Robust and Safe Task-Driven Planning and Navigation for Heterogeneous Multi-Robot Teams with Uncertain Dynamics

Tianyang Pan[1], Christos K. Verginis[2] and Lydia E. Kavraki[1]

***Abstract*— Task and motion planning (TAMP) can enhance intelligent multi-robot coordination. TAMP becomes significantly more complicated in obstacle-cluttered environments and in the presence of robot dynamic uncertainties. We propose a control framework that solves the motion-planning problem for multi-robot teams with uncertain dynamics, addressing a key component of the TAMP pipeline. The principal part of the proposed algorithm constitutes a decentralized feedback control policy for tracking of reference paths taken by the robots while avoiding collision and adapting in real time to the underlying dynamic uncertainties. The proposed framework further leverages sampling-based motion planners to free the robots from local-minimum configurations. Extensive experimental results in complex, realistic environments illustrate the superior efficiency of the proposed approach, in terms of planning time and number of encountered local minima, with respect to state-of-the-art baseline methods.**

I. Introduction

Modern robotics applications require multiple heterogeneous robots to safely execute complex tasks in obstacle-cluttered environments and over long horizons. The robots must be able to reason about high-level tasks and derive and track successfully the respective low-level paths. The complexity of solving such high-dimensional continuous problems explodes with more robots and goals. At the same time, robots evolve subject to dynamics that often suffer from uncertainties and unknown exogenous disturbances that need to be addressed by the underlying algorithms. Task and Motion Planning (TAMP) [1] has become a popular paradigm as it interleaves both high-level reasoning and low-level motion planning to find feasible paths that reach the high-level goal efficiently, while feedback control is often used to accommodate the robot dynamics [2], [3].

This paper considers the problem of motion planning for a team of heterogeneous robots with uncertain 2nd-order dynamics in obstacle-cluttered environments. We consider that the robots have to navigate to certain locations of interest in the workspace, which can be dictated by a higher-level task-planning algorithm (see Fig. 1). Such multi-robot navigation is an indispensable step in all task and motion planning frameworks concerning efficient coordination of multi-robot systems. A straightforward way to solve this problem is to deploy a typical task and motion planner [4], [5] to find multiple steps of paths that achieve the high-level goal, and then execute the plans with standard feedback controllers. However, centralized motion planning typically cannot scale well to large numbers of robots [6], and standard planners cannot handle dynamic uncertainties [3]. Our previous work [6] alleviated these challenges by using potential field-based adaptive controllers, to account for dynamic uncertainties, augmented with sampling-based motion planners (SBMP) [7], [8] to free the robots from local-minima configurations. Although promising, this approach still cannot scale well to complex obstacle environments with multiple robots (as verified in the results of Sec. V-.3) due to the existence of a large number of local-minima configurations that require many expensive SBMP calls.

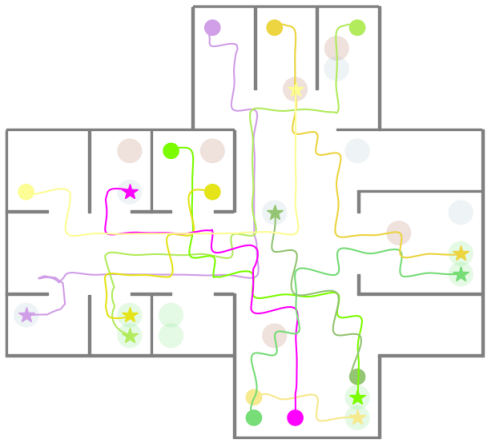

Fig. 1. Ten circular robots driving to their current goals (marked as stars with the same colors), following the proposed method. The purple, green, and yellow robots belong to three different types. The larger circles indicate locations of interest that have task-level constraints. The adjacent green locations must be visited by a yellow robot and a green robot at the same time. A robot must visit a blue location before it can visit a brown location. Every location must be visited at least once, resulting in a multi-step task plan, where the shown trajectories accomplish a single step. Scene adapted from the floor plan of the Duncan Hall at Rice University.

While SBMP is the bottleneck of the scalability of our prior approach [6], Multi-Agent Path-Finding (MAPF) algorithms achieve improved efficiency by searching over some graph representation of the workspace under the assumption of discretized time and space [9], [10]. This motivates us to develop a feedback controller that can track a multi-robot path while avoiding all types of collisions during online execution. The benefit is two-fold. First, it admits multi-robot paths that are only collision-free at a more coarse level of discretization, without requiring an SBMP to compute feasible paths in the high-dimensional continuous

[1]Tianyang Pan and Lydia E. Kavraki are with the Department of Computer Science, Rice University. Kavraki is also affiliated with the Ken Kennedy Institute at Rice University. They are supported in part by DOD W912HZ2320003 and NSF CCF-2336612. {tp36, kavraki}@rice.edu

[2]Christos K. Verginis is with the Division of Signals and Systems, Department of Electrical Engineering, Uppsala University. christos.verginis@angstrom.uu.se

space. Second, it enables a general multi-robot navigation framework that does not suffer from the limiting simplified assumptions of typical MAPF solvers [9], [11]. We will sketch out such a framework in Sec. IV.

The core contribution of this paper is the development of a multi-robot feedback control policy such that the robots simultaneously track given paths while avoiding collisions with each other and workspace obstacles. The policy (i) is decentralized, in the sense that each robot computes its own control input based on local information from the surrounding robots, (ii) adapts in real time to the uncertainties in the robot dynamics, and (iii) provably guarantees inter-robot and robot-obstacle collision avoidance. The potentially conflicting path-tracking and collision-avoidance objectives in the aforementioned policy might trap the robots in local-minimum configurations. Therefore, we augment the feedback control policy with an SBMP to free the robots from such configurations; the SBMP samples geometric paths in the free space of the robots, and a separate path-tracking control policy is used to track them. We conduct extensive experimental studies in simulated environments that illustrate the superiority of the proposed algorithm with respect to state-of-the-art methods. The experimental results illustrate that the proposed control policy outperforms the one based on the previous work [6]; that is, it encounters significantly fewer local-minimum configurations and hence results in lower computational complexity.

The rest of the paper is organized as follows. Sec. II provides an overview of works in the related literature. Sec. III describes the considered problem while Sec. IV illustrates the proposed methodology. Finally, Sec. V is devoted to experimental results and Sec. VI concludes the paper.

II. Related Work

Multi-robot planning: Different lines of research focus on multi-robot motion planning (MRMP). The geometric SBMP [7], [8], [12], [13] find geometric paths in the composite continuous space, but cannot scale well to a large number of robots [6]. MAPF algorithms can find paths for multiple robots with increased efficiency when the typical discretization assumption holds true [9], [10], [11].

Notably, one work proposes a robust execution framework for MAPF paths by post-processing the paths to enforce the dependencies being preserved during execution [14], but it does not consider compliant feedback controllers that can safely track the derived paths. Kinodynamic SBMP takes robot dynamics into consideration by sampling control inputs to forward simulate with the robot's dynamics model in order to find dynamically feasible trajectories [12], [15]. Such methods still suffer from obtaining inaccurate solutions when uncertainties exist in the dynamics model, because the simulation would deviate from the actual trajectories.

