where (i) a set of sufficient modes is generated with a multi-directional feasibility estimation, based on quasi-static analyses for general objects and any number of robots; (ii) a hierarchical hybrid search algorithm is designed to iteratively decompose the navigation path via arc segments and select the optimal parameterized mode; and (iii) a nonlinear model predictive controller is proposed to track the desired pushing velocities adaptively online for each robot. The proposed framework is complete under mild assumptions. Its efficiency and effectiveness are validated in high-fidelity simulations and hardware experiments. Robustness to motion and actuation uncertainties is also demonstrated. More examples and videos are available at https://zilitang.github.io/Collaborative-Pushing.**

I. Introduction

Non-prehensile manipulation skills such as pushing are essential when humans interact with objects, as an important complementary to prehensile skills such as stable grasping. Existing work can be found that exploits this aspect for robotic systems, mostly for a single manipulator within simple environments, e.g., Goyal et al. [8], Lynch et al. [19], Hogan and Rodriguez [11], Xue et al. [32]. However, pushing can also be beneficial for low-cost mobile robots that are not equipped with a manipulator, e.g., obstacles could be pushed away, or target objects could be pushed to destinations. As illustrated in Fig. 1, a fleet of such robots could be more efficient by collaboratively and simultaneously pushing the same object at different points with different forces. Theses combinations generate a rich set of pushing modes, e.g., long-side, short-side, diagonal, and caging, which lead to different motions of the object such translation and rotation, however with varying quality. Despite of its intuitiveness, the collaborative pushing task incorporates several challenges that are typical for contact-rich tasks, i.e., the hybrid dynamic system due to switching between different contact modes, under-actuation due to constrained contact forces, and tight kinematic and geometric constraints (such as narrow passages and sharp turns), yielding it a difficult problem not only for modeling and planning, but also for optimal control.

*Corresponding author: Meng Guo, meng.guo@pku.edu.cn.

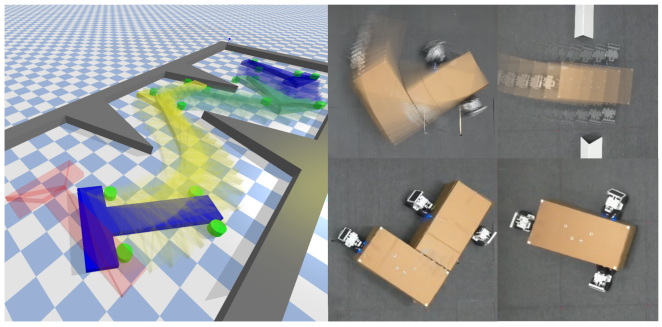

Fig. 1: Snapshots of the collaborative pushing task in simulation and hardware experiments.

A. Related Work

The task and motion planning problem for prehensile manipulation is the predominant research direction with a large body of literature. It has a strong focus on the grasping policies for different objects, and the sequence of manipulating these objects for a long-term manipulation task, such as assembly in Toussaint [24], Guo and Bürger [9], re-arrangement in Kim et al. [14] and construction in Hartmann et al. [10]. On the other hand, non-prehensile planar manipulation is a challenging problem on its own for model-based planning and control without stable grasping. The *single-pusher-single-slider* system has been introduced as the classic model from Goyal et al. [8], where both the discrete decision for interaction modes and continuous optimization for control inputs are essential. However, the inclusion of discrete variables drastically increases the planning complexity, i.e., exponential to the planning horizon. For a small number of pre-defined modes such as sticking and sliding, this decision can be directly formulated as an integer variable in the motion planning scheme from Hogan and Rodriguez [11] thus solved via integer programs. In addition, various search methods can be applied to explore the statespace e.g., via sampling in Wang et al. [29], multi-bound tree search in Toussaint et al. [25], and backtracking in Kaelbling and Lozano-Pérez [12]. To further alleviate the complexity, human demonstrations are used in Xue et al. [32], Le et al. [17] to guide the choice of contact modes, while learning-based methods are proposed to predict the contact

sequence in Simeonov et al. [21] or even raw inputs in Zeng et al. [33] for short-term tasks. However, these methods are mostly developed for a single manipulator and fix-shaped objects, of which the control and planning techniques are not applicable to multi-robot fleets.

Collaborative pushing belongs to a larger application domain of multi-robot systems: cooperative object transportation, see Tuci et al. [26], which includes pushing, grasping and caging as different behaviors. The pioneer work in Kube [15] designs a multi-layer state machine for 5 robots to push a rectangular box without direct communication in free space. A simple yet effective strategy is proposed in Chen et al. [4] to transport any convex object in obstacle-free space, i.e., the robots only push the object at positions where the direct line to the goal is occluded by the object. A combinatorial-hybrid optimization problem is formulated in Tang et al. [22] for several objects but with pre-defined pushing primitives. Fleets of heterogeneous robots are deployed in Vig and Adams [27] to clear movable obstacles via pushing, however via a pre-defined sequence of steps. Model-free approaches can be found in Xiao et al. [31] that map directly the scene to control inputs but lacks the theoretical guarantees on completeness. Other related work neglects the control aspect and focuses only on the high-level task assignment, e.g., García et al. [7].

B. Our Method

This work addresses the general problem of controlling a team of mobile robots to push collaboratively a polytopic object to a goal position in a complex environment, without any pre-defined contact modes or primitives. To begin with, a multi-directional feasibility analysis is conducted for a quasi-static pushing condition, given the desired object motion and a parameterized contact mode. Then, the notion of sufficient contact modes is introduced for pushing the object along a desired trajectory, based on its finite segmentation into arcs. Thus, a hierarchical hybrid search algorithm is proposed for finding a feasible sequence of modes and the associated force profiles. It iteratively decomposes the desired trajectory via key-frames and expands the search tree via generating suitable modes between consecutive frames, in order to minimize the aforementioned feasibility loss, transition cost and control efforts. Lastly, a nonlinear model predictive controller (MPC) is proposed to track these arc segments online with the given mode, while accounting for motion uncertainties and undesired slips adaptively. Theoretical guarantees on completeness and performance are provided. Extensive high-fidelity simulations and hardware experiments are conducted to validate its efficiency and robustness.

Main contribution of this work is two-fold: (i) the novel hierarchical hybrid-search algorithm for the multi-robot planar pushing problem of general polytopic objects, without any pre-defined contact modes or primitives; (ii) the theoretical condition for feasibility and completeness, w.r.t. an arbitrary number of robots, any polytopic shape and any desired object trajectory. To the best of our knowledge, this is the first work that provides such results.

II. Problem Description

A. Model of Workspace and Robots

Consider a team of N robots $\mathcal{R} \triangleq \{R_n,\, n \in \mathcal{N}\}$ that collaborate in a shared 2D workspace $\mathcal{W} \subset \mathbb{R}^2$, where $\mathcal{N} \triangleq \{1, \cdots, N\}$. The robots are homogeneous with a circular or rectangular shape. The state of each robot at time $t \geqslant 0$ is defined by its position and orientation, i.e. $\mathbf{x}_n(t) \triangleq (x_n, y_n)$ is the center coordinate, ψ_n is the rotation, $\mathbf{v}_n(t) \triangleq (v_{nx}, v_{ny})$ is the linear velocity, and ω_n is the angular velocity, $\forall n \in \mathcal{N}$. Let $R_n(t) \subset \mathcal{W}$ also denote the area occupied by robot R_n at time $t \geqslant 0$. Moreover, each robot has a feedback controller for velocity tracking. Thus, the robots follow a simple first-order dynamics in the free space, i.e., $(\dot{\mathbf{x}}_n, \dot{\psi}_n) = (\mathbf{v}_n, \omega_n) \triangleq \mathbf{u}_n$. In addition, the workspace is cluttered with a set of static obstacles, denoted by $\mathcal{O} \subset \mathcal{W}$. Their shape and position are known a priori. Thus, the freespace is defined by $\widehat{\mathcal{W}} \triangleq \mathcal{W} \backslash \mathcal{O}$.

Lastly, there is a single target object $\Omega \subset \widehat{\mathcal{W}}$, of which the shape is an arbitrary polygon formed by $V \geqslant 3$ ordered vertices $p_1 p_2 \cdots p_V$. Similarly, its state at time $t \geqslant 0$ is defined by its position and orientation, i.e. $\mathbf{x}(t) \triangleq (x, y) \in \widehat{\mathcal{W}}$ is the center of mass and $\psi \in [-\pi, \pi]$ is the rotation, $\mathbf{v}(t) \triangleq (v_x, v_y)$ is the linear velocity and ω is the angular velocity, and its occupied area is denoted by $\Omega(t)$. For the ease of notation, let $\mathbf{s}(t) \triangleq (x, y, \psi) \in \mathcal{S}$ denote its full state and $\mathbf{p}(t) \triangleq (v_x, v_y, \omega)$ denote its generalized velocity. In addition, its intrinsics are known a priori, including its mass M, the moment of inertia $\mathcal{I}$ about the center of mass, the pressure distribution at bottom surface, the coefficient of lateral friction $\mu_c > 0$, and the coefficient of ground friction $\mu_s > 0$.

B. Interaction Modes and Coupled Dynamics

The robots can collaboratively move the target by making contacts with the target at different contact points, called *interaction modes*. More specifically, an interaction mode is defined by $\xi \triangleq \mathbf{c}_1 \mathbf{c}_2 \cdots \mathbf{c}_N$, where $\mathbf{c}_n \in \partial\Omega$ is the contact point on the boundary of the target for the n-th robot, $\forall n \in \mathcal{N}$. Since the set of contact points are *not* pre-defined, the complete set of all interaction modes is potentially infinite, denoted by Ξ. Moreover, if any robot R_n is not in contact with the target, it is called a transition mode, denoted by ξ_0. Thus, given a particular interaction mode, the robots can apply pushing forces in different directions with different magnitude. Denote by $\mathbf{f}_1 \mathbf{f}_2 \cdots \mathbf{f}_N$, where $\mathbf{f}_n \in \mathbb{R}^2$ is the contact force of robot R_n at contact point $\mathbf{c}_n$, $\forall n \in \mathcal{N}$. Furthermore, each force $\mathbf{f}_n$ can be decomposed in the directions of the normal vector $\mathbf{n}_n$ and the tangent vector $\boldsymbol{\tau}_n$ w.r.t. the target surface at the contact point $\mathbf{c}_n$, i.e., $\mathbf{f}_n \triangleq \mathbf{f}_n^{\mathrm{n}} + \mathbf{f}_n^{\mathrm{t}} \triangleq f_n^{\mathrm{n}} \mathbf{n}_n + f_n^{\mathrm{t}} \boldsymbol{\tau}_n$. Due to the Coulomb law of friction see [13], it holds that:

$$0 \leqslant f_n^{\mathrm{n}} \leqslant f_{n,\max};\; 0 \leqslant |f_n^{\mathrm{t}}| \leqslant \mu_c f_n^{\mathrm{n}}, \tag{1}$$

where $f_{n,\max} > 0$ is the maximum force each robot can apply; and μ_c is the coefficient of lateral friction defined earlier. Then, these decomposed forces can be re-arranged by:

$$\mathbf{F}_\xi \triangleq (\mathbf{F}_\xi^{\mathrm{n}},\, \mathbf{F}_\xi^{\mathrm{t}}) \triangleq (f_1^{\mathrm{n}}, \cdots, f_N^{\mathrm{n}}, f_1^{\mathrm{t}}, \cdots, f_N^{\mathrm{t}}) \in \mathbb{R}^{2N}, \tag{2}$$

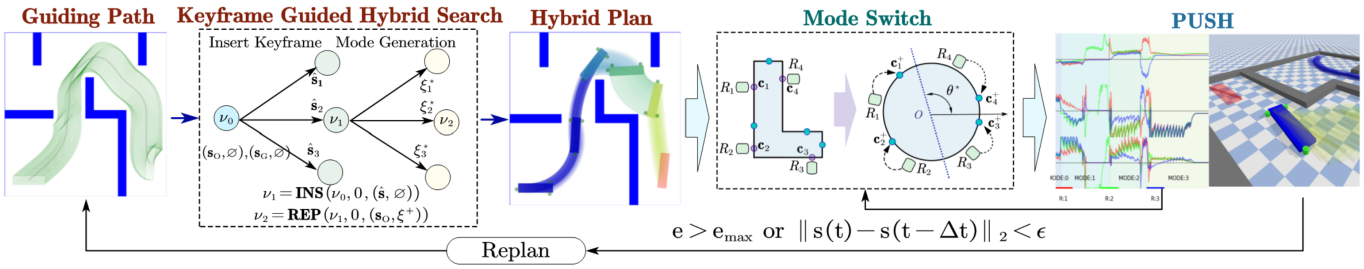

Fig. 2: Overall framework, including the hybrid plan generation in Sec. III-C and the online execution in Sec. III-D.

and further $\mathcal{F}_\xi \triangleq \{\mathbf{F}_\xi\}$ denotes the set of all forces within each mode $\xi \in \Xi$. Furthermore, the combined generalized force $\mathbf{q}_\xi \triangleq (f_x^\star, f_y^\star, m^\star)$ as also used in Lynch et al. [19] is given by:

$$(f_x^\star,\, f_y^\star) \triangleq \textstyle\sum_{n=1}^{N} \mathbf{f}_n;\; m^\star \triangleq \sum_{n=1}^{N} (\mathbf{c}_n - \mathbf{x}_n) \times \mathbf{f}_n, \tag{3}$$

where $\times$ is the cross product and $m^\star$ is the resulting torque from all robots. It can be written in matrix form $\mathbf{q}_\xi \triangleq \mathbf{J}\mathbf{F}_\xi$, where $\mathbf{J} \triangleq \nabla_{\mathbf{F}_\xi}\mathbf{q}_\xi$ is a $3 \times 2N$ Jacobian matrix. Similarly, let $\mathrm{Q}_\xi \triangleq \{\mathbf{q}_\xi\}$ denote the set of all *allowed* combined generalized forces within each mode $\xi \in \Xi$. Given $\mathbf{q}_\xi \in \mathrm{Q}_\xi$, the coupled dynamics of the robots and the target under mode ξ can be determined as follows:

$$\mathbf{M}\dot{\mathbf{p}} = \mathbf{q}_\xi + \mathbf{q}_\mu \triangleq \zeta(\mathbf{p}, \mathbf{u}_\mathcal{N}) + \eta(\mathbf{p}); \tag{4a}$$

$$\dot{\mathbf{x}}_n = \mathbf{v}_n = \mathbf{v} + \boldsymbol{\omega} \times (\mathbf{c_n} - \mathbf{x}), \quad \forall n \in \mathcal{N}; \tag{4b}$$

where $\mathbf{M} \triangleq \mathrm{diag}(M, M, \mathcal{I})$; $\mathbf{q}_\xi \triangleq \zeta(\mathbf{p}, \mathbf{u}_\mathcal{N})$ is the generalized pushing force in (3) given the combined control inputs $\mathbf{u}_\mathcal{N} \triangleq \mathbf{u}_1 \cdots \mathbf{u}_N$ and the velocity of the target $\mathbf{p}$; and $\mathbf{q}_\mu \triangleq \eta(\mathbf{p}) = (f_x, f_y, m)$ is the ground friction, determined by the velocity $\mathbf{p}$ and intrinsics of the target. Namely, (4a) describes how the target moves under the external forces $\mathbf{q}_\xi$ and $\mathbf{q}_\mu$ within mode ξ; and (4b) describes how the robot velocity is determined by the target velocity under the non-slipping constraints.