To solve long-horizon tasks, various TAMP frameworks integrate different types of task planners [16], [17] and motion planners based on the assumptions made [18]. Geometric SBMP is a typical choice in TAMP [1], [4], [5], [19], while some other approaches focus on extending MAPF to solve more complex high-level tasks [20], [21], but these works do not consider problems with uncertain dynamics.

In this work, we focus on developing a feedback controller that is capable of tracking a multi-robot trajectory computed by a multi-robot planning framework, potentially leveraging the above state-of-the-art planners, while avoiding collisions online under uncertain dynamics.

Feedback control: Feedback control is a popular methodology to tackle motion-planning problems, since it offers a closed-form policy for the control input of the system, accommodating simultaneously the robot dynamics. Early works focus on the concept of navigation functions [2] that are based on correct-by-construction artificial potential fields and guarantee convergence from almost all initial conditions. Similar approaches are developed in [22], whereas [23], [24] focus on Model Predictive Control frameworks. However, feedback-control policies usually assume (i) spherical and sparse workspaces, i.e., where the robots and the obstacles are approximated by spheres placed sufficiently far from each other, and (ii) simplified and perfectly known robot dynamics (aka differential constraints) such as single or double integrators. More complex shapes are considered in [25] and [26]. However, [25] does not establish convergence guarantees and [26] is limited to single-integrator dynamics. Our previous works [27], [28], [29] consider more complicated dynamics including uncertainties and disturbances but are limited to sequential, single-robot navigation.

III. Problem Formulation

Consider N mobile robots operating in a compact and bounded workspace $\mathcal{W} \subset \mathbb{R}^n$, with index set $\mathcal{N} := \{1, ..., N\}$. Each robot occupies a sphere of radius $r_i > 0$, denoted by $\mathcal{A}_i(x_i) \subset \mathbb{R}^n$, where $x_i \in \mathbb{R}^n$ is the respective geometric center, representing robot i's state. We consider the following $2nd$-order dynamics for robot $i \in \mathcal{N}$:

$$m_i \dot{v}_i = u_i + f_i(x_i, v_i, t), \tag{1}$$

where $v_i = \dot{x}_i$ is the ith robot's velocity, m_i is its mass, $f_i : \mathbb{R}^{2n} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ are functions modeling disturbances and model uncertainties. We note that both m_i and f_i are unknown, for $i \in \mathcal{N}$, and cannot be used in the control design. We impose the following assumption on $f_i(\cdot)$:

Assumption 1. *The functions $f_i(x_i, v_i, t)$ are locally Lipschitz in $(x_i, v_i) \in \mathbb{R}^{2n}$ for each fixed $t \geq 0$, continuous in $t \geq 0$ for each fixed $(x_i, v_i) \in \mathbb{R}^{2n}$, and satisfy the condition $\|f_i(x_i, v_i, t)\| \leq \alpha_i \|v_i\| + F_i$, for all $(x_i, v_i, t) \in \mathbb{R}^{2n} \times \mathbb{R}_{\geq 0}$, $i \in \mathcal{N}$, where α_i, F_i are positive constants.*

Assumption 1 is inspired by standard friction-like terms, which can be approximated by continuously differentiable velocity functions [27]. Further, Assumption 1 implies that $f_i(\cdot)$ are uniformly bounded with respect to x_i and t. We stress that the constants α_i and F_i are *unknown*.

Moreover, let there be M disjoint obstacles in the workspace $\mathcal{O}_k \subset \mathcal{W}, \forall k \in \mathcal{M} := \{1, ..., M\}$. For simplicity, we consider that the workspace boundary $\partial\mathcal{W}$ is included in the obstacle set $\{\mathcal{O}_1, \ldots, \mathcal{O}_M\}$.

Although this paper primarily focuses on the multi-robot motion-planning problem, we provide here a more general context where the robots have to execute a higher-level complex task specification over a set of locations of interest in the workspace. These locations are predefined configurations $c_i \in \mathcal{W} \backslash \cup_{k\in\mathcal{M}} \{\mathcal{O}_k\}$, $i \in \mathcal{C} := \{1, ..., C\}$ and are assumed to be sufficiently far from the obstacles as well as each other:

$$\|c_i - c_j\| > 2 * \max_{n\in\mathcal{N}} r_n + \nu, \forall i, j \in C, i \neq j \tag{2a}$$

$$\min_{m\in\mathcal{M}} \inf_{y\in\mathcal{O}_k} \|c_i - y\| > \max_{n\in N} r_n + \nu_o, \forall i \in C, k \in M \tag{2b}$$

for positive constants ν_o, ν.

The task specification can include complex symbolic constraints, such as ordering constraints and collaboration constraints (see Fig. 1). Planning Domain Specification Language (PDDL) [30] is a typical choice to express such complex tasks where each robot has different high-level capabilities and many off-the-shelf planners can be used to find a solution [16], [31]. We follow the definition of task planning in typical TAMP literature [4], [5], [19], which is defined over a finite set of states $\mathcal{S}$, a finite set of actions A allowing transitions between states, an initial state $s_{init} \in \mathcal{S}$, and a finite set of goal states $\mathcal{S}_{goal} \subseteq \mathcal{S}$. The solution is to derive a plan $T = (\langle a_1^0, a_2^0, ..., a_N^0\rangle, ..., \langle a_1^h, a_2^h, ..., a_N^h\rangle)$, that transitions from the initial s_0 to the goal state $s_h \in \mathcal{S}_{goal}$, where a_i^j denotes the action assigned to robot i in step j. With this formulation, we are interested in *synchronous* problems [19], where any robot i that finishes executing its action a_i^j must wait until every other robot also completes the execution of their actions in step j.

In this paper, we are only interested in *navigation* actions, i.e., actions that safely navigate the robots among the locations $\mathcal{C}$. Consequently, each step must be grounded to a set of feedback controllers for the robots that are assigned navigation actions. This grounding is natural because a navigation action expresses the start c_i^j and the immediate goal c_i^{j+1} of a robot i in step j. We note that not every robot is always assigned an action in all steps. An"idle" robotis also allowed to move to avoid other robots. Still, it must return to where it started to ensure the state s_{j+1} is correctly achieved after executing the controllers for other actions in step j. At each step, the composite action $\langle a_0^j, a_1^j, ..., a_N^j\rangle$ must be grounded to a set of collision-free trajectories $\boldsymbol{p}^j = (p_1^j, \dots, p_N^j) : [0, T_f] \to \mathcal{F}$, for some duration T_f. Formally, the problem we consider is the following:

Problem 1. *Given a set of goal states $\mathcal{S}_{goal}$ and a feasible guidance trajectory in the form of a task and motion plan $\mathbf{T} = (\langle(a_1^1, p_1^1), ..., (a_N^j, p_N^j)\rangle, ..., \langle(a_1^h, p_1^h), ..., (a_N^h, p_N^h)\rangle)$, design a multi-robot control policy to guarantee that the robots reach a goal state $s \in \mathcal{S}_{goal}$.*

IV. Method

This section presents the main results of the paper. We propose a novel decentralized feedback controller for simultaneous multi-robot path tracking while avoiding collisions and implicitly compensating for uncertain dynamics (Sec. IV-A). The conflicting path-tracking and collision-avoidance objectives of this controller might result in local-minima configurations. Hence, we apply a geometric sampling-based motion planner that frees the robots from such configurations by computing suitable paths in the robots' free space (Sec. IV-C). Such paths are then tracked by the robots using a funnel-based controller [6]. Our designed controller serves as a crucial component in a general multi-robot navigation framework (Sec. IV-D), where a guidance planner, consisting of a PDDL task planner and a MAPF solver, derives the guidance reference trajectories the controller must track in the form of a task and motion plan. This framework is agnostic to the choice of the planners being used.