Remark 1. Due to the complex nature of the multi-body physics under contact, both the pushing forces $\mathbf{q}_\xi$ and friction $\mathbf{q}_\mu$ in (4a) lack an analytical form, yielding difficulties in state prediction and control. However, it is certain that $\mathbf{q}_\xi$ must be *allowed* in mode ξ, i.e., $\mathbf{q}_\xi \in \mathrm{Q}_\xi$. ■

C. Problem Statement

The planning objective is to compute a *hybrid plan*, including a target trajectory $\mathbf{s}(t)$, a sequence of interaction modes $\xi(t)$ and the associated pushing forces $\mathbf{q}_\xi(t)$, such that the target is moved from any initial state $\mathbf{s}_0$ to a given goal state $\mathbf{s}_G$. Meanwhile, the robots and the target should avoid collision with all obstacles at all time. More precisely, it is formulated as a hybrid optimization problem:

$$\begin{aligned} \min_{\{\xi(t), \mathbf{q}_\xi(t), \mathbf{s}(t)\}, T} \; & \left\{ T + \alpha \sum_{t=0}^{T} J\big(\xi(t), \mathbf{q}_\xi(t), \mathbf{s}(t)\big) \right\} \\ \textbf{s.t.} \quad & \mathbf{s}(0) = \mathbf{s}_0,\, \mathbf{s}(T) = \mathbf{s}_G; \\ & \Omega(t) \subset \widehat{\mathcal{W}},\, R_n(t) \subset \widehat{\mathcal{W}},\, \forall n \in \mathcal{N},\, \forall t \in \mathcal{T}; \\ & \xi(t) \in \Xi,\, \mathbf{q}_{\xi(t)} \in \mathrm{Q}_{\xi(t)},\, \forall t \in \mathcal{T}; \end{aligned} \tag{5}$$

where $T > 0$ is the task duration to be optimized; $\mathcal{T} \triangleq \{0, 1, \cdots, T\}$; $J : \Xi \times \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}_+$ is a given function to measure the feasibility, stability and control cost of choosing a certain mode and the applied forces given the desired target trajectory; and $\alpha > 0$ is the weight parameter to balance the task duration and the control performance.

Remark 2. The design of function $J(\cdot)$ is non-trivial and technically involved, thus explained further in the sequel. ■

Remark 3. The hybrid search space of (5) above is over the timed sequence of pushing modes, forces and the target trajectory, which has dimension $(N + 6) \cdot T_{\max}$ with $T_{\max}$ being the maximum duration. ■

Remark 4. Existing work in Wang and De Silva [28], Goyal et al. [8], Chen et al. [4], Tang et al. [22] considers rectangular objects in freespace or assumes a set of a hand-picked modes and contact points. In contrast, general polytopic objects in cluttered workspace are adopted here by allowing multiple simultaneous contacts without any pre-defined modes. ■

III. Proposed Solution

As illustrated in Fig. 2, the proposed solution tackles the above collaborative pushing problem with an efficient keyframe-guided hybrid search algorithm over the timed sequence of pushing modes, forces and target trajectories. The resulting hybrid plan is then executed by a mode switching strategy and a NMPC controller. A re-planning module governs the execution performance and triggers the adaptation of high-level hybrid plan as needed.

A. Mode Generation in Freespace

This section first presents how to evaluate the feasibility of a given mode to push the target at a desired velocity in the quasi-static condition, which is then extended to an arc transition in the freespace. Based on this measure, a sparse optimization algorithm is proposed to generate a set of effective modes given the desired target motion.

1) Quasi-static Analyses: As described in Sec. II-B, it is difficult to derive an analytical form of the coupled dynamics in (4). As also adopted in Lynch et al. [19], the quasi-static analyses assume that the motion of the target is sufficiently slow, such that its acceleration is approximately zero and the inertia forces can be neglected. Consequently, the target can be constantly treated as in static equilibrium, i.e., $\mathbf{q}_\mu + \mathbf{q}_\xi = \mathbf{0}$.

Under quasi-static assumption, the friction $\mathbf{q}_\mu = (f_x, f_y, m) \in \mathbb{R}^3$ is constrained on a limit surface which is approximated by the following ellipsoid:

$$(f_x/f_{\max})^2 + (f_y/f_{\max})^2 + (m/m_{\max})^2 = 1, \tag{6}$$

where $f_{\max}$ and $m_{\max}$ are the maximum ground friction and moment, defined at the center of the target. Then, the relation between $\mathbf{q}_\mu$ and $\mathbf{p}$ is given by $(f_x, f_y, m) = -\kappa(v_x, v_y, c^2\omega)$, where $c \triangleq m_{\max}/f_{\max}$ and $\kappa > 0$. Consequently, given the desired velocity $\mathbf{p}^\star$ of the target, the generalized friction force $\mathbf{q}_\mu$ is computed as:

$$\mathbf{q}_\mu \triangleq \widetilde{\eta}(\mathbf{p}^\star) = -\|\mathbf{D}_1\mathbf{D}_2\mathbf{p}\|_2^{-1}\mathbf{D}_2\mathbf{p}^\star, \tag{7}$$

where $\mathbf{D}_1 \triangleq \operatorname{diag}(f_{\max}^{-1}, f_{\max}^{-1}, m_{\max}^{-1})$, $\mathbf{D}_2 \triangleq \operatorname{diag}(1, 1, c^2)$, and $\widetilde{\eta}(\cdot)$ is the quasi-static approximation of $\eta(\cdot)$ in (4a). Thus, assuming that no slipping happens during pushing, the coupled dynamics in (4) under control inputs $\mathbf{u}_\mathcal{N}$ and interaction mode ξ is approximated by:

$$\mathbf{q}_\xi = -\mathbf{q}_\mu = -\widetilde{\eta}(\mathbf{p}^\star) \in \mathrm{Q}_\xi; \tag{8a}$$

$$\mathbf{v} + \boldsymbol{\omega} \times (\mathbf{c}_n - \mathbf{x}) = \mathbf{u}_n,\ \forall n \in \mathcal{N}; \tag{8b}$$

where (8a) is a condition to check if a velocity $\mathbf{p}$ is *allowed* within mode ξ. This quasi-static equilibrium can be maintained if there exists a velocity $\widetilde{\mathbf{p}}^\star$ that satisfies both (8a) and (8b), in which case the target moves at velocity $\widetilde{\mathbf{p}}^\star$ with each robot pushing at velocity $\mathbf{u}_n$. If not, the quasi-static condition is violated and the desired target motion can not be maintained.

2) Multi-directional Feasibility Estimation: For further clarification, let $\mathbb{S}$ denote the ground coordinate system and $\mathbb{B}$ denote the body coordinate system fixed at the center of the target $\mathbf{x}(t)$. The rotation matrix from $\mathbb{B}$ to $\mathbb{S}$ is denoted by $\mathbf{R}_{\mathrm{B}\to\mathrm{S}}$, which is time-varying as the target moves. Then, the generalized velocity of the target in $\mathbb{B}$ is defined as $\mathbf{p}^{\mathrm{B}} \triangleq (v_x^{\mathrm{B}}, v_y^{\mathrm{B}}, \omega^{\mathrm{B}}) = \mathbf{R}_{\mathrm{B}\to\mathrm{S}}^{-1}\mathbf{p}$. Moreover, the coordinate $\mathbb{B}$ provides a *rotation-invariant* description of the mode ξ. Denote by $\mathrm{Q}_\xi^{\mathrm{B}} \triangleq \{\mathbf{R}_{\mathrm{B}\to\mathrm{S}}^{-1}\mathbf{q}_\xi, \forall \mathbf{q}_\xi \in \mathrm{Q}_\xi\}$ the set of allowed forces Q_ξ transformed to $\mathbb{B}$. It can be seen that $\mathrm{Q}_\xi^{\mathrm{B}}$ is a fixed set for a given mode ξ and independent of the target rotation ψ. Similarly, denote by $\mathcal{P}_\xi^{\mathrm{B}} \triangleq \{\mathbf{p}^{\mathrm{B}} | -\eta(\mathbf{p}^{\mathrm{B}}) \in \mathrm{Q}_\xi^{\mathrm{B}}\}$ the set of allowed velocities in mode ξ in $\mathbb{B}$, which is also fixed.

The feasibility of pushing the target at a generalized velocity $\mathbf{p}^{\mathrm{B}}$ in a specific mode ξ can be evaluated by solving the following constrained optimization problem:

$$\mathrm{J}_{\mathrm{F}}(\xi, \mathbf{p}^{\mathrm{B}}) \triangleq \min_{\mathbf{q}_\xi^{\mathrm{B}} \in \mathrm{Q}_\xi^{\mathrm{B}}} \left\|\mathbf{q}_\xi^{\mathrm{B}} + \eta(\mathbf{p}^{\mathrm{B}})\right\|_1, \tag{9}$$

where $\mathbf{q}_\xi^{\mathrm{B}}$ is the combined generalized force of all robots, and $\eta(\mathbf{p}^{\mathrm{B}})$ is the generalized friction of the target. Note that $\mathbf{q}_\xi^{\mathrm{B}} \in \mathrm{Q}_\xi^{\mathrm{B}}$ is a linear constraint, and the $\|\cdot\|_1$ norm can be transferred to a linear objective by adding auxiliary variables. Thus, (9) can be readily solved by LP solvers, e.g., `CVXOPT` from Diamond and Boyd [6]. If $\mathrm{J}_{\mathrm{F}} = 0$, it means that the generalized force $\mathbf{q}_\xi^{\mathrm{B}} = -\eta(\mathbf{p}^{\mathrm{B}})$ is allowed in mode ξ, i.e., the target can maintain the quasi-static condition at $\mathbf{p}^{\mathrm{B}}$, yielding a necessary condition for feasibility.

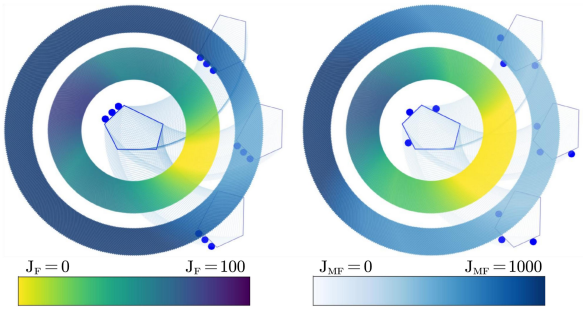

Fig. 3: Comparison of J_{F} in (9) and J_{MF} in (10) for different directions between different modes (robots as blue dots and object as a white polygon).

However, due to unmodeled noises and uncertainties during the contact-rich pushing process, the target motion may deviates from the expected trajectory. Therefore, to facilitate online adaptation during execution, it is essential to estimate the feasibility *in multiple directions*. Denote by $\mathcal{D}$ the set of directions, which contains the desired velocity $\mathbf{p}^{\mathrm{B}}$ as the main direction, and several orthogonal auxiliary directions. Since $\dim(\mathbf{p}) = 3$, five auxiliary directions are chosen to span the velocity space with only positive weights, i.e., $\mathbf{p}_{[1]}^{\mathrm{B}} \triangleq \mathbf{p}^{\mathrm{B}}, \mathbf{p}_{[2]}^{\mathrm{B}} \triangleq \mathbf{e}_3 \times \mathbf{p}_{[1]}^{\mathrm{B}}, \mathbf{p}_{[3]}^{\mathrm{B}} \triangleq \mathbf{p}_{[1]}^{\mathrm{B}} \times \mathbf{p}_{[2]}^{\mathrm{B}}, \mathbf{p}_{[4]}^{\mathrm{B}} \triangleq -\mathbf{p}_{[1]}^{\mathrm{B}}, \mathbf{p}_{[5]}^{\mathrm{B}} \triangleq -\mathbf{p}_{[2]}^{\mathrm{B}}, \mathbf{p}_{[6]}^{\mathrm{B}} \triangleq -\mathbf{p}_{[3]}^{\mathrm{B}}$, where $\mathbf{e}_3 \triangleq (0, 0, 1)$ and $\mathcal{D} \triangleq \{1, \cdots, 6\}$. Consequently, the multi-directional feasibility estimation is the weighted sum of feasibility in each direction, i.e.,

$$\mathrm{J}_{\mathrm{MF}}(\xi, \mathbf{p}^{\mathrm{B}}) \triangleq \sum_{d\in\mathcal{D}} w_d \mathrm{J}_{\mathrm{F}}(\xi, \mathbf{p}_{[d]}^{\mathrm{B}}), \tag{10}$$

where the weights $w_d > 0$, $\forall d \in \mathcal{D}$ with w_0 being the maximum. As illustrated in Fig. 3, these two modes (all from one side or each at different sides) have an equally-small J_{F} in certain directions. But the second mode has a significantly lower J_{MF} in a wide range of directions *around* the desired direction, of which the balance between different directions is tunable by the weights $\{w_d\}$.