A. Trajectory-Tracking-and-Collision-Avoidance Controller

We design a decentralized feedback control policy to navigate the robots to their goals. The policy integrates adaptive-control techniques and reciprocal barrier functions to account for collision avoidance and the uncertain robot dynamics. We consider that a higher-level MAPF algorithm has provided multi-robot time-varying reference paths $p_i(t)$, $i \in \mathcal{N}$, connecting the robots' initial and goal configurations.

Since the shapes of the obstacles might not have a closed-form expression describing them, we approximate obstacle k by L_k spheres, and define $\bar{\mathcal{L}} := \{1, \dots, L_1, L_1 + 1, \dots, \sum_{k\in\mathcal{M}} L_k\}$ as the respective index set for all the obstacles spheres[1]. Moreover, we denote by o_k and r_{o_k} the center and radius of the kth obstacle sphere. Note that the spheres might be intersecting. By defining $e_i := x_i - p_i(t) \in \mathbb{R}^n$, $\forall i \in \mathcal{N}$, $d_{ij} := \|x_i - x_j\|^2 - (r_i + r_j)^2 - \sigma$, $\forall i, j \in \mathcal{N}$, $i \neq j$, and distances to obstacle spheres as $d_{ik}^o := \|x_i - o_k\|^2 - (r_i + r_{o_k})^2 - \sigma_o$, $\forall k \in \bar{\mathcal{L}}$, the control design aims at guaranteeing $\lim_{t\to\infty} e_i(t) = 0$, and $d_{ij}(t) > 0$, $d_{ik}^o(t) > 0$, $\forall t \geq 0$, $i, j \in \mathcal{N}$, $i \neq j$, $k \in \bar{\mathcal{L}}$. The constants σ and σ_o are safety margins that will be used in the subsequent sections.

In order to reduce the number of potential local minima configurations, we wish the robots to be affected by other robots and the obstacles (to avoid collisions) only when they are sufficiently close to each other, reducing thus the configurations where the counteracting objectives (go-to-goal and collision avoidance) can cancel each other. Therefore, we design the smooth switch $\beta(\cdot, y) : [0, y] \to [0, \bar{\beta}]$, with

$$\beta(*, y) := \begin{cases} \vartheta(*), & 0 \leq * \leq y \\ \bar{\beta}, & y \leq * \end{cases},$$

for positive constants $y, \bar{\beta}$, and an appropriate polynomial $\vartheta(\cdot)$ that guarantees that $\beta(\cdot, y)$ is twice continuously differentiable. We define now $\beta_{ij} := \beta_{ij}(d_{ij}) := \beta(d_{ij}, \underline{s})$, $\forall i, j \in \mathcal{N}$, $i \neq j$, with $\underline{s}$ being a small positive constant representing the distance where the agents take each other into account for collision avoidance (e.g., a sensing radius - observe that β_{ij}'s derivative is zero when $d_{ij} > \underline{s}$). Given β_{ij}, we define then the barrier-like functions $b_{ij} := \beta_{ij}(d_{ij})^{-1}$ that blow up to infinity when $d_{ij} = 0$, $\forall i, j \in \mathcal{N}$, $i \neq j$. Similarly, we define the barrier-like functions $b_{ik}^o := (\beta_{ik}^o)^{-1}$, where $\beta_{ik}^o := \beta_{ik}^o(d_{ik}^o) := \beta(d_{ik}^o, \underline{s}_o)$, $\forall i \in \mathcal{N}, k \in \bar{\mathcal{L}}$, for a small

[1]The spherical approximation is dropped in the next section.

positive constant $\underline{s}_o$ respresenting the distances where the robots take the obstacles into account for collision avoidance.

We now define the free space of the robots as

$$\mathcal{F} \coloneqq \{x \in \mathbb{R}^{Nn} : d_{ij} > 0, d^o_{ik} > 0, i, j \in \mathcal{N}, i \neq j, k \in \bar{\mathcal{L}}\}$$

and a potential function for each robot as $\phi_i : \mathcal{F} \times \mathbb{R}^n$, with

$$\phi_i(x, e_i) \coloneqq \frac{k_{i,1}}{2}\|e_i\|^2 + \frac{k_2}{2}\sum_{j\in\mathcal{N}\setminus\{i\}} b_{ij} + \frac{k_{i,3}}{2}\sum_{k\in\bar{\mathcal{L}}} b^o_{ik}, \quad (3)$$

where $k_{i,1}$, k_2, $k_{i,3}$ are positive constant gains, for all $i \in \mathcal{N}$. We further define the aggregated function $\phi : \mathcal{F} \times \mathbb{R}^{Nn}$, with $\phi(x, e) \coloneqq \sum_{i\in\mathcal{N}} \phi_i(x, e)$, where $e \coloneqq [e_1^\top, \dots, e_N^\top]^T \in \mathbb{R}^{Nn}$. By computing the time derivative of each ϕ_i, we obtain

$$\begin{aligned}\dot{\phi}_i =& k_{i,1}e_i^\top(v_i - \dot{p}_i) + k_2\sum_{j\in\mathcal{N}\setminus\{i\}} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j)^\top(v_i - v_j)\\ &+ k_{i,3}\sum_{k\in\bar{\mathcal{L}}} \frac{\partial b^o_{ik}}{\partial d^o_{ik}}(x_i - o_k)^\top v_i.\end{aligned}$$

Since $d_{ij} = d_{ji}$ and hence $b_{ij} = b_{ji}$, for all $i, j \in \mathcal{N}$ with $i \neq j$, we conclude that $\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}\setminus\{i\}} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j)^\top(v_i - v_j) = 2\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}\setminus\{i\}} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j)^\top v_i$ leading to $\dot{\phi} = \sum_{i\in\mathcal{N}}\{\Delta\phi_i^\top v_i - e_i^\top \dot{p}_i\}$, with

$$\begin{aligned}\Delta\phi_i \coloneqq& k_{i,1}e_i + 2k_2\sum_{j\in\mathcal{N}\setminus\{i\}} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j)\\ &+ k_{i,3}\sum_{k\in\bar{\mathcal{L}}} \frac{\partial b^o_{ik}}{\partial d^o_{ik}}(x_i - o_k),\ i \in \mathcal{N}. \quad (4)\end{aligned}$$

We design next the feedback-control algorithm. Define first the reference velocity signals as

$$v_{r_i} \coloneqq -\Delta\phi_i, \quad i \in \mathcal{N}, \quad (5)$$

Note that $\Delta\phi_i$ uses only local information from robot i's neighbors, as dictated by the communication radius s_i. Further note that $v_{r_i} \to \infty$ in a collision, which is used by the control design to guarantee safety.