Remark 5. Note that given enough number of robots, a mode similar to the caging strategy in Lynch et al. [19] is preferred by (10), i.e., surrounding the target from multiple sides instead of all pushing from the same side. This allows the robots to potentially rectify the target motion online whenever it deviates from the desired velocity during execution. ■

Remark 6. The above analyses still apply if the robots can *pull* the target. The only difference is that the contact force $\mathbf{f}_n$ is no longer constrained by the friction cone in (13b), which often leads to a smaller J_{MF} and higher feasibility. ■

3) Loss Estimation for Arc Transition: Given any allowed target velocity $\mathbf{p}^{\mathrm{B}} = (\mathbf{v}^{\mathrm{B}}, \omega^{\mathrm{B}}) \in \mathcal{P}_\xi^{\mathrm{B}}$ in mode ξ, the resulting motion of the target is analyzed as follows. Assuming that the robots push the target with velocity $\mathbf{p}(t) = \mathbf{R}_{\mathrm{B}\to\mathrm{S}}(t)\mathbf{p}^{\mathrm{B}}$ for time duration $\bar{t} > 0$, the quasi-static condition is ensured as $\mathbf{p}^{\mathrm{B}} = \mathbf{R}_{\mathrm{B}\to\mathrm{S}}^{-1}(t)\mathbf{p}(t) \in \mathcal{P}_\xi^{\mathrm{B}}$ holds. Consequently, the change

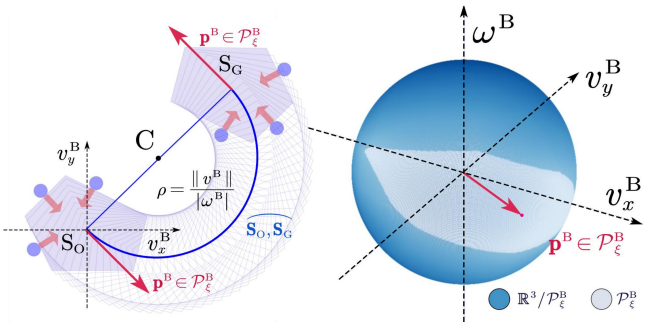

Fig. 4: Arc transitions (**Left**) and the set of allowed velocities $\mathcal{P}^{\mathrm{B}}_{\xi}$ (**Right**) visualized on a unit sphere.

of target state within $\mathbb{S}$ during period $[0, \bar{t}]$ is given by:

$$\begin{aligned}\Delta\mathbf{s} &\triangleq \left(\int_{t=0}^{\bar{t}} \mathbf{R}_{\mathrm{B}\to\mathrm{S}}(t)dt\right)\mathbf{p}^{\mathrm{B}} \\ &= \begin{bmatrix} \lim\limits_{\Delta\psi\to\Delta\Psi} \mathbf{Rot}\left(\Delta\psi + \psi_0 - \frac{\pi}{2}\right)|_0^{\Delta\psi}\frac{1}{\Delta\psi} & 0 \\ 0 & 1 \end{bmatrix}\bar{t}\ \mathbf{p}^{\mathrm{B}} \\ &= \mathbf{\Phi}(\psi_0, \Delta\Psi)\ \bar{t}\ \mathbf{p}^{\mathrm{B}},\end{aligned} \tag{11}$$

where $\mathbf{Rot} : (-\pi, \pi] \to \mathbb{R}^{2\times 2}$ is the standard rotation matrix of a given angle; $\Delta\Psi = \omega^{\mathrm{B}}\bar{t}$ is the change of target orientation. Note that the limit operation above is employed due to the singularity at $\omega^{\mathrm{B}} = 0$, in which case the limit is given by $\mathbf{Rot}(\psi_0)$, yielding a pure translational motion $\Delta\mathbf{s} = \mathbf{R}_{\mathrm{B}\to\mathrm{S}}(0)\mathbf{p}^{\mathrm{B}}\bar{t}$. Thus, the final state of the target is given by $\mathbf{s}_{\bar{t}} \triangleq \mathbf{s}_0 + \Delta\mathbf{s}$ and its trajectory by $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$.

Lemma 1. *Given a fixed velocity $\mathbf{p}^{\mathrm{B}}$, the resulting trajectory $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$ is an arc with radius $\rho = \|\mathbf{v}^{\mathrm{B}}\|/|\omega^{\mathrm{B}}|$ and angle $\Delta\psi$; or a straight line with length $\|\mathbf{v}^{\mathrm{B}}\|\bar{t}$.*

Proof: (Sketch) If $\Delta\Psi \neq 0$, consider the point $\mathbf{x}_c \triangleq \mathbf{x}_0 - \mathbf{Rot}(\psi_0 - \pi/2)\mathbf{v}^{\mathrm{B}}/\omega^{\mathrm{B}}$. For any $t \in [0, \bar{t}]$, the target position by (11) is given by $\mathbf{x}_t \triangleq \mathbf{x}_0 + \Delta\mathbf{x}$. It can be verified that $\|\mathbf{x}_t - \mathbf{x}_c\|_2$ is a constant and equals to $\|\mathbf{v}^{\mathrm{B}}\|/|\omega^{\mathrm{B}}|$. The rest of the proof follows similar arguments as for (11). ■

Lemma 2. *Any given pair of initial state $\mathbf{s}_0$ and goal state $\mathbf{s}_{\mathrm{G}}$ can be connected by a unique arc trajectory $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$ such that $\omega^{\mathrm{B}}\,\bar{t} \in [-\pi, \pi)$.*

Proof: (Sketch) Let $\Delta\Psi$ in (11) be $\psi_{\mathrm{G}} - \psi_0$ for the arc transition. Since the matrix $\mathbf{\Phi}(\psi_0, \Delta\Psi)$ is invertible, it implies that $\mathbf{p}^{\mathrm{B}}\bar{t} = \mathbf{\Phi}^{-1}(\psi_0, \Delta\Psi)(\mathbf{s}_{\mathrm{G}} - \mathbf{s}_0)$ is uniquely determined. Moreover, scaling the velocity $\mathbf{p}^{\mathrm{B}}$ by a factor $k \in \mathbb{R}^+$ leads to the same arc trajectory with duration $\bar{t}/k$. Thus, the trajectory $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$ is uniquely determined. ■

With Lemma 2, we denote by $\widehat{\mathbf{s}_0\mathbf{s}_{\mathrm{G}}} \triangleq \mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$ the unique arc transition from $\mathbf{s}_0$ to $\mathbf{s}_{\mathrm{G}}$, of which the arc length is given by $\|\widehat{\mathbf{s}_0\mathbf{s}_{\mathrm{G}}}\| \triangleq \|\mathbf{p}^{\mathrm{B}}\bar{t}\|_2$. Consequently, a loss estimation is proposed for choosing a mode ξ under an arc transition $\widehat{\mathbf{s}_0\mathbf{s}_{\mathrm{G}}}$:

$$\mathrm{J}_{\mathrm{MF}}(\xi, \widehat{\mathbf{s}_0\mathbf{s}_{\mathrm{G}}}) = \mathrm{J}_{\mathrm{MF}}(\xi, \mathbf{p}^{\mathrm{B}})\,\|\mathbf{p}^{\mathrm{B}}\bar{t}\|_2, \tag{12}$$

where $\mathrm{J}_{\mathrm{MF}}(\cdot)$ is the multi-directional feasibility estimation in (10). In other words, the loss estimation for choosing a mode under an arc transition is estimated by the feasibility of the associated velocity but weighted by the arc length.

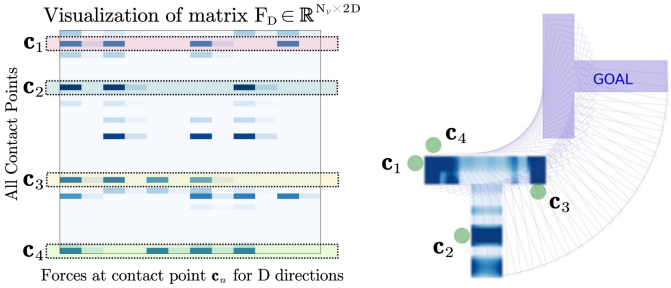

Fig. 5: Illustration of the mode generation process via sparse optimization in (13). **Left**: the optimized matrix $\mathbf{F}^{\star}_{\mathbf{D}}$ of dimention $N_\mathcal{V} \times 2D$, where the magnitude of elements is represented by the color intensity; **Right**: the best mode selected for the given arc transition within $\mathbf{\Xi}(\widehat{\mathbf{s}_0\mathbf{s}_{\bar{t}}})$.

4) Mode Generation for Arc Transition: With the proposed loss function in (12), the optimal mode $\xi^\star$ for a desired arc transition $\widehat{\mathbf{s}_0\mathbf{s}_{\mathrm{G}}}$ can be determined. However, as described earlier, the set of all possible modes $\mathbf{\Xi}$ in this work is infinite, of which the parameter space grows exponentially with the number of robots and the boundary length of the target. To tackle this issue, a fast and scalable method to generate suitable modes for a given arc transition is proposed, called *mode generation via sparse optimization* (MG-SO). To begin with, each side (p_v, p_{v+1}) of the target is divided into $N_v > 0$ segments, $\forall v \in \{1, \cdots, V\}$. The center of each segment is denoted by $p_{v,i}$, $\forall i \in \{1, \cdots, N_v\}$. Then, consider a virtual mode $\xi^\star$ that employs $\{p_{v,i}, \forall i, \forall v\}$ as the contact points, i.e., $\xi^\star \triangleq p_{1,1}\cdots p_{1,N_1}\cdots p_{V,1}\cdots p_{V,N_V}$. Note that the total number of contact points is $N_\mathcal{V} = \sum_{v=1}^{V} N_v$, which is much larger than the number of robots N. The associated force $\mathbf{F}_{\xi^\star} \triangleq (\mathbf{F}_{\mathrm{n}}, \mathbf{F}_{\mathrm{t}})$ follows the definition in (2), among which the forces along the multiple directions $\mathbf{p}^{\mathrm{B}}_{[d]}$ in (10) are given by $\mathbf{F}^{[d]}_{\xi^\star} \triangleq (\mathbf{F}^{[d]}_{\mathrm{n}}, \mathbf{F}^{[d]}_{\mathrm{t}})$, $\forall d \in \mathcal{D}$. Subsequently, the following sparse optimization problem is formulated:

$$\min_{\mathbf{F_D}} \sum_{n\in\mathcal{N}_\mathcal{V}} (\|\mathbf{F}^n_{\mathbf{D}}\|_\infty + \|\mathbf{F}^n_{\mathbf{D}}\|_1) + \sum_{d\in\mathcal{D}} \|\mathbf{q}^{\mathrm{B}}_{[d]} + \eta(\mathbf{p}^{\mathrm{B}}_{[d]})\|_1 \tag{13a}$$

$$\text{s.t.} \quad \mathbf{q}^{\mathrm{B}}_{[d]} = \mathbf{J}\mathbf{F}^{[d]}_{\xi^\star},\ \mathbf{F}^{[d]}_{\xi^\star} \in \mathcal{F}_{\xi^\star},\ \forall d \in \mathcal{D}; \tag{13b}$$

where the decision variables $\mathbf{F_D} \triangleq \mathbf{F}^{[1]}_{\mathrm{n}}\mathbf{F}^{[1]}_{\mathrm{t}}\cdots\mathbf{F}^{[D]}_{\mathrm{n}}\mathbf{F}^{[D]}_{\mathrm{t}} \in \mathbb{R}^{N_\mathcal{V}\times 2D}$; $\mathbf{F}^n_{\mathbf{D}} \in \mathbb{R}^{2D}$ is the n-th row of $\mathbf{F_D}$; and the first term in (13a) contains both the $\mathcal{L}_1$ norm and $\mathcal{L}_\infty$ norm of $\mathbf{F}^n_{\mathbf{D}}$, while the second term is the feasibility estimation in (9) for each direction; the constraint of friction cone in (13b) is defined in (1). Note the first term in (13a) is designed to improve the sparsity of matrix $\mathbf{F_D}$, i.e., $\sum_{n\in\mathcal{N}_\mathcal{V}} \|\mathbf{F}^n_{\mathbf{D}}\|_1$ penalizes the number of non-zero elements in $\mathbf{F_D}$, while $\sum_{n\in\mathcal{N}_\mathcal{V}} \|\mathbf{F}^n_{\mathbf{D}}\|_\infty$ penalizes the number of non-zero rows in $\mathbf{F_D}$, i.e., the *row sparsity* in Leon et al. [18]. This is crucial as a row of zeros indicates that forces at the corresponding contact point are not required. In other words, only key contact points are associated with the non-zero rows of $\mathbf{F_D}$.

The above optimization problem is again a linear program (LP), which can be solved efficiently with a LP solver. Denote

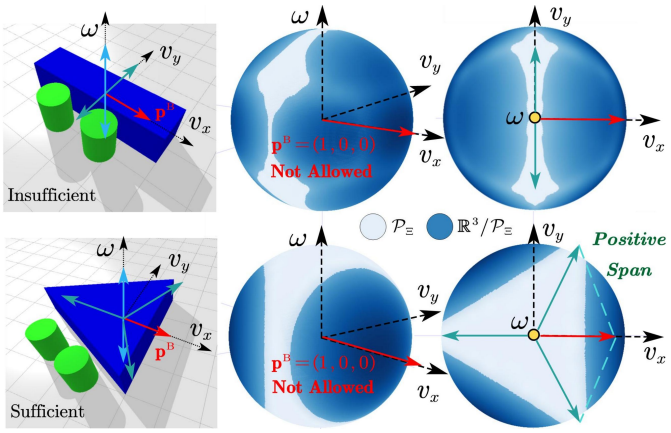

Fig. 6: Given all modes Ξ for two robots and different objects (**Left**), the set of allowed velocities $\mathcal{P}_\Xi$ is shown (in white) on a unit sphere of $v_x - v_y - \omega$ velocities (**Middle**) with the top-down projection (**Right**) from the angular velocity ω.

by $\mathbf{F}_\mathbf{D}^\star$ the optimal solution, which might contain more than N non-zero rows. Thus, a set of modes for this arc transition, denoted by $\mathbf{\Xi}(\widehat{\mathbf{s}_0\mathbf{s}_\mathrm{G}})$, can be generated as follows. All rows of $\mathbf{F}_\mathbf{D}^\star$ are sorted in descending order by the value of $\|\mathbf{F}_\mathbf{D}^n\|_\infty + \|\mathbf{F}_\mathbf{D}^n\|_1$. Then, the first mode is given by the collecting the first N rows, and the subsequent mode is generated by combing the first $N-1$ rows with a random row. This process is repeated until a certain size of $\mathbf{\Xi}(\widehat{\mathbf{s}_0\mathbf{s}_\mathrm{G}})$ is reached. An example of the results is shown in Fig. 5. It can be seen that the sparsity optimization is effective as most elements of the matrix $\mathbf{F}_\mathbf{D}^\star$ are zero, while the non-zero elements are concentrated in few rows, as visualized by the color intensity.

Remark 7. Due to the polynomial complexity of solving linear programs, the above mode generation method is scalable to arbitrary number of robots and any polytopic target shape. For non-polytopic shapes, its exterior contour should be discretized first to obtain the set of potential contact points, which are then filtered by whether there exists enough room for a robot. Given these contact points, the same optimization in (13) can be adopted to generate modes for an arc transition. More case studies of are provided in Sec. IV. ■

B. Sufficient Modes for General Trajectory

The previous section provides an efficient method to generate a set of modes for an arc transition in the freespace, i.e., via a constant velocity $\mathbf{p}^\mathrm{B}$ in $\mathbb{B}$. However, this velocity may not be allowed within Ξ, i.e., $\mathbf{p}^\mathrm{B} \notin \bigcup_{\xi\in\Xi} \mathcal{P}_\xi^\mathrm{B}$, or this arc might intersect with obstacles. Consequently, given a general collision-free trajectory $\overline{\mathbf{S}}$, the main idea is to approximate it by a sequence of *allowed* arc transitions, i.e.,

$$\widetilde{\mathbf{S}} \triangleq \bigcup_{\ell\in\mathcal{L}} \widehat{\mathbf{s}_{\ell-1}\mathbf{s}_\ell}, \tag{14}$$

where $\mathcal{L} \triangleq \{1, \cdots, L\}$ and $L > 0$ is the number of arcs. The approximation error is measured by the Hausdorff distance $\mathcal{H}(\overline{\mathbf{S}}, \widetilde{\mathbf{S}})$ from Rockafellar and Wets [20].

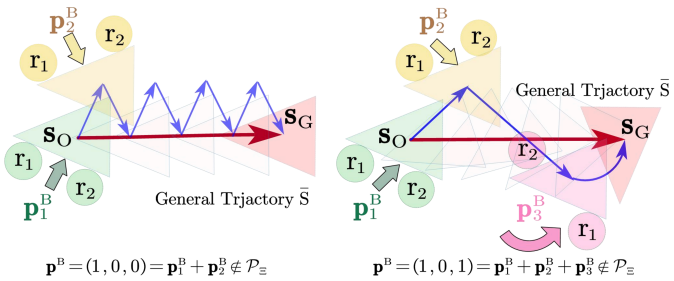

Fig. 7: Trajectory approximation via a set of sufficient modes.

To begin with, when any generalized velocity $\mathbf{p}$ is allowed in at least one mode within Ξ, i.e., $\mathbf{p} \in \bigcup_{\xi\in\Xi} \mathcal{P}_\xi$ holds for all $\mathbf{p} \in \mathbb{R}^3$, the above approximation can be achieved by dividing $\overline{\mathbf{S}}$ into sufficiently small segments, each of which can be approximated by an arc. However, this condition is not easily met, e.g., when the number of robots is small, or certain points on the edge can not be contact points, or the combined forces are not sufficient in certain directions. As shown in the Fig. 6, when the maximum pushing force of each robot is only half of the maximum friction force, the above condition can not be satisfied. More importantly, verifying this condition requires traversing all velocities, and calling MG-SO for each direction to verify whether it is allowed in at least one mode, which is computationally expensive. Thus, a weaker and easier-to-check notion of sufficiency is proposed as follows:

Definition 1. The set of all possible modes Ξ is called *sufficient* if two conditions hold: (i) any generalized velocity $\mathbf{p}$ can be positively decomposed into several key velocities, i.e., $\mathbf{p} = \sum_{i\in\mathcal{I}} \lambda_i \mathbf{p}_i$, where $\lambda_i \geqslant 0$; (ii) $\mathbf{p}_i$ is allowed in at least one mode $\xi \in \Xi$, $\forall i \in \mathcal{I}$, i.e., $\mathbf{p}_i \in \mathcal{P}_\Xi \triangleq \bigcup_{\xi\in\Xi} \mathcal{P}_\xi$. ■

In other words, only the key velocities need be verified as allowed in at least one mode, rather than all possible velocities. For instance, the six directions $\{\mathbf{p}_{[d]}^\mathrm{B}\}$ in (10) can be chosen as the key velocities, for which the above conditions holds by definition. Furthermore, it is shown below that a set of sufficient modes can generate arc transitions to approximate any general trajectory of the target with arbitrary accuracy. As shown in Fig. 6, the allowed velocities $\mathcal{P}_\Xi$ for all possible modes Ξ of two robots and a triangular object has a triangular distribution, thus easier to span all directions. In contrast for rectangular objects, the allowed velocities are concentrated on $v_y - \omega$ plane, i.e., difficult to span in the v_x direction.

Theorem 1. *If the set of all possible modes Ξ is sufficient, then for any collision-free trajectory $\overline{\mathbf{S}}$ and any error bound $e > 0$, there exists a sequence of allowed arc transitions $\widetilde{\mathbf{S}}$ by* (14) *such that $\mathcal{H}(\overline{\mathbf{S}}, \widetilde{\mathbf{S}}) \leqslant e$.*

Proof: First, $\overline{\mathbf{S}}$ can be divided into L segments evenly by length, such that the ℓ-th segment $(\mathbf{s}_{\ell-1}, \mathbf{s}_\ell)$ is denoted as $\overline{\mathbf{S}}_\ell$. Assuming that $\overline{\mathbf{S}}_{\ell-1}$ is already approximated by an arc that ends at $\mathbf{s}_{\ell-1}^+$. Then, the desired state change $\Delta\mathbf{s}_\ell = \mathbf{s}_\ell - \mathbf{s}_{\ell-1}^+$ can be approximated by the arc associated with velocity $\mathbf{p}_\ell^\mathrm{B}$ and duration T_ℓ, i.e., $\Delta\mathbf{s}_\ell = \mathbf{\Phi}(\psi_{\ell-1}, \omega_\ell^\mathrm{B} T_\ell)\, \mathbf{p}_\ell^\mathrm{B}\, T_\ell$. Second,

since Ξ is sufficient, it holds that $\mathbf{p}_\ell^{\text{B}} = \sum_{i\in\mathcal{I}} \lambda_i \mathbf{p}_i$, where $\lambda_i \geqslant 0$ and $\mathbf{p}_i \in \mathcal{P}_\Xi$. Since $\dim(\mathcal{P}) = 3$, there exists $\lambda_j^\star \geqslant 0$ and $\mathbf{p}_1^\star, \mathbf{p}_2^\star, \mathbf{p}_3^\star \in \mathcal{P}_\Xi$ such that $\mathbf{p}_\ell^{\text{B}} = \sum_{j=1}^3 \lambda_j^\star \mathbf{p}_j^\star$. Consider the sequence of arcs $(\mathbf{p}_1^\star, t_1)(\mathbf{p}_2^\star, t_2)(\mathbf{p}_3^\star, t_3)$ that starts from $\mathbf{s}_{\ell-1}^+$, where $t_j = \lambda_j^\star t$ is the duration of velocity $\mathbf{p}_j^\star$. The resulting change of state and its derivative at $t = 0$ is given by:

$$\Delta\mathbf{s}_3^{\text{seq}} \triangleq \sum_{j=1}^{3} \mathbf{\Phi}(\psi_{j-1}, \omega_j t_j)\mathbf{p}_j^\star\, t_j;\ \ \frac{\mathrm{d}\mathbf{s}_3^{\text{seq}}}{\mathrm{d}t} = \sum_{j=1}^{3} \lambda_j \mathbf{p}_j^\star, \quad (15)$$

where $\psi_j \triangleq \psi_0 + \sum_{k=1}^{j} \omega_k t_k$. Denote by $\widetilde{\mathbf{S}}_\ell$ the trajectory from this 3-step motion, for the ℓ-th segment $\overline{\mathbf{S}}_\ell$. Their Hausdorff distance is given by: $h_\ell \leqslant (\max_{j,t} \{|\mathbf{p}_j^{\text{B}} - \frac{\mathrm{d}\overline{\mathbf{S}}(t)}{\mathrm{d}t}|\})T_\ell + \epsilon_{\ell-1} \leqslant \mathcal{O}(\tau) + \epsilon_{\ell-1}$, where $\tau \triangleq \frac{1}{L}$ and $\epsilon_{\ell-1} \triangleq \|\mathbf{s}_{\ell-1}^+ - \mathbf{s}_{\ell-1}\|_2$. The difference in the final state between $\overline{\mathbf{S}}_\ell$ and $\widetilde{\mathbf{S}}_\ell$ is given by $\epsilon_\ell = \Delta\mathbf{s}_\ell - \Delta\mathbf{s}_3^{\text{seq}} = \mathbf{p}^{\text{B}} T_\ell - \mathbf{p}^{\text{B}} T_\ell + \mathcal{O}(T_\ell^2) = \mathcal{O}(\tau^2)$. Moreover, via the same approximation for $\overline{\mathbf{S}}_{\ell-1}$, the error $\epsilon_{\ell-1}$ is also bounded by $\mathcal{O}(\tau^2)$, meaning that the distance $h_\ell \leqslant \mathcal{O}(\tau) + \mathcal{O}(\tau^2) = \mathcal{O}(\tau)$. Thus, the Hausdorff distance between $\overline{\mathbf{S}}$ and $\widetilde{\mathbf{S}}$ is bounded by $\mathcal{H}(\overline{\mathbf{S}}, \bigcup_{l\in\mathcal{L}} \widetilde{\mathbf{s}_{\ell-1}\mathbf{s}_\ell}) \leqslant \max_{\ell\in\mathcal{L}}\{h_\ell\} \leqslant \mathcal{O}(\tau)$. Thus, for any error bound $e > 0$, there exits a sufficiently large L such that $\mathcal{H}(\overline{\mathbf{S}}, \widetilde{\mathbf{S}}) < e$ holds. ■

Remark 8. As shown in Fig. 7, the proof above also provides a general method for approximating a collision-free trajectory by a sequence of allowed arcs. However, due to the uniform discretization, it requires a large number of arcs for a high accuracy, yielding over-frequent mode switches. Thus, a more efficient hybrid search algorithm is proposed in the sequel. ■

Remark 9. Note that the sufficiency in Def. 1 is defined for *all* possible modes Ξ, which is determined by the multi-pusher system and different from the generated modes $\Xi(\widetilde{\mathbf{s}_0\mathbf{s}_{\text{G}}})$ for an arc transition via (13). However, it can be verified whether these modes are sufficient for a given trajectory. ■

C. Hierarchical Hybrid Search

Given the above mode generation method, the section presents the hierarchical hybrid search algorithm to generate the hybrid plan as the timed sequence of modes and target states. In particular, it consists of two main steps: (i) a collision-free guiding path is generated for the target given the initial and goal states; (ii) a hybrid search algorithm that iteratively decomposes the guiding path into arc segments and selects the optimal modes, to minimize a balanced cost of the resulting hybrid plan.

1) Collision-free Guiding Path: A collision-free guiding path for the target from $\mathbf{s}_{\text{O}}$ to $\mathbf{s}_{\text{G}}$ is determined first as a sequence of states within its statespace $\mathcal{S}$, such that its polygon boundary does not intersect with any obstacle at all time. The $\mathbb{A}^\star$ algorithm as in LaValle [16] is adopted with the distance-to-goal being the search heuristic. The cost function for edges in the search graph $C_{\mathbb{A}^\star} : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^+$ is modified to incorporate the feasibility of generating such motion via collaborative pushing, i.e.,

$$C_{\mathbb{A}^\star}(\mathbf{s}_1, \mathbf{s}_2) \triangleq \|\mathbf{s}_1 - \mathbf{s}_2\|_2 + \text{J}_{\text{MF}}(\xi^\star, \widetilde{\mathbf{s}_1\mathbf{s}_2}), \quad (16)$$

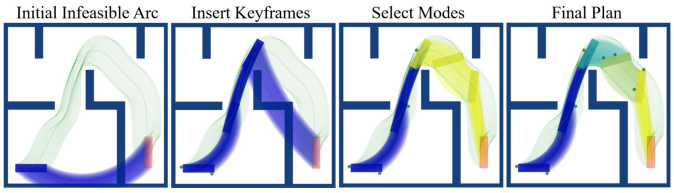

Fig. 8: Illustration of the KG-HS algorithm in Sec. III-C.

where $(\mathbf{s}_1, \mathbf{s}_2)$ is any edge in the search graph; J_{MF} is the loss estimation from (10), within which $\xi^\star$ is the optimal interaction mode for the arc transition $\widetilde{\mathbf{s}_1\mathbf{s}_2}$. Note that given a fixed time step, the minimum loss J_{MF} for different arcs within the local coordinate can be pre-calculated and stored for online evaluation. Denote by $\overline{\mathbf{S}} \triangleq \mathbf{s}_0 \cdots \mathbf{s}_L$ the resulting path.

2) Keyframe-guided Hybrid Search: Given the guiding path of the target, a hybrid plan is determined as a sequence of trajectory segment and the associated interaction mode (including contact points and forces), such that the target can be pushed along the guiding path. As shown in Fig. 8, a hybrid search algorithm called keyframe-guided hybrid search (KG-HS) is presented in this section. It iteratively decomposes the guiding path $\overline{\mathbf{S}}$ into segments of arcs between keyframes, such that each segment can be tracked by single pushing mode.

Specifically, the search space is defined as $\mathcal{V} \triangleq \{\nu\}$, where each node ν is a hybrid plan defined as a sequence of *keyframes*, i.e., $\nu \triangleq \kappa_0 \cdots \kappa_\ell \cdots \kappa_{L_\nu}$, with $\kappa_\ell \triangleq (\mathbf{s}_\ell, \xi_\ell)$ being the ℓ-th keyframe, $\forall \ell = 1, \cdots, L_\nu$. Note that $\mathbf{s}_\ell \in \overline{\mathbf{S}}$ is the state of the target such that $\mathbf{s}_0 = \mathbf{s}_{\text{O}}$ and $\mathbf{s}_{L_\nu} = \mathbf{s}_{\text{G}}$, while ξ_ℓ is the mode for the segment $\widetilde{\mathbf{s}_l\mathbf{s}_{l+1}}$. Furthermore, the cost of a hybrid plan is estimated as follows:

$$\text{C}(\nu) \triangleq \sum_{l=0}^{L_\nu - 1} \text{J}_{\text{MF}}(\xi_\ell, \widetilde{\mathbf{s}_l\mathbf{s}_{l+1}}) + w_{\text{t}} \text{T}_{\text{SW}}(\xi_\ell, \xi_{\ell+1}), \quad (17)$$

where the loss $\text{J}_{\text{MF}}(\cdot)$ is defined in (12); $w_{\text{t}} > 0$ is a design parameter; and $\text{T}_{\text{SW}}(\cdot) > 0$ estimates the time required for the robots to switch from mode ξ_ℓ to $\xi_{\ell+1}$, which is described later. Note that the objective function $J(\cdot)$ in the original problem of (5) is the summed cost of a hybrid plan ν by (17).

First, a priority queue is used to store the nodes to be visited, which is initialized as $Q = \{\nu_0\}$, where $\nu_{\text{O}} \triangleq (\mathbf{s}_{\text{O}}, \varnothing)(\mathbf{s}_{\text{G}}, \varnothing)$ and $\varnothing$ indicates that no mode has been assigned to the arc. Then, the search space $\mathcal{V}$ is explored via an iterative procedure of node selection and expansion, as summarized in Alg. (1).

(I) **Selection**. The node with the lowest estimated cost within Q is popped, i.e., $\nu^\star \triangleq \textbf{argmin}_{\nu\in\mathcal{V}}\{\text{C}(\nu) + \text{H}(\nu)\}$, where $\text{H}(\cdot)$ is an under-estimation of the cost for the segments in ν that are not assigned, i.e., $\text{H}(\nu) \triangleq \sum_{\xi_\ell = \varnothing} \|\widetilde{\mathbf{s}_\ell\mathbf{s}_{\ell+1}}\|$ is the generalized length of arc $\widetilde{\mathbf{s}_l\mathbf{s}_{l+1}}$ from (12).

(II) **Expansion**. The selected node $\nu^\star$ is expanded by finding the first keyframe $\mathbf{k}_\ell$ that has not been *assigned* with a mode, i.e. $\ell = \textbf{min}\big\{\ell \in \{1, \cdots, L_{\nu^\star}\} \,|\, \xi_\ell = \varnothing\big\}$. Then, the following two cases are discussed: **(i)** If the arc $\widetilde{\mathbf{s}_\ell\mathbf{s}_{\ell+1}}$ intersects with any obstacle, a new keyframe $(\mathbf{s}_{\hat{\ell}}, \varnothing)$ is generated by selecting an intermediate state $\mathbf{s}_{\hat{\ell}}$ between $\mathbf{s}_\ell$ and $\mathbf{s}_{\ell+1}$ within the guiding

Algorithm 1: KG-HS Algorithm.

Input: Guiding path $\overline{\mathbf{S}}$ by $\mathtt{A}^\star$ and (16).
Output: Hybrid plan $\nu^\star$.