Next, we define the variables $\hat{m}_i$ and $\hat{\alpha}_i$ for each agent, which aim to compensate for the unknown masses m_i and terms related to α_i in Assumption 1. We design now the decentralized control law for each agent as

$$u_i = -\Delta\phi_i + \hat{m}_i\dot{v}_{r_i} - (k_{v_i} + \hat{\alpha}_i)e_{v_i}, \quad (6a)$$

where $e_{v_i} \coloneqq \dot{x}_i - v_{r_i}$, and k_{v_i} are positive constant gains, $\forall i \in \mathcal{N}$. We further define the adaptation laws

$$\dot{\hat{m}}_i = -k_{m_i}(e_{v_i}^\top\dot{v}_{r_i} + \hat{m}_i) \quad (6b)$$

$$\dot{\hat{\alpha}}_i = k_{\alpha_i}(\|e_{v_i}\|^2 - \hat{\alpha}_i), \quad (6c)$$

where k_{m_i} and k_{α_i} are positive constant gains, for all $i \in \mathcal{N}$. The correctness of (6) is given in the following theorem.

Theorem 1. *Consider the multi-robot system described by the dynamics* (1) *subject to Assumption 1. The control* (5)-(6) *guarantees that $d_{ij}(x_i(t), x_j(t)) > 0$ for all $i, j \in \mathcal{N}$, $i \neq j$, $d^o_{ik}(x_i(t)) > 0$, for all $i \in \mathcal{N}$, as well as the convergence of $\Delta\phi_i$ and e_{v_i} to a set around zero.*

Proof: First note that the boundedness of $\mathcal{W}$ and $p_i(t)$ implies that, for $x \in \mathcal{F}$, it holds that $\|e_i\| < \bar{E} \coloneqq \max_{i\in\mathcal{N}}\{\bar{p}_i\} + \sup\mathcal{W}$, where $\bar{p}_i$ is the upper bound of $p_i(t)$, $i \in \mathcal{N}$, and $\sup\mathcal{W}$ is finite. Next, we define the adaptation errors $\widetilde{m}_i \coloneqq \hat{m}_i - m_i$, $\widetilde{\alpha}_i \coloneqq \hat{\alpha}_i - (\alpha_i + \frac{\alpha_i^2}{2} + \frac{1}{2})$, for all $i \in \mathcal{N}$, as well as the stacked vectors $\hat{m} \coloneqq [\hat{m}_1, \dots, \hat{m}_N]^\top \in \mathbb{R}^N$, $\hat{\alpha} \coloneqq [\hat{\alpha}_1, \dots, \hat{\alpha}_N]^\top \in \mathbb{R}^N$, $e_v \coloneqq [e_{v_1}^\top, \dots, e_{v_N}^\top]^\top$. The choice of $\widetilde{\alpha}$ will be clarified later. Next, we define the overall state $\zeta \coloneqq [x^\top, e^\top, e_v^\top, \widetilde{m}^\top, \widetilde{\alpha}^\top]^\top$ and the set $\Omega \coloneqq \mathcal{F} \times \bar{E}^N \times \mathbb{R}^{Nn+2N}$. Note that, since the robots start at collision-free initial positions, it holds that $\zeta(0) \in \Omega$. Furthermore, the dynamics (1) and control policy (6) are locally Lipschitz in ζ over Ω and continuous in t over $\mathbb{R}_{\geq 0}$. Therefore, according to [32, Theorem 2.1.3], we conclude that a solution $\zeta(t)$ exists for all $t \in [0, \tau_{\max})$, for a positive constant $\tau_{\max}$ such that $\zeta(t) \in \Omega$, for $t \in [0, \tau_{\max})$. Our goal is to prove that $\tau_{\max} = \infty$.

Let now the Lyapunov function candidate

$$V = \phi(x, t) + \sum_{i\in\mathcal{N}}\left\{\frac{m_i}{2}\|e_{v_i}\|^2 + \frac{1}{2k_{m_i}}\widetilde{m}_i + \frac{1}{2k_{\alpha_i}}\widetilde{\alpha}_i\right\},$$

which is well-defined for $\zeta \in \Omega$ and hence for $t \in [0, \tau_{\max})$. Differentiating V and employing Assumption 1, $\dot{\phi} = \sum_{i\in\mathcal{N}}\{\Delta\phi_i^\top v_i - e_i^\top\dot{p}_i\}$, $v_i = e_{v_i} + v_{r_i}$, (5), (6), and $\|e_i\| \leq \bar{E}$, we obtain

$$\begin{aligned}\dot{V} \leq \sum_{i\in\mathcal{N}}\Big\{&\bar{E}\bar{v} - \|\Delta\phi_i\|^2 - (k_{v_i} + \hat{\alpha}_i)\|e_{v_i}\|^2 + \alpha_i\|e_{v_i}\|\|v_i\|\\ &+ \|e_{v_i}\|F_i + \widetilde{m}_i e_{v_i}^\top\dot{v}_{r_i} - \widetilde{m}_i(e_{v_i}^\top\dot{v}_{r_i} + \hat{m}_i) + \widetilde{\alpha}_i(\|e_{v_i}\|^2 - \hat{\alpha}_i)\Big\}\end{aligned}$$

for $t \in [0, \tau_{\max})$, where $\bar{v}$ is the upper bound of $\dot{p}_i$. By completing the squares and using $v_i = e_{v_i} + \Delta\phi_i$, we obtain $\alpha_i\|e_{v_i}\|\|v_i\| \leq \alpha_i\|e_{v_i}\|^2 + \frac{\alpha_i^2}{2}\|e_{v_i}\|^2 + \frac{1}{2}\|\Delta\phi_i\|^2$ and $\|e_{v_i}\|F_i \leq \frac{1}{2}\|e_{v_i}\|^2 + \frac{1}{2}F_i^2$. Similarly, it is easy to obtain $-\widetilde{m}_i\hat{m}_i \leq -\frac{1}{2}\widetilde{m}_i^2 + \frac{1}{2}m_i^2$ and $-\widetilde{\alpha}_i\hat{\alpha}_i \leq -\frac{1}{2}\widetilde{\alpha}_i^2 + \frac{1}{2}\alpha_i^2$, for all $i \in \mathcal{N}$. By using the aforementioned relations and in view of $\widetilde{\alpha}_i = \hat{\alpha}_i - (\alpha_i + \frac{\alpha_i^2}{2} + \frac{1}{2})$, $i \in \mathcal{N}$, $\dot{V}$ becomes

$$\dot{V} \leq -\frac{1}{2}\|\Delta\phi\|^2 - \underline{k}_v\|e_v\|^2 - \frac{1}{2}\|\widetilde{m}\|^2 - \frac{1}{2}\|\widetilde{\alpha}\|^2 + \bar{F}$$

where $\underline{k}_v \coloneqq \min_{i\in\mathcal{N}}\{k_{v_1}, \dots, k_{v_N}\}$, $\Delta\phi \coloneqq [\Delta\phi_1^\top, \dots, \Delta\phi_N^\top]^\top$, and $\bar{F} \coloneqq N\bar{E}\bar{v} + \frac{1}{2}\sum_{i\in\mathcal{N}}\{F_i^2 + m_i^2 + \alpha_i^2\}$. Therefore, by employing [33, Theorem 4.18], we conclude that $\Delta\phi(t)$, $e_v(t)$, $\widetilde{m}(t)$, $\widetilde{\alpha}(t)$ are upper bounded and converge to a set around zero, whose value is proportional to $\bar{F}$ and inversely proportional to k_{v_i}, $i \in \mathcal{N}$. In view of (4), the boundedness of $\Delta\phi$ implies that the robots avoid collisions with each other and the workspace obstacles, i.e., there exist positive constants $\underline{d}_{ij}$ and $\underline{d}^o_{ik}$ such that $d_{ij}(x_i(t), x_j(t)) \geq \underline{d}_{ij}$, for all $i, j \in \mathcal{N}$ and $d^o_{ik}(x_i(t) \geq \underline{d}^o_{ik}$, for all $i \in \mathcal{N}$, and $t \in [0, t_{\max})$. Furthermore, the boundedness of e_v, $\widetilde{m}$, $\widetilde{\alpha}$ implies the boundedness of $\hat{m}_i(t)$ and $\hat{\alpha}_i(t)$, for all $i \in \mathcal{N}$, and $t \in [0, t_{\max})$. By differentiating (5), we further conclude the boundedness of $\dot{v}_{r_i}(t)$ and hence of the signals $u_i(t)$, $\dot{\hat{m}}_i$, $\dot{\hat{\alpha}}_i$, for all $i \in \mathcal{N}$, and $t \in [0, t_{\max})$.

Finally, the aforementioned analysis implies that $\zeta(t) \in \Omega'$ for $t \in [0, t_{\max})$, where Ω' is a compact subset of Ω. Hence, by invoking [32, Theorem 2.1.4], we conclude that $t_{\max} = \infty$ and the conclusion of the proof. □

Theorem 1 guarantees the convergence of the multi-robot system to a set around a configuration that satisfies $\Delta\phi_i = 0$, for all $i \in \mathcal{N}$, i.e., the set $\mathbb{S} = \{x \in \mathcal{F} : \Delta\phi_i = 0, \forall i \in \mathcal{N}\}$. In particular, (4) implies that such configurations $x^* = [(x_1^*)^\top, \ldots, (x_N^*)^\top]^\top \in \mathbb{S}$ satisfy

$$k_{i,1} e_i^* = -2k_2 \sum_{j \in \mathcal{N}\setminus\{i\}} \frac{\partial b_{ij}}{\partial d_{ij}} (x_i^* - x_j) - k_{i,3} \sum_{k \in \bar{\mathcal{L}}} \frac{\partial b_{ik}^o}{\partial d_{ik}^o} (x_i^* - o_k) \tag{7}$$

with $e_i^* = x_i^* - p_i(t)$, for all $i \in \mathcal{N}$. Note that, since the robot goals l_i are sufficiently far from each other as per (2), we can choose sufficiently small constants $\underline{s}$, $\underline{s}_o$ in the definition of the collision functions β_{ij} and β_{ik}^o, respectively, to guarantee $\frac{\partial b_{ij}}{\partial d_{ij}}|_{(x_i,x_j)=(l_i,l_j)} = \frac{\partial b_{ik}^o}{\partial d_{ik}^o}|_{x_i=l_i} = 0$ for all $i, j \in \mathcal{N}$ with $i \neq j$, $k \in \bar{\mathcal{L}}$, i.e., there's no inter-robot or robot-obstacle influence at the goal configurations. Therefore, the goal configurations l_i belong to the set $\mathbb{S}$, which implies that the robots can safely converge to an area around their goals. Nevertheless, the set $\mathbb{S}$ contains local-minimum configurations, i.e., configurations $x_i^* \neq l_i$ that satisfy (7). In order to escape such undesired configurations, we use the respective scheme of our previous work [6], which we present here for completeness. The scheme consists of a (i) controller that guarantees tracking of a geometric path within predefined bounds and (ii) a sampling-based motion planner that derives such a path. We briefly describe these procedures in the next subsections.

B. Path Tracking with Predefined Bounds

We present here a control policy that guarantees tracking of a collision-free path x_{d} within predefined bounds, using the Prescribed Performance Control (PPC) methodology [34]. This path is the output of a geometric sampling-based motion planner described in the next section, which is called by the robots to escape from the aforementioned local minima configurations. Let the path be endowed with a time interval, resulting in the time trajectory $x_{\mathrm{d}}(t)$, and let x_i^*, $\forall i \in \mathcal{N}$ denote the state of the robots at this configuration. Without loss of generality, we assume that the domain of x_{d} is $[0, t_f]$ for a designer-specified positive constant t_f. We next fix the index i since the control design concerns one robot. Moreover, we consider that $x_{\mathrm{d}}(0) = x_i^*(0)$, which can be guaranteed by next section's sampling-based planner.

PPC achieves evolution of an error state in prescribed bounds, regardless of the unknown dynamic terms m_i and $f_i(\cdot)$ of (1). More specifically, PPC aims at guaranteeing $|e_\ell(t)| < \rho$, $\forall \ell \in \{1, \ldots, n\}$, $t \geq 0$, where e_ℓ is the ℓth component of the error metric $e \coloneqq x_i - x_{\mathrm{d}}(t)$, and $\rho > 0$ is a *prescribed* positive constant[2] to bound $e(t)$. This ρ is used in the subsequent section to define an extended free space for the motion planner, since it dictates how close the trajectory of the robot can evolve to x_{d}.

The safety guarantees of Theorem 1 imply that, at a local-minimum configuration x^*, the robots will be far enough from each other the obstacles, i.e.,

$$\inf_{y \in \mathcal{A}_j(x_j^*)} \|x_i^* - y\| > \widetilde{\sigma}, \forall i, j \in \mathcal{N}, i \neq j \tag{8a}$$

$$\inf_{y \in \mathcal{O}_k} \|x_i^* - y\| > \widetilde{\sigma}_o, \forall i \in \mathcal{N}, k \in \mathcal{M}, \tag{8b}$$

for some positive constants $\widetilde{\sigma}$, $\widetilde{\sigma}_o$. In the motion planner of the next subsection, $x_{\mathrm{d}}(t)$ is chosen such that if robot i tracks it sufficiently close (as defined by $\min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$), it will not collide with other robots or obstacles, i.e.,

$$|e_\ell| \leq \min\{\widetilde{\sigma}, \widetilde{\sigma}_o\} \Rightarrow \mathcal{A}_i(x_i(t)) \cap \mathcal{A}_j(x_j^*) = \mathcal{A}_i(x_i(t)) \cap \mathcal{O}_k = \emptyset, \tag{9}$$

$\forall \ell \in \{1, \ldots, n\}$, $j \in \mathcal{N}\setminus\{i\}, k \in \mathcal{M}$ and $t \in [0, t_f]$. In particular, by choosing the constant ρ such that $\rho < \min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$, we guarantee avoidance of collisions between robot i and other robots/obstacles since the path-tracking control policy ensures $-\rho < e_\ell(t) < \rho$, $\forall i \in [0, t_f]$, $\ell \in \{1, \ldots, n\}$. We describe now the PPC control policy and refer the reader to [34] for the theoretical analysis of the establishment of the aforementioned inequalities.