Initialize $Q = \{\nu_\circ\}$;
while $Q \neq \varnothing$ **do**
  /* **Selection** */
  $\nu^\star = \mathbf{argmin}_{\nu \in Q}\{\mathcal{C}(\nu) + \mathcal{H}(\nu)\}$ by (17);
  Store $\nu^\star$ if feasible;
  /* **Expansion** */
  $\ell = \mathbf{min}\{\ell \in \{1, \cdots, L_{\nu^\star}\} \,|\, \xi_\ell = \varnothing\}$;
  if $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ *is collision-free* **then**
    $\mathbf{\Xi} = \mathtt{MG\text{-}SO}(\mathbf{s}_\ell, \mathbf{s}_{\ell+1})$ by (13);
    $\nu^+ = \mathtt{REP}\big(\nu^\star, \ell, (\mathbf{s}_\ell, \xi^\star)\big)$ by (20), $\forall \xi^\star \in \mathbf{\Xi}$;
    if $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ *is not allowed in* $\mathbf{\Xi}$ **then**
      $\boldsymbol{\kappa}_\ell = \mathtt{SATA}\big(\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}, e\big)$ by (21);
      $\nu^+ = \mathtt{REP}(\nu^\star, \ell, \boldsymbol{\kappa}_\ell)$ by (20);
  else
    Select $\mathbf{s}_{\widehat{\ell}}$ by (18) and compute $\widehat{\mathbf{S}}_{\widehat{\ell}}$;
    $\nu^+ = \mathtt{INS}\big(\nu^\star, \ell, (\hat{\mathbf{s}}, \varnothing)\big)$ by (19), $\forall \hat{\mathbf{s}} \in \widehat{\mathbf{S}}_{\widehat{\ell}}$;
  $Q \leftarrow Q \cup \{\nu^+\}$;
  /* **Termination** */
  if Q *is empty or the time limit is reached* **then**
    Return the best $\nu^\star$;

path $\overline{\mathbf{S}}$, which satisfies the following condition:

$$\big(\|\mathbf{s}_{\widehat{\ell}} - \mathbf{s}_\ell\| - \alpha_{\min}\big)\big(\|\mathbf{s}_{\widehat{\ell}} - \mathbf{s}_{\ell+1}\| - \alpha_{\min}\big) > 0, \tag{18}$$

where $\alpha_{\min} > 0$ is a chosen parameter such that if $\|\mathbf{s}_1 - \mathbf{s}_2\| \leqslant \alpha_{\min}$, the arc $\overset{\frown}{\mathbf{s}_1 \mathbf{s}_2}$ is collision-free, $\forall \mathbf{s}_1, \mathbf{s}_2 \in \overline{\mathbf{S}}$. This condition ensures that the length of both segments is either greater than $\alpha_{\min}$ or less than $\alpha_{\min}$, thus avoiding uneven segmentation. Thus, a new node ν^+ is generated by inserting the keyframe $(\mathbf{s}_{\widehat{\ell}}, \varnothing)$ between κ_ℓ and $\kappa_{\ell+1}$ within $\nu^\star$, i.e.,

$$\nu^+ \triangleq \kappa_0 \cdots \kappa_\ell \, (\mathbf{s}_{\widehat{\ell}}, \varnothing) \, \kappa_{\ell+1} \, \cdots \, \kappa_{L_{\nu^\star}}, \tag{19}$$

such that the original arc $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ is split into two sub-arcs $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\widehat{\ell}}}$ and $\overset{\frown}{\mathbf{s}_{\widehat{\ell}} \mathbf{s}_{\ell+1}}$. This step is encapsulated by $\nu^+ \triangleq \mathtt{INS}\big(\nu, \ell, (\mathbf{s}_{\widehat{\ell}}, \varnothing)\big)$. Meanwhile, to account for possible disturbances, a set of close-by states $\widehat{\mathbf{S}}_{\widehat{\ell}}$ are generated by adding perturbations to $\mathbf{s}_{\widehat{\ell}}$, e.g., small rotation and translation. The associated set of new nodes $\{\nu^+\}$ are all added to Q; (**ii**) If $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ is collision-free, a set of feasible modes is generated by solving the associated optimization in (13), denoted by $\mathbf{\Xi}(\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}})$. For each mode $\xi^\star \in \mathbf{\Xi}$, a new node is generated by replacing the keyframe $(\mathbf{s}_\ell, \varnothing) \in \nu^\star$ with $(\mathbf{s}_\ell, \xi^\star)$, i.e.,

$$\nu^+ \triangleq \kappa_0 \cdots \kappa_{\ell-1} \, (\mathbf{s}_\ell, \, \xi^\star) \, \kappa_{\ell+1} \cdots \kappa_{L_{\nu^\star}}, \tag{20}$$

which is encapsulated by $\nu^+ \triangleq \mathtt{REP}\big(\nu, \ell, (\mathbf{s}_\ell, \xi^\star)\big)$. Then the set of new nodes $\{\nu^+\}$ are added to the queue Q. Meanwhile, as discussed, it is possible that the arc $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ may not be allowed in any mode $\xi^\star \in \mathbf{\Xi}^\star$. In this special case, the method described in the proof of Theorem 1 is applied to approximate the segment of path $\overline{\mathbf{S}}$ by a sequence of shorter arcs with a error bound in Hausdorff distance. For brevity, this process of approximation via sequential arc transitions is denoted by:

$$\boldsymbol{\kappa}_\ell \triangleq \mathtt{SATA}\big(\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}, e\big), \tag{21}$$

where $\boldsymbol{\kappa}_\ell$ is the resulting sequence of keyframes between $\mathbf{s}_\ell$ and $\mathbf{s}_{\ell+1}$; and $e > 0$ is the chosen bound. Thus a new node is generated by replacing the keyframe $(\mathbf{s}_\ell, \varnothing) \in \nu^\star$ with $\boldsymbol{\kappa}_\ell$, i.e., $\nu^+ \triangleq \mathtt{REP}(\nu, \ell, \boldsymbol{\kappa}_\ell)$, which is also added to the queue Q.

(III) **Termination**. If all the keyframes in the selected node $\nu^\star$ are assigned with modes, it indicates that $\nu^\star$ is a *feasible* solution. More importantly, the arc transitions in $\nu^\star$ are collision-free, and all allowed in the mode set Ξ. The KG-HS algorithm continues until Q is empty or the maximum planning time is reached. Once terminated, the current-best hybrid plan is selected and sent to the robots for execution.

Theorem 2. *Given that the set of modes Ξ is sufficient and the guiding path $\overline{\mathbf{S}}$ is collision-free, the proposed KG-HS algorithm finds a feasible hybrid plan $\nu^\star$ in finite steps.*

Proof: (Sketch) The key insight is that any infeasible arc is split into two arcs with shorter length, until all the arcs are collision-free. Therefore, there must exists a minimum arc length $\alpha_{\min} > 0$ such that any segments $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ in $\nu^\star$ with length less than $\alpha_{\min}$ is collision-free. Consequently, the search depth is bounded by $2|\overline{\mathbf{S}}|/\alpha_{\min}$ with $|\overline{\mathbf{S}}|$ being the total length, yielding that the termination condition is reached in finite steps. Lastly, since the set of modes Ξ is sufficient, the set of arcs $\overset{\frown}{\mathbf{s}_l \mathbf{s}_{l+1}}$ is feasible when $\alpha_{\min}$ is sufficiently small by Theorem 1. Thus, the KG-HS algorithm is guaranteed to find a feasible solution in finite steps. ■

D. Online Execution and Adaptation

The online execution of the hybrid plan $\nu^\star$ consists of three parts: (i) trajectory tracking under each arc segment with the specified mode; (ii) mode switching between segments; and (iii) event-triggered adaptation of the hybrid plan.

1) Trajectory Tracking: The nonlinear model predictive control (NMPC) framework from Allgower et al. [1] is adopted for the robots such that the object pose can track the arc transitions $\overset{\frown}{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ with mode ξ_ℓ. The main motivation is to account for motion uncertainties and perturbations both in robot motion and pushing, such as slipping and sliding. More specifically, the tracking problem is formulated as an online optimization problem for the pushing forces $\mathbf{q}_\xi(t)$ and the refined target trajectory $\mathbf{s}(t)$, i.e.,

$$\begin{aligned} \min_{\{\mathbf{q}_\xi^{\mathrm{B}}(t), \mathbf{s}(t)\}} \sum_{t \in \widetilde{\mathcal{T}}} \Big(\big\|\mathbf{q}_\xi^{\mathrm{B}}(t) - \eta(\mathbf{p}^{\mathrm{B}}(t))\big\|_1 + w_{\mathrm{I}}(\Delta \mathbf{p}_t)^2 \Big) \\ \textbf{s.t.} \quad \mathbf{s}(0) = \widetilde{\mathbf{s}}_0, \ \mathbf{s}(T) = \widetilde{\mathbf{s}}_{\mathrm{G}}, \ \Omega(t) \subset \widehat{\mathcal{W}}; \\ \mathbf{s}(t+1) = \mathbf{s}(t) + \mathbf{R}_{\mathrm{B}} \mathbf{p}^{\mathrm{B}}(t), \ \forall t \in \widetilde{\mathcal{T}}; \\ \mathbf{q}_\xi^{\mathrm{B}}(t) \in \mathrm{Q}_\xi^{\mathrm{B}}, \ \forall t \in \widetilde{\mathcal{T}}; \end{aligned} \tag{22}$$

where $\widetilde{T}$ is the planning horizon of NMPC; $\widetilde{\mathcal{T}} \triangleq \{1, \cdots, \widetilde{T}\}$; $\Delta \mathbf{p}_t \triangleq \mathbf{p}^{\mathrm{B}}(t) - \mathbf{p}^{\mathrm{B}}(t-1)$ is the acceleration of the object;

$w_{\mathrm{I}} > 0$ is a design parameter; $\widetilde{\mathbf{s}}_0$ is current state of the target; and $(\widetilde{\mathbf{s}}_{\mathrm{G}}, \widetilde{\xi})$ are the local reference state and the associated mode, selected from the hybrid plan $\nu^\star$. The second objective measures the smoothness of the resulting trajectory, which is ignored in quasi-static analysis but considered here.

The optimization above can be solved by a nonlinear optimization solver, e.g., `IPOPT` within CasADi from Andersson et al. [2]. Denote by $\mathbf{S}^\star(t) \triangleq \mathbf{s}_1 \cdots \mathbf{s}_{\widetilde{T}}$ the resulting trajectory, of which the first state is chosen as the subsequent desired state $\widehat{\mathbf{s}}$. The associated desired contact point and orientation for robot R_n are given by $\widehat{\mathbf{c}}_n$ and $\widehat{\psi}_n$, $\forall n \in \mathcal{N}$. Thus, their control inputs are given by $\mathbf{v}_n = K_v(\widehat{\mathbf{c}}_n - \mathbf{c}_n)$ and $\omega_n = K_\psi(\widehat{\psi}_n - \psi_n)$, where K_v and K_ψ are positive gains; and $\mathbf{c}_n, \psi_n$ are the current contact point and orientation of R_n. With simple P-controllers and a small step size, the robots can push the target along $\mathbf{S}^\star(t)$, which is updated by solving (22) recursively online at a lower rate.

2) Mode Switching: Initially, since the robots are homogeneous, they are assigned to their closest contact points given the first mode ξ_0. Afterwards, as the segment $\overline{\mathbf{s}_\ell \mathbf{s}_{\ell+1}}$ is traversed, the fleets switch from mode ξ_ℓ to $\xi_{\ell+1}$. To reduce switching time and avoid possible collisions, the switching strategy is designed as follows. Let the set of old contact points be $\{\mathbf{c}_n\}$ for mode ξ_ℓ and the new set $\{\mathbf{c}_n^+\}$ for mode $\xi_{\ell+1}$. First, the periphery of the object is transformed into a circle such that the relative ordering of the contact points is preserved. Second, an diameter associated with angle $\theta^\star \in [0, \pi]$ is chosen such that the number of old contact points and the number of new contact points on both side of the diameter are equal. Consequently, the points in $\{\mathbf{c}_n^+\}$ are numbered clockwise starting from $\theta = \theta^\star$, while the same applies for the robots at $\{\mathbf{c}_n\}$. Lastly, each robot is assigned to the new contact point with the same number and a local path is planned for navigation. It can be shown that no collisions between the robots can happen and the length of each path is not longer than half of the perimeter, if the minimum distance between the target and obstacles is greater than the robot radius; and all robots move at similar speed, as illustrated in Fig. 2.

3) Online Adaptation: Due to model uncertainties such as communication delays and actuation noises or slipping, the NMPC scheme in (22) might not be able to rectify the system towards the reference state, i.e., the object can not follow the refined trajectory $\mathbf{S}^\star$. In this case, a re-planning mechanism for the hybrid plan $\nu^\star$ is necessary to find a new guiding path and a new associated hybrid plan. The triggering condition is rule-based: (i) the target object deviates from the planned trajectory beyond a given threshold; (ii) the distance between the object and obstacles is less than the size of the robot; (iii) the object is stuck within an area for a certain period; (iv) new static obstacles are detected; or (iv) one or several robots fail and can not participate in the task anymore. Then, as shown in Fig. 2, the KG-HS algorithm is executed again with the current system state as the initial state and the functional robots, yielding a new hybrid plan $\nu^\star$. The NMPC is recomputed to generate new reference position for the robots. Similar approaches can be applied to the case of dynamic obstacles, where the safety constraint in the NMPC can modify the $\mathbf{S}^\star$ given the perceived or predicted motion of the obstacles. As validated by numerical results in Sec. IV, this mechanism is effective and essential for improving the overall robustness, especially in uncertain environments or during robot failures. Last but not least, if measurements of the object state are not always available or temporary lost. An extended Kalman filter (EKF) can be adopted to estimate the object state at all time, i.e., given the desired target velocity, the contact points, and the state of all robots. Detailed formulations are omitted here for brevity.

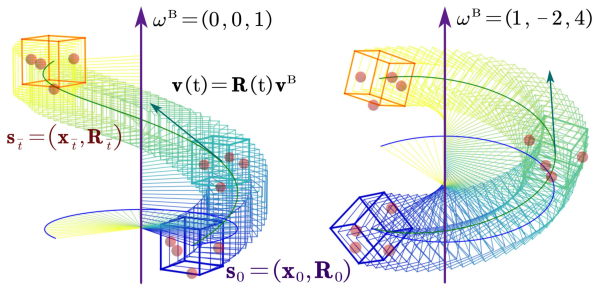

Fig. 9: Illustration of the spiral transition in 3D workspace.

E. Discussion

1) Computation Complexity: The time complexity of the proposed KG-HS algorithm is mainly determined by the mode generation, the collision checking and the size of the search space. The MG-SO method in (13) for mode generation is a LP, with the variable size proportional to the number of all contact points $N_{\mathcal{V}}$. Thus, it has a complexity of $\mathcal{O}(N_{\mathcal{V}}^{3.5})$ from Terlaky [23]. The time complexity of the GJK algorithm for 2D collision checking from Bergen [3] is $\mathcal{O}(V + V_O)$, where V_O is the maximum number of vertices for the obstacles. Meanwhile, for each iteration of node expansion in Alg. 1, at most W_k keyframe variations in the first case or W_ξ modes in the second case are generated. Meanwhile, the computation of the loss J_{MF} in Eq. (17) is a LP with the variable size proportional to the number of robots, so the complexity is $\mathcal{O}(N^{3.5})$. Recall that the search depth is bounded by $2\alpha/\alpha_{\min}$ from Theorem 2. Therefore, the total time complexity of the hybrid search algorithm is approximated by $\mathcal{O}\Big((N_{\mathcal{V}}^{3.5} + N^{3.5} + V + V_O) \cdot \big(\max\{W_\xi, W_k\}\big)^{2\alpha/\alpha_{\min}}\Big)$. More numerical analyses on how the proposed algorithm scales across different fleet sizes can be found in Sec. IV-A3.