PPC relies on the transformation $\mathbb{T}(*) = \ln\left(\frac{1+*}{1-*}\right)$. In particular, we introduce the transformed error $\varepsilon = [\varepsilon_1, \ldots, \varepsilon]^\top$, with $\varepsilon_\ell = \mathbb{T}\left(\rho^{-1} e_\ell\right)$, $\ell \in \{1, \ldots, n\}$. The control policy aims then to retain the boundedness of $\varepsilon(t)$, $t \in [0, t_f]$, which implies that $-\rho < e_\ell(t) < \rho$, for all $t \in [0, t_f]$ and $\ell \in \{1, \ldots, n\}$. We further define $J_T(*) = \frac{\partial \mathbb{T}(*)}{\partial *}$ as the gradient of $\mathbb{T}$. The PPC policy is now defined as follows:

Step I: Design the reference velocity $v_r = -k\rho^{-1} r_\varepsilon \varepsilon$, where $r_\varepsilon = \mathrm{diag}\left\{\left[J_T(\rho^{-1} e_\ell)\right]_{\ell \in \{1,\ldots,n\}}\right\}$ and k is a positive constant gain.

Step II: Define the velocity error $e_v = [e_{v_1}, \ldots, e_{v_n}]^\top = \dot{x}_i - v_r$ and let a constant ρ_v satisfying $\rho_v > \|e_v(0)\|$.

Step III: Design the controller $u_i = -k_v \rho_v^{-1} r_{\varepsilon_v} \varepsilon_v$, where $\varepsilon_v = [\varepsilon_{v_1}, \ldots, \varepsilon_{v_n}]^\top$, with $\varepsilon_{v_i} = \mathbb{T}(\rho_v^{-1} e_{v_i})$, $\ell \in \{1, \ldots, n\}$, $r_{\varepsilon_v} = \mathrm{diag}\left\{\left[J_T(\rho_v^{-1} e_{v_\ell})\right]_{\ell \in \{1,\ldots,n\}}\right\}$, and k_v is a positive constant gain.

C. Application of Motion planner

We execute the decentralized feedback control policy of Sec. IV-A to track the reference paths until the multi-robot system is trapped in a local minimum. We can detect the local minimum by checking if the robots all have a velocity close to zero (below a certain threshold) while not every robot has arrived at its goal configuration. Similar to our previous work [6], we select a subset of the robots that haven't arrived to perform sampling-based motion planning. The motion planner finds a path with a clearance of at least $\min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$ from the obstacles and the other robots, to ensure the PPC path-tracking controller tracks it without collisions. Different heuristics can be used to select the robots for motion planning. We prioritize choosing a single

[2]The original PPC methodology actually considers time-varying $\rho(t)$, but a constant ρ suffices in our case.

"unarrived" robot for planning efficiency and incrementally add arrived robots that might be blocking the passages and creating local minima, which was proven to be efficient in [6]. The probabilistic completeness guarantee of SBMP ensures escaping from the local minima, and any typical SBMP, such as RRT-Connect [13], can be applied.

D. A General Multi-robot Navigation Framework

The proposed algorithm presented in Sec. IV-A and IV-C can be used in a general task and motion planning framework, as presented in the sequel. Such a framework consists of two layers: a guidance planner and the designed control policy. The guidance planner follows typical TAMP paradigm [1], [4], [5], except that we use a MAPF solver instead of the SBMP. The input to the guidance planner is the task planning domain, the motion planning domain, and a set of goal states $\mathcal{S}_{goal}$ for the given task, encoded in PDDL. We use a task planner to reason over the high-level constraints and compute a task plan T, minimizing some heuristic costs that approximate the path lengths between locations.

Our designed multi-robot trajectory-tracking control policy guarantees real-time collision avoidance, allowing us hence to derive geometric paths that are only collision-free at a more coarse level of discretization of time and space. This makes a MAPF solver [10] suitable because we can leverage its efficiency while not suffering from its limitations of simplified assumptions (e.g., point robot, discrete time, and discrete space) of real-world problems. Most state-of-the-art MAPF solvers can be applied. The MAPF solver computes paths on a discrete graph, and a popular way to obtain such a graph is to apply a grid discretization on the given environment [11]. Since this paper does not focus on how to obtain such a graph, we assume that we can simply apply a grid discretization that preserves the connectivity of the environment, and guarantees the locations of interest to be in different grids (i.e., the occupancy of a grid depends on if part of a spherical obstacle resides in it, and the radius of the robot is guaranteed to be less than the grid size), thus not violating the typical assumptions of the MAPF solvers, such as the vertex conflicts. We refer the readers to the typical MAPF literature for detailed definitions [10], [11].

In each step, the query to the MAPF solver is constructed as follows. If a robot is assigned a navigation action, we add the start-goal pair into the MAPF query. Otherwise, we set the goal of the robot to be the same as its start location (i.e., $c_g = c_s$), to force driving it back to its current location in case it needs to move for coordination; this helps resolve conflicts when c_s blocks the path for another robot. The MAPF solver searches for a solution in the discrete graph. It either returns geometric paths that are conflict-free at the discrete resolution or reports failure, indicating an alternative task plan is needed. Then, we perform time parametrization over the paths to obtain a multi-robot reference trajectory as the output of the planner.

The guidance trajectory $\mathbf{T}$ is sent to the controller we designed in Sec. IV-A. For each step $\langle (a_1^j, p_1^j), ..., (a_N^j, p_N^j) \rangle$ in $\mathbf{T}$, we execute the controller to track trajectories $p_1^j, ..., p_N^j$ simultaneously until a local minima configuration is detected. We then invoke SBMP followed by a PPC path tracking controller as described in Sec. IV-C repeatedly until all the robots arrive at their current goal locations.

V. Experiments

1) Baselines: We present a comparison of our proposed framework with two baselines. For all baselines and our proposed method, we use the same open-source temporal planner, OPTIC [16]. Other temporal planners can also be applied. For the MAPF solver [10] used by our method, we apply a grid discretization to the environments with grid size being $\max_{i \in \mathcal{N}} r_i$, i.e., the maximum robot radius.

TPMP: This baseline is directly adapted from a typical TAMP framework [4] to the multi-robot navigation problem considered in this paper. It uses a task planner to assign robots to locations across multiple steps, and leverages a centralized SBMP, RRT-Connect [13], for motion planning. Then, we use the PPC controller to track the paths, as they are guaranteed to be collision-free by the clearance.

TPPF: This is a direct extension of our previous work [6] to the complex tasks considered in this paper. The task planner computes a task plan, and for each step in the plan, the start and goal locations of the involved robots are directly sent to the control policy, which is a potential field-based controller augmented by sampling-based motion planners. It keeps switching between the potential field controller and tracking paths computed by SBMP upon local minima detection. More details can be found in the previous paper [6].

2) Benchmarks: We test the methods on five scenes as shown in Fig. 2; scene 1 and 2 are the same used in previous work [6], scene 3 is a new scene created with several narrow passages to introduce difficulties, and the other scenes are adapted from real-world buildings (e.g., scene 4 from the Duncan Hall in Rice University and scene 5 from the Ångström Laboratory in Uppsala University). In the navigation domain, we have three types of robots (shown in green, yellow, and purple) and three types of locations (shown in blue, brown, and green), requiring the ordering, collaboration, and capability constraints to be satisfied: a robot (with any color) must visit a blue location before visiting a brown location, while a yellow and a green robot must simultaneously visit a pair of adjacent green locations. We consider robots of $m_i = 1$ and $r_i = 2$, $i \in \mathcal{N}$. The functions $f_i(\cdot)$ are given by $f_i = C_{i,1}^f v_i + C_{i,2}^f \tanh(v_i) + A_i^f [\sin(\omega_{i,1}^f t + \phi_{i,1}^f), \cos(\omega_{i,2}^f t + \phi_{i,2}^f)]^\top$, where $C_{i,1}^f$, $C_{i,2}^f$, $A_{i,1}^f$, $A_{i,2}^f$ are diagonal matrices with values randomly chosen in $(-1, 1)$. Similarly, $\omega_{i,1}^f$, $\omega_{i,2}^f$, $\phi_{i,1}^f$, $\phi_{i,2}^f$ are parameters randomly chosen in $(-1, 1)$.

3) Results: Each method is run 20 times for each scene with an increasing number of robots (5, 10, 15, 20). Similar to the previous work [6], we are interested in comparing the average total motion planning time (see Fig. 2), because the feedback controllers are run online in real time. For baselines, we count the total time spent in SBMP, and for the proposed method, we count both the SBMP time and

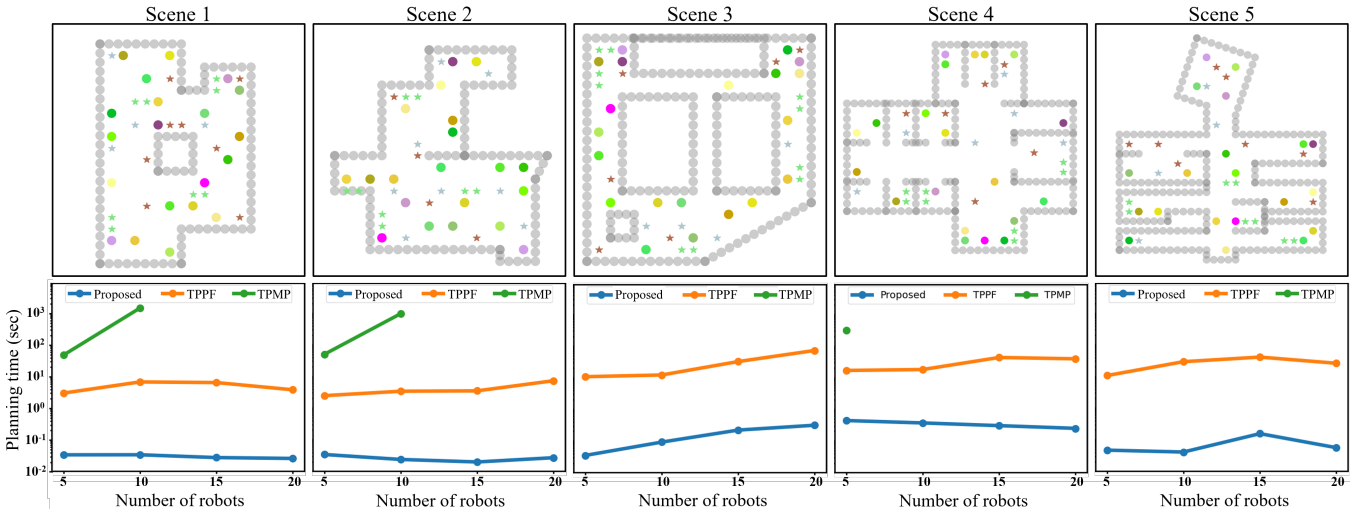

Fig. 2. **Top:** The five scenes with the initial positions of 20 robots as colored circles and the 20 goals as stars. Yellow, green, and purple circles represent three different types of robots. Blue, brown, and green stars represent three different types of locations. We use (gray) circular obstacles to approximate the obstacles. **Bottom:** the average motion planning time (20 runs each scenario) of the methods. An omitted line shows that no runs succeeded.

TABLE I

AVERAGE NUMBER OF LOCAL MINIMA ENCOUNTERED

# robot	5		10		15		20	
method	Proposed	TPPF	Proposed	TPPF	Proposed	TPPF	Proposed	TPPF
scene 1	0	3.4	0	11.6	0	8.6	0	3.05
scene 2	0	4.2	0	4.25	0	3.4	0	6.85
scene 3	0	7.75	0	7.8	0.05	15.0	0	15.8
scene 4	1.00	15.15	0.85	15.65	0.70	17.2	0.55	15.85
scene 5	0	20.42	0	22.5	0.05	18.35	0	18.3

the MAPF time. The TPMP baseline performs the worst. It can scale to 10 robots in the easier scene 1 and 2, and to 5 robots in scene 4, but it takes more than 100 seconds in motion planning, which suggests that directly applying typical TAMP frameworks designed for single-robot problems [4], [5] cannot scale well in multi-mobile-robot navigation tasks. The TPPF baseline performs much better than TPMP, but it still takes up to more than 50 seconds in motion planning in complex scenes, because unlike the proposed method which leverages some guidance paths to drive the robots, driving the robots to their current goals in a greedy way is likely to create local minima in cluttered settings with uncertainties, as already observed in our previous work [6].

Our proposed method succeeds in all scenarios, with an average total motion planning time of less than a second. It further vastly outperforms TPPF because the proposed controller can follow a trajectory while avoiding collisions, which in turn allows us to compute paths that are collision-free on a more coarse discretization of the time and space, enabling us to leverage efficient discrete space planners.

Even the proposed controller can get stuck in local minima due to uncertainties, we empirically found that our framework encounters much fewer local minima than the TPPF method. Table I shows the average number of times when the system is trapped in a local minimum.

Our method encounters the most number of local minima in scene 4, where we encounter 1.0 local minima on average in 5-robot cases, probably due to locations of interest being adjacent to cluttered narrow doors, while the TPPF baseline encounters local minima 15.15 times. In scene 4, our method encounters fewer local minima with more robots because oftentimes some robots have to travel through many narrow passages under uncertain dynamics, while with more robots, the task planner can assign locations to hopefully closer robots. In most other scenarios, our method does not encounter any local minima. Note that in the experiments, we have a fixed number of locations for each scenario, so with an increasing number of agents available to the planner, the number of steps in the task plan decreases. This explains why TPPF sometimes may encounter fewer local minima with even a larger number (15, 20) of robots (e.g., scene 5). Each local minimum, however, is usually more difficult to resolve in those cases, because it usually has more robots stuck. This is also reflected in a much higher planning time shown in Fig. 2. Generally, in complex scenarios, the robots are cluttered around narrow passages, making the controller used in TPPF often trapped, while our proposed method leverages guidance trajectories that attempt to resolve many such conflicts even before executing the controller.

Note that since OPTIC is an anytime planner that keeps improving task plans, we set the timeout for each call to be 30 seconds and use the last plan generated. The task planning time is not reported for comparison because we only need to call OPTIC once for each scenario, and the task planning time becomes the same for all the methods.