2) Additional Heuristics: Some additional heuristics are applied to further improve the search performance: (I) Selection of new keyframes. Instead of simply choosing the middle point between $\mathbf{s}_\ell$ and $\mathbf{s}_{\ell+1}$ in (19), the new keyframe $\mathbf{s}_{\hat{\ell}^\star}$ can be selected by minimizing the distance between the arc transitions $(\overline{\mathbf{s}_\ell \mathbf{s}_{\hat{\ell}^\star}}, \overline{\mathbf{s}_{\hat{\ell}^\star} \mathbf{s}_{\ell+1}})$ and the obstacles, typically yielding the tuning points on the guiding path; (II) Local reachability. The close-by states $\widehat{\mathbf{S}}_{\hat{\ell}}$ in Line 13 of Alg. 1 are first filtered to ensure all states inside are reachable via a collision-free arc from $\mathbf{s}_\ell$, to avoid adding infeasible nodes in the search tree; (III) Inflation of obstacles. The static obstacles are inflated by at least the robot radius when generating the guiding path. Moreover, a

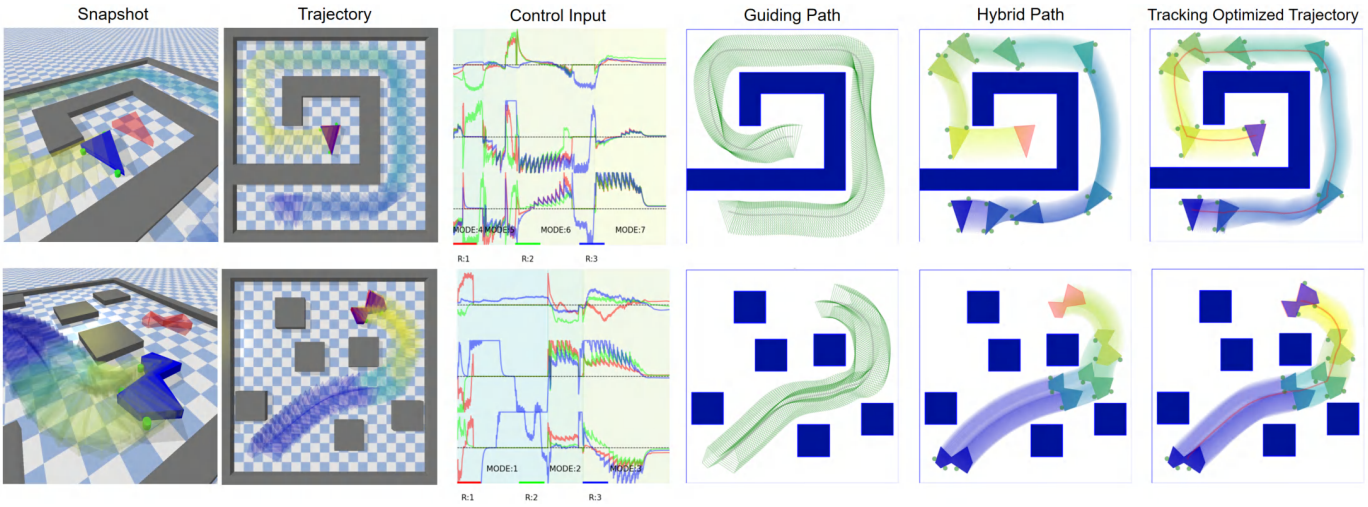

Fig. 10: Simulation results of three robots pushing different target objects (goal state in red), including the guiding path $\overline{\mathbf{S}}$, the hybrid plan $\nu^\star$ with keyframes and modes, the final trajectory $\mathbf{S}^\star$ after execution, and the control inputs of the robots.

differentiable penalty is added in the objective of (22) for a sufficient safety margin of the robots.

3) Generalization: The proposed method can be generalized to different scenarios: (I) Heterogeneous robots. When the robots are heterogeneous with different capabilities, the interaction mode should be defined as $\xi \triangleq (\mathbf{c}_1, R_{c_1}) \cdots (\mathbf{c}_N, R_{c_N})$, which assigns specific robot to each contact point $\mathbf{c}_n$ based on the associated force constraints. For instance, robots with larger maximum forces are assigned to contact points that require larger forces, while robots with smaller forces are assigned to assistive contact points that provide additional geometric constraints; (II) 3D pushing tasks. If the task is to push a target object with 6D pose in a complex 3D scene, the proposed framework can be adopted with minor modifications, under the condition that the workspace is zero-gravity and resistant, due to the quasi-static assumption. More specifically, consider a target object of which the state is denoted by $\mathbf{s}(t) \triangleq (\mathbf{x}(t), \mathbf{R}(t))$, where $\mathbf{x}(t) \in \mathbb{R}^3$ is the position and $\mathbf{R}(t) \in \mathbb{R}^{3\times3}$ is the rotation matrix for orientation. Additionally, the robots are modeled as spheres with radius r, with the translational velocity $\mathbf{v}_n \in \mathbb{R}^3$ as control inputs. When moving dynamically, the target experiences a force of resistance that is proportional to its generalized velocity, i.e., $\mathbf{q}_{\texttt{res}} \triangleq -\mathbf{K}_{\texttt{res}}\mathbf{p} \in \mathbb{R}^6$ with $\mathbf{K}_{\texttt{res}}$ being a diagonal matrix, and $\mathbf{p} = (\mathbf{v}, \boldsymbol{\omega}) = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z) \in \mathbb{R}^6$ as the generalized velocity. Similar to the 2D case, consider that the robots push the target with velocity $\mathbf{p}(t) = \mathrm{diag}(\mathbf{R}(t), \mathbf{R}(t))\mathbf{p}^{\mathrm{B}}$ for a time duration $\bar{t} > 0$. Then, the arc transition in (11) can be extended to 3D workspace by:

$$
\begin{aligned}
&\mathrm{d}\mathbf{R}(t) \triangleq \boldsymbol{\omega}(t) \times \mathbf{R}(t)\,\mathrm{d}t = (\mathbf{R}_0\,\boldsymbol{\omega}^{\mathrm{B}}) \times \mathbf{R}(t)\,\mathrm{d}t,\\
&\mathbf{R}(t) = \mathbf{P}_{\boldsymbol{\omega}} \begin{bmatrix} \mathbf{Rot}\left(\|\boldsymbol{\omega}^{\mathrm{B}} t\|_2\right) & 0 \\ 0 & 1 \end{bmatrix} \mathbf{P}_{\boldsymbol{\omega}}^{-1} \mathbf{R}_0,\\
&\Delta\mathbf{x} \triangleq \Big(\int_{t=0}^{\bar{t}} \mathbf{R}(t)dt\Big)\,\mathbf{v}^{\mathrm{B}} = \boldsymbol{\Phi}_{3\mathrm{D}}(\mathbf{R}_0,\, \boldsymbol{\omega}^{\mathrm{B}}\bar{t})\;\bar{t}\;\mathbf{v}^{\mathrm{B}},
\end{aligned}
\tag{23}
$$

where $\mathbf{P}_{\boldsymbol{\omega}} \in \mathbb{R}^{3\times3}$ is an unitary matrix of which the 3-th column is aligned with the rotation axis $\boldsymbol{\omega}$, i.e. $\mathbf{P}_{\boldsymbol{\omega}}(3) = \mathbf{R}_0\boldsymbol{\omega}^{\mathrm{B}}$. As shown in Fig. 9, this motion results in a spiral trajectory in 3D space, i.e., a combination of a constant rotation about the axis $\boldsymbol{\omega}$ and a translation along $\boldsymbol{\omega}$.

Lemma 3. *For any given pair of initial state* $\mathbf{s}_0$ *and goal state* $\mathbf{s}_G$ *of the object in the 3D workspace, there exists a spiral trajectory* $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$ *such that* $\|\boldsymbol{\omega}\bar{t}\|_2 \in [0, \pi]$.

Proof: (Sketch) This is a straightforward corollary of the Chasles's theorem [30]. Analogous to the 2D case, the desired velocity $\mathbf{p}^{\mathrm{B}} = (\mathbf{v}^{\mathrm{B}}, \boldsymbol{\omega}^{\mathrm{B}})$ can be determined by (23), yielding the spiral trajectory $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\mathrm{B}}, \bar{t})$. ■

Moreover, in total $D_{3\mathrm{D}} = 12$ directions are chosen to evaluate the multi-directional feasibility $J_{\texttt{MF}}$. Similarly, the mode generation in (13) should adapt to the constraints resulting from the 3D friction cone, by allowing 3 degrees of freedom for the contact force at each point. Consequently, the number of optimization variables becomes $N_{\mathcal{V}} \times 3D_{3\mathrm{D}}$. Note that the hybrid search algorithm KG-HS still applies, with the 6D statespace $\mathcal{S}$. Lastly, the control input of each robot R_n is adjusted to be $\mathbf{v}_n = K_v(\hat{\mathbf{c}}_n - \mathbf{c}_n)$, $\forall n \in \mathcal{N}$.

IV. Numerical Experiments

To further validate the proposed method, extensive numerical simulations and hardware experiments are presented in this section. The proposed method is implemented in Python3 and tested on a laptop with an Intel Core i7-1280P CPU. Numerical simulations are conducted in the `PyBullet` environment as a high-fidelity physics engine from Coumans and Bai [5], while `ROS1` is adopted for the hardware experiments. The `GJK` package from Bergen [3] is used for collision checking, and the `CVXOPT` package from Diamond and Boyd [6] for solving liner programs. Simulation and experiment videos are available in the supplementary material.

A. Numerical Simulations

1) System Setup: Three distinct scenarios of size $20m \times 20m$ with concave and convex obstacles are considered, including the first scenario as shown in Fig. 2, and the second

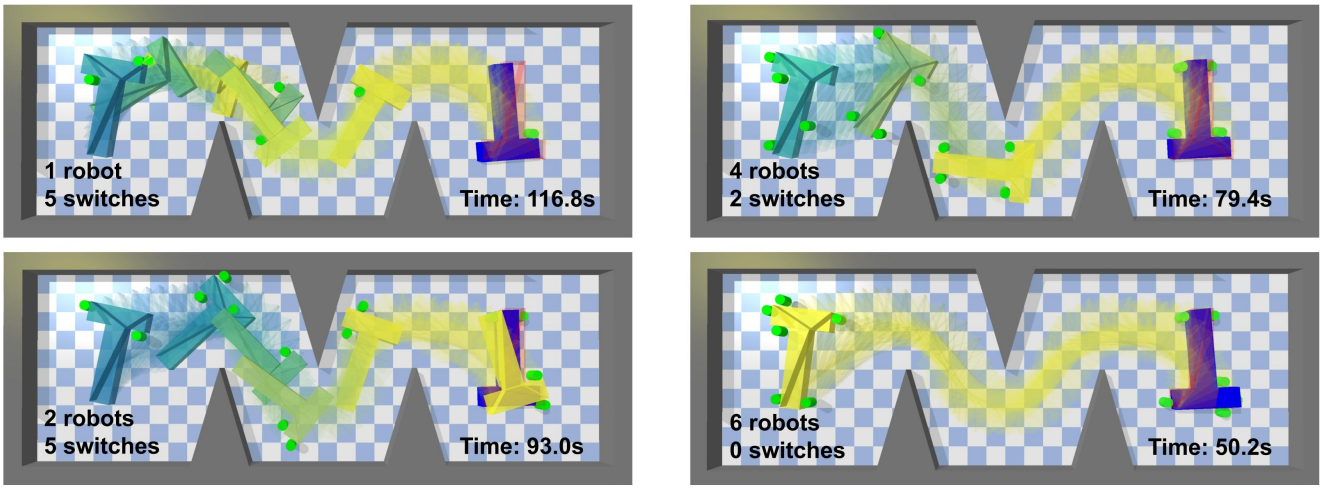

Fig. 11: Final trajectories when different number of robots are deployed, yielding different modes and task durations.

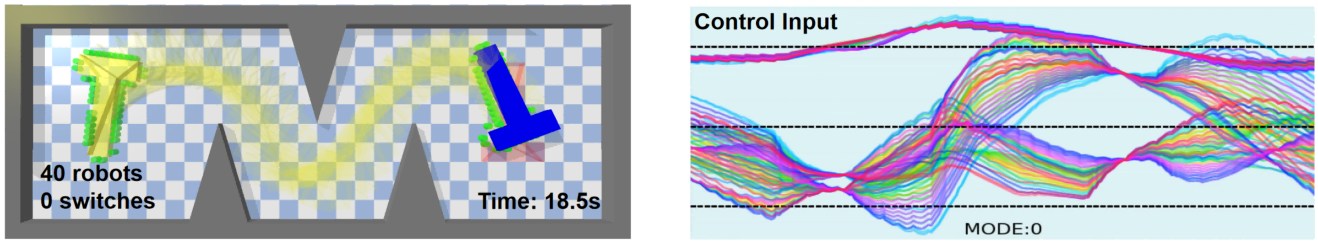

Fig. 12: Scalability analysis for 40 robots: final trajectories (**Left**) and control inputs of all robots (**Right**).

and third scenarios in Fig. 10. The target object has different shapes and sizes in each scenario, with a mass of $10kg$, a ground friction coefficient of 0.5, a side friction coefficient of 0.2, and a maximum thrust of 30N. Three homogeneous robots with a diameter of $0.25m$ are adopted for the pushing task. More specifically, a long rectangular object should be pushed through a narrow passage in the first scenario; a triangular object should be pushed following a spiral corridor in the second scenario; and a concave object should pass through cluttered rectangular pillars in the last scenario. The task is considered completed if the distance between the target object and the goal is less than $0.2m$, after which the robots can continue improving the accuracy. The Pybullet simulator runs at a frequency of 240Hz. The MPC-based online optimization for object trajectory updates at a frequency of $10Hz$, with a time horizon of $2s$ and an average speed of $0.5m/s$. The frequency to update the intermediate reference for MPC is $1Hz$. Other parameters are set as follows: the weight for multi-directional feasibility $[w_d] = [5, 1, 1, 1, 1, 1]$ in (9), the weight $w_{\text{t}} = 10$ in (17), the weight $w_{\text{I}} = 10^4$ in (22), the minimum split length $\alpha_{\min} = 0.1m$ in (9).