VI. CONCLUSION

We propose a decentralized feedback controller that can track given guidance trajectories and guarantees collision avoidance among the robots and between robots and obstacles, under 2nd-order uncertain dynamics. The controller is enhanced by an SBMP to free the controller from local minima. The proposed controller can serve as a crucial component in an efficient multi-robot coordination framework, composed of a guidance planner and the proposed controller.

Such a framework is necessary to solve complex tasks in the real world that require both planning for challenging task-level constraints and execution under high-order uncertain dynamics. The framework we proposed is general and agnostic to the planners being used. The application of the proposed controller allows the guidance planner to focus on resolving conflicts (collisions) at a more coarse level, leading to the computational efficiency demonstrated in the experimental results. Our approach also encounters fewer local minima compared to our previous work, resulting in much less planning time required to finish each task. This work opens up promising future directions, such as exploring more meaningful connections between the controller and the task and motion planner, decentralizing planning and sensing (potentially leading to asynchronous multi-robot problems), and reasoning over partially observable environments.

REFERENCES

[1] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez, "Integrated task and motion planning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, no. 1, pp. 265–293, 2021.

[2] D. V. Dimarogonas and K. J. Kyriakopoulos, "A feedback control scheme for multiple independent dynamic non-point agents," *International Journal of Control*, vol. 79, no. 12, pp. 1613–1623, 2006.

[3] C. K. Verginis, D. V. Dimarogonas, and L. E. Kavraki, "Kdf: Kinodynamic motion planning via geometric sampling-based algorithms and funnel control," *IEEE Transactions on robotics*, vol. 39, no. 2, pp. 978–997, 2022.

[4] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1134–1151, 2018. [Online]. Available: https://doi.org/10.1177/0278364918761570

[5] S.-Y. Lo, S. Zhang, and P. Stone, "The petlon algorithm to plan efficiently for task-level-optimal navigation," *The Journal of Artificial Intelligence Research (JAIR)*, vol. 67, October 2020. [Online]. Available: http://www.cs.utexas.edu/users/ai-lab?JAIR20-yunl

[6] T. Pan, C. K. Verginis, A. M. Wells, L. E. Kavraki, and D. V. Dimarogonas, "Augmenting control policies with motion planning for robust and safe multi-robot navigation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 6975–6981.

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

[8] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete rrt for exploration of implicit roadmaps in multi-robot motion planning," *The International Journal of Robotics Research*, vol. 35, no. 5, pp. 501–513, 2016. [Online]. Available: https://doi.org/10.1177/0278364915615688

[9] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artificial Intelligence*, vol. 219, pp. 40–66, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370214001386

[10] J. Li, D. Harabor, P. J. Stuckey, H. Ma, G. Gange, and S. Koenig, "Pairwise symmetry reasoning for multi-agent path finding search," *Artificial Intelligence*, vol. 301, p. 103574, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370221001259

[11] R. Stern, N. R. Sturtevant, A. Felner, S. Koenig, H. Ma, T. T. Walker, J. Li, D. Atzmon, L. Cohen, T. K. S. Kumar, E. Boyarski, and R. Barták, "Multi-agent pathfinding: Definitions, variants, and benchmarks," in *Proceedings of the International Symposium on Combinatorial Search (SoCS)*, 2019, pp. 151–159.

[12] S. M. LaValle, *Planning Algorithms*. USA: Cambridge University Press, 2006.

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

[14] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Persistent and robust execution of mapf schedules in warehouses," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1125–1131, 2019.

[15] D. Le and E. Plaku, "Multi-robot motion planning with unlabeled goals for mobile robots with differential constraints," *IEEE International Conference on Robotics and Automation*, pp. 7950–7956, 2021.

[16] J. Benton, A. Coles, and A. Coles, "Temporal planning with preferences and time-dependent continuous costs," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 22, no. 1, pp. 2–10, May 2012. [Online]. Available: https://ojs.aaai.org/index.php/ICAPS/article/view/13509

[17] A. Messing, G. Neville, S. Chernova, S. Hutchinson, and H. Ravichandar, "Grstaps: Graphically recursive simultaneous task allocation, planning, and scheduling," *The International Journal of Robotics Research*, vol. 41, no. 2, pp. 232–256, 2022.

[18] L. Antonyshyn, J. Silveira, S. Givigi, and J. Marshall, "Multiple mobile robot task and motion planning: A survey," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1–35, 2023.

[19] T. Pan, A. M. Wells, R. Shome, and L. E. Kavraki, "A general task and motion planning framework for multiple manipulators," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 3168–3174.

[20] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Conflict-based search with optimal task assignment," in *International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '18, 2018, p. 757–765.

[21] Z. Ren, S. Rathinam, and H. Choset, "Cbss: A new approach for multiagent combinatorial path finding," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2669–2683, 2023.

[22] D. Panagou, "A distributed feedback motion planning protocol for multiple unicycle agents of different classes," *IEEE Transactions on Automatic Control*, vol. 62, no. 3, pp. 1178–1193, 2016.

[23] A. Filotheou, A. Nikou, and D. V. Dimarogonas, "Robust decentralised navigation of multi-agent systems with collision avoidance and connectivity maintenance using model predictive controllers," *International Journal of Control*, vol. 93, no. 6, pp. 1470–1484, 2020.

[24] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, 2020.

[25] C. K. Verginis and D. V. Dimarogonas, "Closed-form barrier functions for multi-agent ellipsoidal systems with uncertain lagrangian dynamics," *IEEE Control Systems Letters*, vol. 3, no. 3, pp. 727–732, 2019.

[26] S. G. Loizou, "The multi-agent navigation transformation: tuning-free multi-robot navigation." *Robotics: Science and Systems*, vol. 6, pp. 1516–1523, 2014.

[27] C. K. Verginis and D. V. Dimarogonas, "Adaptive robot navigation with collision avoidance subject to 2nd-order uncertain dynamics," *Automatica*, vol. 123, p. 109303, 2021.

[28] C. K. Verginis, Y. Kantaros, and D. V. Dimarogonas, "Planning and control of multi-robot-object systems under temporal logic tasks and uncertain dynamics," *Robotics and Autonomous Systems*, p. 104646, 2024.

[29] C. K. Verginis and D. V. Dimarogonas, "Adaptive leader-follower coordination of lagrangian multi-agent systems under transient constraints," *IEEE Conference on Decision and Control*, pp. 3833–3838, 2019.

[30] M. Ghallab, C. Knoblock, D. Wilkins, A. Barrett, D. Christianson, M. Friedman, C. Kwok, K. Golden, S. Penberthy, D. Smith, Y. Sun, and D. Weld, "Pddl - the planning domain definition language," 08 1998.

[31] M. Helmert, "The fast downward planning system," *J. Artif. Int. Res.*, vol. 26, no. 1, p. 191–246, jul 2006.

[32] A. Bressan and B. Piccoli, *Introduction to the mathematical theory of control*. American institute of mathematical sciences Springfield, 2007, vol. 1.

[33] H. K. Khalil, *Nonlinear systems*. Prentice Hall, 2002.

[34] C. P. Bechlioulis and G. A. Rovithakis, "A low-complexity global approximation-free control scheme with prescribed performance for unknown pure feedback systems," *Automatica*, vol. 50, no. 4, pp. 1217–1226, 2014.