2) Results: As is shown in Fig. 2 and Fig. 10, the proposed method successfully completed all tasks in three distinctive scenarios. The intermediate results are also shown including the guiding path, the hybrid plan (with keyframes and modes highlighted in different colors), the resulting trajectory of tracking the hybrid plan via the proposed online NMPC, and the control inputs of all robots during execution. The planning time of three scenarios is $12.3s$, $13.2s$ and $21.1s$, while the execution time is $79.2s$, $117s$ and $64.1s$. The three tasks take 3, 7 and 3 mode switches, which is due to the fact that more rotations are required for the second scenario. The proposed NMPC can effectively track the optimized trajectory, with the average tracking errors being $0.025m$,$0.026m$,and $0.048m$. More metrics are discussed in the comparisons below. Moreover, it is interesting to note that despite the NMPC problem in (22) can be solved in approximately $40ms$, yielding a maximum rate of $25Hz$, higher updating rates for NMPC does not significantly improve control performance. Instead, the tracking control is more sensitive to the update rate of the intermediate goal selected from the hybrid plan. If the goal is chosen too close, the resulting trajectory might have drastic changes of the velocity, while a too far goal might lead to a delayed response to errors. Therefore, a lower rate to update the intermediate goal would allow a more gradual convergence. Lastly, it is worth mentioning that in all three scenarios the online execution of the hybrid plan does not require any replanning, different from the hardware experiments.

3) Complexity and Scalability: To evaluate the computational complexity and scalability of the proposed method, a larger environment of $30m \times 10m$ is considered where the target is a T-shape object with a mass of $5kg$ and a ground friction coefficient of 0.5. Since the maximum pushing force of one robot is $30N$, a single robot can move this object. As shown in Fig. 11, the proposed method is applied to robotic fleets with 1, 2, 4 and 6 robots for the same task, where the robots are randomly initialized. The planning time is $10.3s$, $13.5s$, $15.2s$ and $18.1s$, while the execution time is

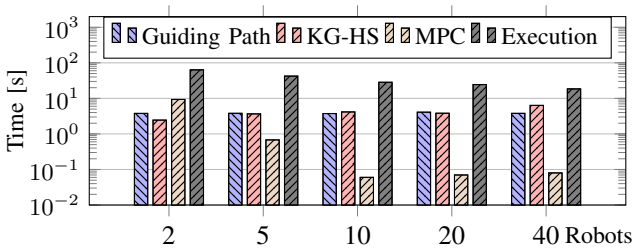

Fig. 13: Computation time of each component across different fleet sizes.

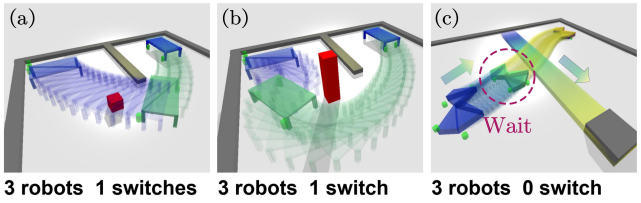

Fig. 14: Online adaptation under unknown obstacles (in red) and dynamic obstacles (in shaded grey).

$116.8s$, $93.0s$, $79.4s$ and $50.2s$. The resulting hybrid plans and trajectories are shown, where the required number of modes are 6, 6, 3 and 1, respectively; and the average tracking errors are $0.08m$, $0.05m$, $0.01m$ and $0.01m$. It indicates that fewer robots often require more mode switches to complete the task, while more than 6 robots only need one mode.

Furthermore, to demonstrate the applicability to even larger robot teams, the radius of the robots is reduced to $0.15m$ and the maximum pushing force to $15N$. Under the same setting, the proposed method is applied again to 5, 10, 20 and 40 robots, of which the computation time of the main components are summarized in Fig. 13. As expected, to search for the guiding path is independent of the fleet size (around $3.7s$). The computation time of KG-HS aligns with our complexity analysis in Sec. III-E1, which takes $2.5s$ for 2 robots and $6.4s$ for 40 robots. The cost of MG-SO and GJK is independent of the number of robots, so the increase in time primarily stems from the $O(N^{3.5})$ complexity of computing J_{MF}. Lastly, it is interesting to note that the computation time for NMPC decreases as the number of robots increases. This is because the arc transition $\mathbf{Trj}(\mathbf{s}_0, \mathbf{p}^{\text{B}}, \bar{t})$ for any planned mode can already be tracked accurately with enough robots, which complies with the intuition that the pushing task is easier with more robots. This leads to an offline planning time of only around $20s$ for 40 robots and near-zero online planning time, with a high tracking accuracy as shown in Fig. 12.

4) Adaptability: To further demonstrate the adaptability of the proposed method, the following scenarios are evaluated.

(I) **Unknown and dynamic obstacles**. Since some obstacles might be unknown during offline planning, it is important to adapt to them during online execution. Fig.14 shows two cases that might appear: (a) the guided path and refined trajectory of the object does not collide with the small obstacle; and (b) a potential collision is detected with a larger obstacle, and a new hybrid plan is obtained via the adaptation scheme described in Sec. III-D3. Moreover, in case of a dynamic obstacle, the current pose and predicted trajectory of the obstacle is incorporated in the NMPC constraints in (22). Consequently, as shown in Fig.14, the robots slow down and wait for the obstacle to pass, while following the *same* hybrid plan.

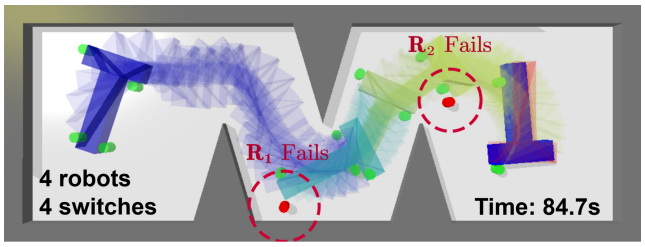

Fig. 15: Plan adaptation when two robots (in red) fail consecutively during execution.

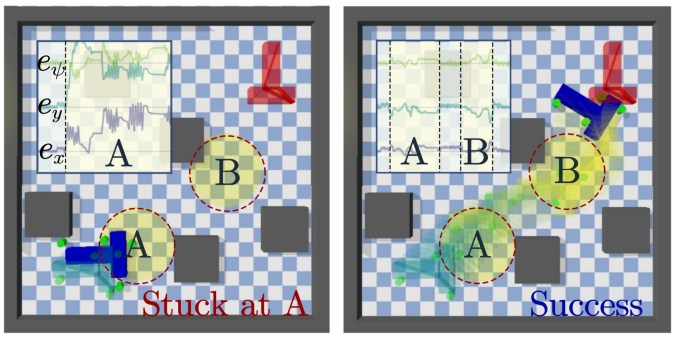

Fig. 16: Robustness to temporary "blackout" (in Regions A and B), where no measurements of the object state are obtained. Final trajectories and state estimation error are shown by linear extrapolation (**Left**) and constrained EKF (**Right**).

(II) **Robot failures**. As the robots fail, the proposed online adaptation is essential for the system to recover. Fig.15 shows the same pushing task as in Fig. 10 with 4 robots, but 2 of them fail consecutively at $25.2s$ and $58.3s$. Via the replanning scheme, a new hybrid plan is generated each time a robot fails. The task is successfully completed with the remaining 2 robots at time $84.7s$, which however requires 3 mode switches (compared with 2 switches in the nominal case). It is worth noting that since the failed robots are treated as obstacles during re-planning, the resulting hybrid plan avoids collision with these robots.

(III) **Temporary "blackout"**. Accurate observation of the state of the target object might not be always possible, e.g., the localization system for the object might be occluded temporarily thus no measurement of the object state is available. As described in Sec. III-D3, an EKF is adopted to estimate the object state, given the states of the robots and the current pushing mode. Fig.16 shows the scenario where the objects can not be measured within two circular areas on the guided path. The proposed estimator has an accuracy of $0.05m$ in position and $0.05rad$ in orientation, which leads to a successful task under 26% blackout in time. In contrast, significant mismatch is generated via simple linear extrapolation, yielding a trajectory that is completely off in (22). Consequently, the robots deviate quickly from the correct path.

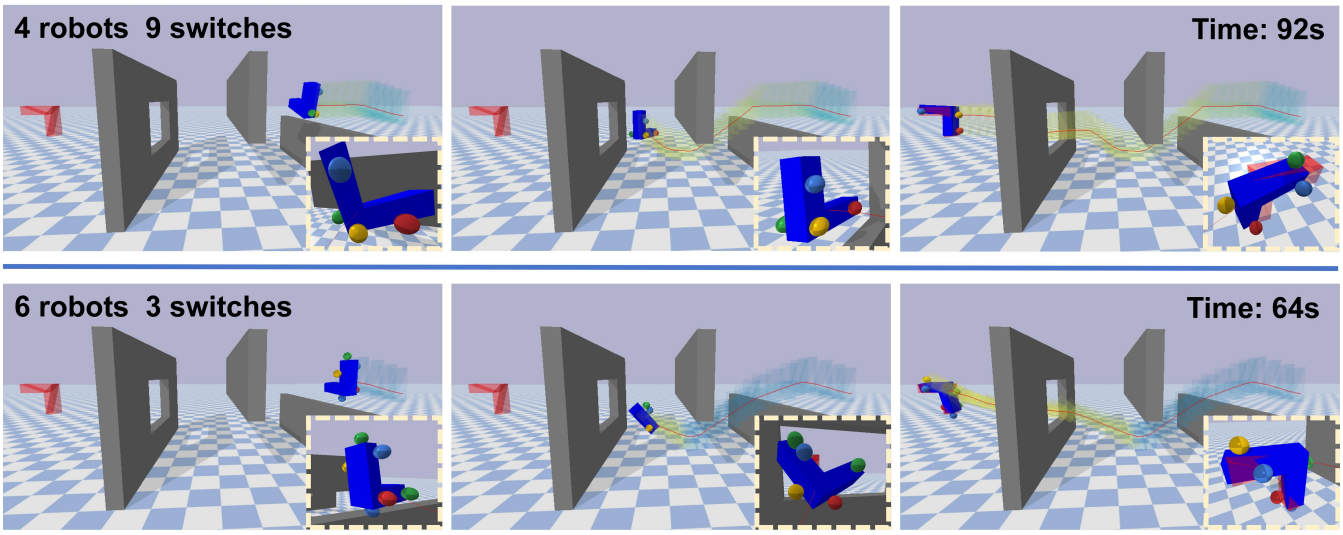

Fig. 17: Snapshots of 3D pushing tasks, where 4 robots (**Top**) and 6 robots (**Bottom**) are deployed.

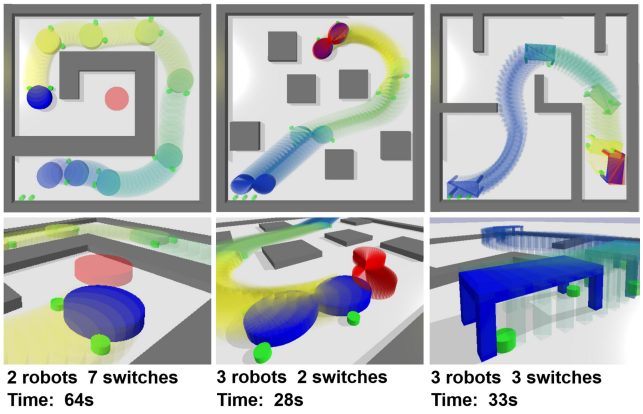

Fig. 18: Extension to non-polytopic objects: cylinder (**Left**), 8-shape (**Middle**), and table (**Right**).

5) Generalization: As discussed in Sec. III-E3, several notable generalizations of the proposed method are evaluated.

(I) **Non-polytopic objects**. As shown in the Fig.18, the proposed approach is tested with three objects with general shape: cylinder, 8-shape, and a table, i.e., non-polytopic objects. To obtain all possible contact points for general shapes, we traverse all possible positions of robots near the 3D boundaries of the object with a certain discretization precision, and further refine these points to ensure close contact with the object via 3D collision checking. Then, the same planning algorithm is applied for $2, 3, 3$ robots. The resulting hybrid plan and trajectories are shown in Fig.18: the cylinder is pushed to the goal with 8 modes and $64.0s$, while the 8-shape reaches the goal with 3 modes and $28.4s$. It is interesting to note that although the table is only partially in contact with the ground, the algorithm can find the optimal pushing modes around four legs of the table with 3 mode switches.

(II) **Heterogeneous robots**. Three heterogeneous robots are deployed for the same pushing task as in Fig. 11, i.e., the maximum pushing forces $f_{n,\max}$ in (1) are $60N$, $40N$ and $20N$, respectively. The assignment and planning strategy as described in Sec. III-E3 is adopted, completing the task with 4 modes and $60.8s$. Compared with the nominal case where all robots are identical, the assignments of robots to contact points are significantly different: (i) The robot with the largest force is always assigned to the contact point that requires the largest force (the top-left corner); (ii) the robot with the smallest force is assigned to the assistive contact points (the bottom-right corner). Lastly, if this heterogeneity is neglected, the task can still be completed but with a much longer time of $82.2s$.

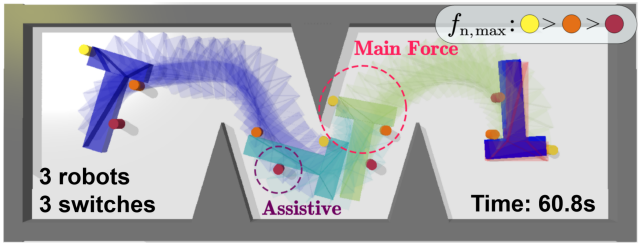

Fig. 19: Pushing task with 3 heterogeneous robots with different maximum forces.

(III) **6D pushing in 3D workspace**. The extension to pushing tasks in complex 3D workspace is demonstrated here with a L-shape object and different number of robots. The object has a mass of $5kg$ with the resistance coefficient $\mathbf{K} = 100$, while the robots have a radius of $0.15m$ and a maximum pushing force of $20N$. As shown in Fig. 17, the workspace has a size of $10m \times 20m \times 5m$ with two walls and a small window. The first row shows the snapshots of 4 robots performing the task, which requires 10 modes and a total mission time of $92.2s$. In contrast, the second row shows the results for 6 robots, which requires 4 modes and a total mission time of $64.3s$. Thus, similar to the 2D case, a larger fleet often requires less number of mode switches and faster completion.

6) Comparisons: The following **four** baselines are compared with the proposed method (**KG-HS**):

(I) **POCB**, which follows the occlusion-based method proposed in [4], i.e., assuming that a light source is placed at the goal point, each robot randomly selects a contact point on the occluded surface and applies a certain amount of pushing

TABLE I: Comparison with four baselines.

Algorithm	SR[1]	TE[2]	CC[3]	SM[4]	PT[5]	ET[6]	EE[7]
KG-HS	1.00	0.03	2241	0.31	13.21	44.31	0.14
AUS	1.00	0.03	2521	0.52	12.28	51.29	0.14
SDF	0.98	0.05	3369	0.58	20.88	60.18	0.14
POCB	0.57	1.94	6066	1.86	0.15	128.47	1.61
IAB	0.65	1.82	5130	0.73	3.56	117.31	1.70

[1] Success rate; [2] Tracking error; [3] Control cost; [4] Smoothness; [5] Planning time; [6] Execution time; [7] End Error.

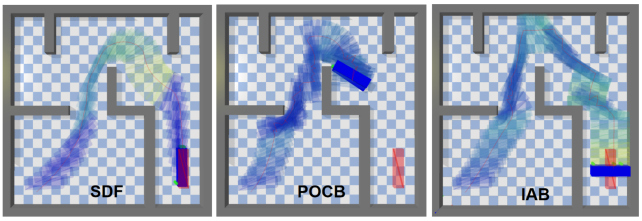

Fig. 20: Trajectories of baselines for the first scenario.

force. Since the original algorithm only applies to star-shaped objects and mainly for free-space (intermediate goal points are set manually for cluttered environments), two modifications are made for the comparison: (i) the occlusion condition is replaced with an angular condition, i.e., $\Delta\theta_{\mathrm{G,n}} \triangleq \theta_{\mathrm{G}} - \theta_{\mathrm{n}} \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, where θ_{G} is angle between the target center and the goal; and θ_{n} is the angle of the normal vector at the contact point; and (ii) the A* algorithm is applied to generate a guiding path consisting of intermediate goal points;

(II) **IAB**, which enhances POCB by introducing a probability of choosing a contact point, which is inversely correlated with the angle $\Delta\theta_{\mathrm{G,n}}$. Thus, the robot can prioritize pushing directions that are more aligned towards the target, which is particularly significant for long rectangular objects in Fig. 20.

(III) **SDF**, which replaces MDF in KGHS with single directional feasibility in both mode generation and cost estimation;

(IV) **AUS**, which does not follow the hybrid search framework, but divides the guiding path equally into L segments until the resulting segmented arc trajectories are collision-free with obstacles, as described in Theorem 1.

The tests are repeated 50 times for each method in all three scenarios, by choosing random initial and goal positions for the target object. Sample trajectories are shown in Fig. 20 for the first scenario. As shown in Table I, **seven** metrics are compared. It can be seen that our KG-HS algorithm achieves the best performance in all metrics except for the planning time, as both POCB and IAB requires no optimization, while AUS performs less times of mode generations. Nonetheless, the KG-HS algorithm has the shortest runtime $57.5s$ by summing up the planning and execution time.

More specifically, consider the following three aspects: (I) **Importance of mode generation**. It is worth noting that both AUS and SDF still outperform the POCB and IAB, mainly due to the hybrid search framework and the proposed feasibility analysis. Particularly, both POCB and IAB only take into account the difference between the pushing directions and the

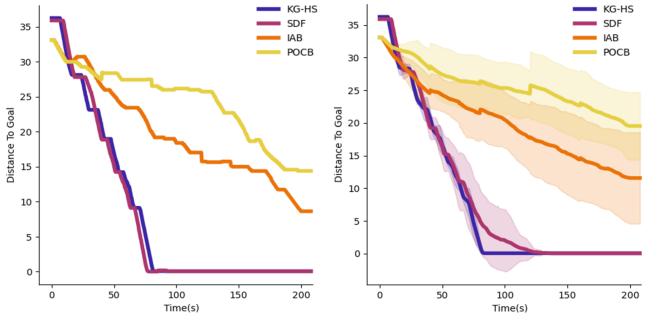

Fig. 21: Distance to goal during execution without (**Left**) and with (**Right**) disturbances, where the shaded area indicates the standard deviation over 50 runs.

goal direction, thus completely neglecting the different torques generated at the contact points. Thus, the target object can only be probabilistically moved to the target position, rather than a continuous smooth trajectory, especially for the first scenario where precise orientations are crucial. In contrast, both SDF and KG-HS consider the expected rotation of the object as determined by the arc transition. (II) **Effectiveness of the hybrid search framework**. The difference in tracking error between AUS and KG-HS algorithms is small as they share the same modules for mode optimization and trajectory optimization. However, compared with the equal segmentation in AUS, the hybrid search process in KG-HS can reduce the execution time by 13% and the control cost by 11%, while improving the smoothness by 42%. (III) **Robustness against disturbances**. For the first scenario, random disturbances are added to the target velocity at a frequency of $10Hz$ with a variance proportional to its actual velocity at each time step. The execution results are summarized in Fig. 21, which shows that a significantly smaller variance in the tracking error is achieved by KG-HS compared to the other methods.

B. Hardware Experiments

1) System Description: As shown in Fig. 22, the hardware experiments are conducted in a $4m \times 5m$ environment, with the motion capture system `OptiTrack` providing global positioning of the target and the robots. Three homogeneous ground robots `LIMO` have a rectangular shape of $0.2m \times 0.3m$, with the maximum pushing force of about $10N$. Each robot is equipped with a NVIDIA Jetson Nano that runs the `ROS1` system for communication and control. The centralized planning and control algorithm is implemented on a laptop with an Intel Core i7-1280P CPU, which runs the master node of `ROS1` and sends the velocity commands to all robots via wireless. Moreover, a L-shape object with a mass of $1.5Kg$ is adopted for the second scenario, while a rectangular object with a mass of $1Kg$ for the first scenario. Both objects have a ground friction coefficient of 0.5, and a side friction coefficient of 0.3. The obstacles are made of white cardboards at known positions. The main challenge for the second scenario lies in the non-convex shape of the target with a relatively larger mass. The object shape in the first scenario is simpler but with tighter geometric constraints such as narrow passages.

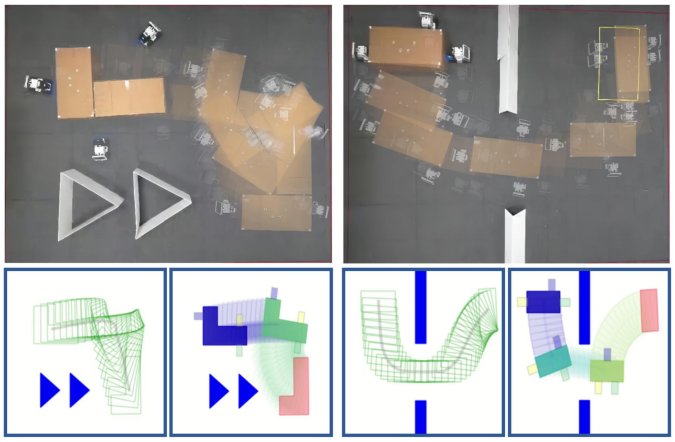

Fig. 22: Two scenarios of the hardware experiments (**Top**), the guiding path and the hybrid plan (**Bottom**).

2) Results: Overall, the experiment results are consistent with the simulations for both free and cluttered environments. Namely, via the proposed method, both tasks are completed successfully within $50s$ and $55s$, which require 2 and 3 modes, respectively. The average tracking errors are $0.023m$ and $0.052m$. Compared with simulations, the model uncertainties caused by communication delays and actuation noises or slipping are much more apparent in the hardware experiments. Thus, the replanning mechanism is necessary during online execution, as described in Sec. III-D3. Once the object deviates from the planned trajectory by more than $0.2m$, or remains static for $5s$, the KG-HS algorithm is re-executed to generate a new hybrid plan, which is then tracked by the NMPC controller. As shown in Fig. 23, the system in some cases can not recover from the large deviation from the reference trajectory in the current interaction mode. Thus, a replanning is triggered, after which another mode is adopted.

V. Conclusion

This work proposes a collaborative planar pushing strategy of polytopic objects with multiple robots in complex scenes. It utilizes a hybrid search algorithm to generate intermediate keyframes and a sparse optimization to generate sufficient pushing modes. It is shown to be scalable and effective for any polytopic object and an arbitrary number of robots in obstacle-cluttered scenes. The completeness and feasibility of our algorithm have been demonstrated both theoretically and experimentally. Extensive numerical simulation and hardware experiments validate its effectiveness for collaborative pushing tasks, with numerous extensions to more general scenarios.

There are some limitations of the proposed approach that are worth mentioning: (i) The quasi-static analyses in Sec. III-A1 are only valid for slow movements of the object and the robots. In high-speed scenarios, the object may experience large instantaneous accelerations, which could require the robotic fleet to react with larger forces and torques that may not be allowed in the current pushing mode. Although this can be partially mitigated by the multi-directional feasibility estimation proposed in (10) and the inertia term to NMPC in (22), a comprehensive solution for the high-speed scenarios remains part of our future work; (ii) The intrinsics of the object such as mass, center of gravity and friction coefficient are all assumed to be known for the evaluation of MG-SO in (10). Although slight deviations can be handled by the NMPC in (22) and the re-planning process as described in Sec. III-D3, large uncertainties might require other techniques, e.g., model-free and learning-based methods that adapt pushing strategies online. This is part of our ongoing work; (iii) The workspace is assumed to be feasible in the sense that a guiding path $\overline{\mathbf{S}}$ always exists for the given initial and goal states. Despite of being rather mild, it can be further relaxed, e.g., by allowing movable obstacles or by temporarily pulling out some robots to make room and re-join later, which is part of future work; (iv) Lastly, a centralized perception system is assumed that has a full knowledge of the system state and the environment. A decentralized method for perception and state estimation would further improve the applicability.

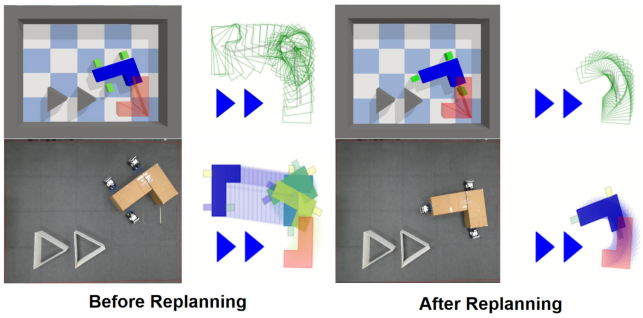

Fig. 23: The hybrid plan and the resulting trajectory before (**Left**) and after (**Right**) replanning.

Acknowledgement

This work was supported by the National Key Research and Development Program of China under grant 2023YFB4706500; the National Natural Science Foundation of China (NSFC) under grants 62203017, U2241214, T2121002; and the Fundamental Research Funds for the central universities.

References

[1] Frank Allgower, Rolf Findeisen, Zoltan K Nagy, et al. Nonlinear model predictive control: From theory to application. *Journal-Chinese Institute Of Chemical Engineers*, 35(3):299–316, 2004.

[2] Joel A E Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1): 1–36, 2019.

[3] Gino van den Bergen. A fast and robust gjk implementation for collision detection of convex objects. *Journal of graphics tools*, 4(2):7–25, 1999.

[4] Jianing Chen, Melvin Gauci, Wei Li, Andreas Kolling, and Roderich Groß. Occlusion-based cooperative transport with a swarm of miniature mobile robots. *IEEE Transactions on Robotics*, 31(2):307–321, 2015.

[5] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2019.

[6] Steven Diamond and Stephen Boyd. Cvxpy: A python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.

[7] Paula García, Pilar Caamaño, Richard J Duro, and Francisco Bellas. Scalable task assignment for heterogeneous multi-robot teams. *International journal of advanced robotic systems*, 10(2):105, 2013.

[8] Suresh Goyal, Andy Ruina, and Jim Papadopoulos. Limit surface and moment function descriptions of planar sliding. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 794–795, 1989.

[9] Meng Guo and Mathias Bürger. Geometric task networks: Learning efficient and explainable skill coordination for object manipulation. *IEEE Transactions on Robotics*, 38(3):1723–1734, 2022.

[10] Valentin N. Hartmann, Andreas Orthey, Danny Driess, Ozgur S. Oguz, and Marc Toussaint. Long-horizon multi-robot rearrangement planning for construction assembly. *Transactions on Robotics*, 2022.

[11] Francois R Hogan and Alberto Rodriguez. Reactive planar non-prehensile manipulation with hybrid model predictive control. *The International Journal of Robotics Research*, 39(7):755–773, 2020.

[12] Leslie Pack Kaelbling and Tomás Lozano-Pérez. Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation*, pages 1470–1477, 2011.

[13] Imin Kao, Kevin M Lynch, and Joel W Burdick. Contact modeling and manipulation. *Springer Handbook of Robotics*, pages 931–954, 2016.

[14] Beomjoon Kim, Zi Wang, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Learning to guide task and motion planning using score-space representation. *The International Journal of Robotics Research*, 38(7):793–812, 2019.

[15] C Ronald Kube. Task modelling in collective robotics. *Autonomous robots*, 4:53–72, 1997.

[16] Steven M LaValle. *Planning Algorithms*. Cambridge univ. press, 2006.

[17] An T. Le, Meng Guo, Niels van Duijkeren, Leonel Rozo, Robert Krug, Andras G. Kupcsik, and Mathias Bürger. Learning forceful manipulation skills from multi-modal human demonstrations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[18] Steven J Leon, Lisette G De Pillis, and Lisette G De Pillis. *Linear algebra with applications*. Pearson Prentice Hall Upper Saddle River, NJ, 2006.

[19] Kevin M Lynch, Hitoshi Maekawa, and Kazuo Tanie. Manipulation and active sensing by pushing using tactile feedback. In *IROS*, volume 1, pages 416–421, 1992.

[20] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009.

[21] Anthony Simeonov, Yilun Du, Beomjoon Kim, Francois Hogan, Joshua Tenenbaum, Pulkit Agrawal, and Alberto Rodriguez. A long horizon planning framework for manipulating rigid pointcloud objects. In *Conference on Robot Learning*, pages 1582–1601. PMLR, 2021.

[22] Zili Tang, Junfeng Chen, and Meng Guo. Combinatorial-hybrid optimization for multi-agent systems under collaborative tasks. In *IEEE Conference on Decision and Control (CDC)*, 2023.

[23] Tamás Terlaky. *Interior point methods of mathematical programming*, volume 5. Springer Science & Business Media, 2013.

[24] Marc Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *IJCAI*, pages 1930–1936, 2015.

[25] Marc A Toussaint, Kelsey Rebecca Allen, Kevin A Smith, and Joshua B Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. Robotics: Science and Systems Foundation, 2018.

[26] Elio Tuci, Muhanad HM Alkilabi, and Otar Akanyeti. Cooperative object transport in multi-robot systems: A review of the state-of-the-art. *Frontiers in Robotics and AI*, 5:59, 2018.

[27] Lovekesh Vig and Julie A Adams. Multi-robot coalition formation. *IEEE transactions on robotics*, 22(4):637–649, 2006.

[28] Ying Wang and Clarence W De Silva. Multi-robot box-pushing: Single-agent q-learning vs. team q-learning. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 3694–3699, 2006.

[29] Zi Wang, Caelan Reed Garrett, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Learning compositional models of robot skills for task and motion planning. *The International Journal of Robotics Research*, 40(6-7):866–894, 2021.

[30] Edmund Taylor Whittaker. *A treatise on the analytical dynamics of particles and rigid bodies*. CUP Archive, 1964.

[31] Yuchen Xiao, Joshua Hoffman, Tian Xia, and Christopher Amato. Learning multi-robot decentralized macro-action-based policies via a centralized q-net. In *IEEE International conference on robotics and automation (ICRA)*, pages 10695–10701, 2020.

[32] Teng Xue, Hakan Girgin, Teguh Santoso Lembono, and Sylvain Calinon. Guided optimal control for long-term non-prehensile planar manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4999–5005, 2023.

[33] Andy Zeng, Shuran Song, Stefan Welker, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4238–4245, 2018.