2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

# Augmenting Control Policies with Motion Planning for Robust and Safe Multi-robot Navigation

Tianyang Pan[1], Christos K. Verginis[2], Andrew M. Wells[1], Lydia E. Kavraki[1] and Dimos V. Dimarogonas[2]

***Abstract*— This work proposes a novel method of incorporating calls to a motion planner inside a potential field control policy for safe multi-robot navigation with uncertain dynamics. The proposed framework can handle more general scenes than the control policy and has low computational costs. Our work is robust to uncertain dynamics and quickly finds high-quality paths in scenarios generated from real-world floor plans. In the proposed approach, we attempt to follow the control policy as much as possible, and use calls to the motion planner to escape local minima. Trajectories returned from the motion planner are followed using a path-following controller guaranteeing robustness. We demonstrate the utility of our approach with experiments based on floor plans gathered from real buildings.**

## I. Introduction

The availability of new hardware and advent of new applications for multi-robot systems (e.g., monitoring) has triggered significant research efforts in planning for such systems. This paper considers the safe navigation problem for realistic robots: A team of robots with 2nd-order dynamics, including uncertain terms, needs to navigate from predefined start to goal positions while avoiding collisions in an obstacle-cluttered environment. Current approaches for this type of problem can be divided into feedback-based control methods and motion planning methods. The advantages and disadvantages of these approaches are detailed in the next paragraphs. Aiming to integrate their advantages, this paper draws from both methodologies and proposes a framework that provides explicit policies for the robots. The policies are robust against the dynamic uncertainties and suitable for real-time execution.

Feedback-based control approaches provide *explicit* closed-form feedback-based expressions for the inputs of the robots, usually derived by evaluating vector fields of artificial potential functions [2], [3], [4], [5], [6], [7], [8]. A common approach uses so-called navigation functions [7], [6], originally introduced in [9], which constitute correct-by-construction artificial potential field that guarantee convergence from *almost* all initial conditions (local minima configurations, where the gradient of the potential field is zero, but where the potential field is not at it's minimum, have an attraction set of measure zero). Other types of potential field and barrier functions are developed in [2] and [3], respectively. Such expressions provide efficient, low complexity solutions, since no discrete algorithms are required, and simultaneously tackle the planning *and* the control problems by providing an explicit policy for the input of the robots, which is to be executed in real time. The framework proposed in this paper is inspired by such control policies leveraging the aforementioned attributes.

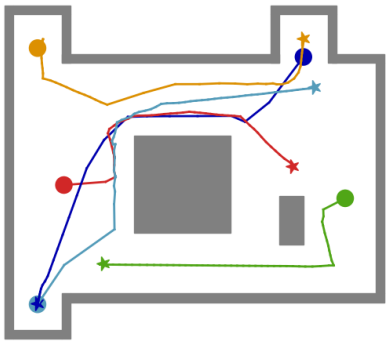

Fig. 1. Five circular robots reaching their goals (marked as stars with the same colors) in a complex workspace, by following the proposed method. Scene adapted from [1].

Nevertheless, the convergence and collision avoidance guarantees of feedback control policies usually assume spherical workspaces, or "sphere worlds" [9], i.e., where the robots and the obstacles are approximated by spheres, which must be separated by a sufficiently large distance. These assumptions might not hold in some scenarios e.g., office buildings or storage facilities with narrow passages (see, e.g., Fig. 1), and hence the aforementioned techniques fail to provide sufficient guarantees.

Non-spherical workspaces have been considered for *single-robot* navigation in the special case of "star-worlds" [10] or arbitrary 2D workspaces [11], without, however being straightforward to extend to multi-robot scenarios; [7] deals with multi-robot navigation of arbitrary shapes, limited to 2D workspaces, but without completely avoiding local minima configurations. An online Model Predictive Control approach is employed in [8], which also suffers from undesired local equilibria in complex workspaces. Finally, an alternative hybrid control framework using hierarchical clustering is proposed for spherical robots without the presence of obstacles in [4]. Another assumption of the aforementioned works is the simplified robot dynamics (aka differential constraints) they consider, usually being single- or double-integrators, which neglect potential uncertainties (e.g., unknown friction terms in ground robots) or other disturbances.

Motion planners, on the other hand, can tackle more

[1]TP, AMW, and LEK are with the Dept. of Comp. Sci. at Rice Univ. Emails: {tp36, andrew.wells, kavraki}@rice.edu. This work is supported in part by NSF 1830549. Work on this project by AMW has been supported by NASA 80NSSC17K0162.

[2]CKV and DVD are with the KTH Center of Autonomous Systems, School of Electr. Engin. and Comp. Sci. Emails: {cverginis, dimos}@kth.se . CKV and DVD are supported by the H2020 ERC Starting Grant BUCOPHSYS and the European Union's Horizon 2020 Research and Innovation Programme under the GA No. 731869 (Co4Robots).

general problems, dealing with a larger variety of robot and obstacle shapes. Importantly, they provide a single plan and not policies. A special class of motion planners are sampling-based motion planners, which have become popular due to their ability to provide plans in high-dimensional spaces while providing probabilistic completeness under a loose set of assumptions. The main idea behind these planners is to sample random configurations in the free space and attempt connections between these, either geometrically (geometric motion planning) or by applying random control inputs in the robot dynamic model - differential constraints (kinodynamic planning) [12]. For multiple robots, these planners usually treat the entire team as a single system. This "centralized" approach suffers from the curse of dimensionality [12].

Several works in motion planning present specialized planners aimed at reducing the effect of dimensionality. In [13] the authors compose roadmaps for each robot and search using this composite map. This method is not directly applicable to kinodynamic planning as it may be infeasible to construct a road map, depending on the underlying dynamics. Other approaches have focused on discrete planning, for example over a pebble graph [14]. In these works, a plan is valid if it never places two robots at the same vertex at the same time. This abstraction makes the search more efficient, but geometric information is lost as robots are treated as points on a discrete graph. In [15], the authors employ previous results in discrete leads for the multi-robot motion planning problem. A discretization (e.g., a triangulation) is imposed on the workspace of the robots, which is then used for multi-robot planning. Once a multi-robot plan is found, each robot attempts to follow the discrete lead corresponding to its role in the overall plan. The attempts to follow these leads are adapted over time in order to provide probabilistic completeness.

Even multi-robot motion planners that consider dynamics (e.g., [15]) simulate forward the robot dynamics with randomly generated inputs and output the ones that lead the system safely to the goal. If there are uncertainties in the dynamics model, as considered in this paper, this forward simulation will deviate from the actual robot trajectories, leading to inaccurate solutions.

Since closed-form feedback control policies provide efficient real-time solutions for the multi-robot safe navigation problem, and can be made robust with respect to model uncertainties and disturbances, we aim in this work to exploit their capabilities, by integrating them with sampling-based motion planning techniques. In this way, we can overcome the limitations of potential field, such as local minima. Multi-robot motion planners suffer from poor scalability and inaccurate solutions in the case of uncertain dynamics [16], [12]. By exploiting the advantages of both methodologies, we provide a computationally efficient solution that navigates the robots safely to their goals from all collision-free initial configurations, even in the case of uncertain dynamics.

In this paper, we present a multi-robot feedback control policy, based on potential field, for safe navigation in a complex workspace cluttered with (potentially non-convex) obstacles. In the case the robots fall in a local minima configuration, led by the complexity of the workspace, a simple geometric sampling-based motion planner is used to derive a geometric path that frees the robots from this configuration. This is performed sequentially for increasing number of robots, alleviating the curse of dimensionality when possible. Path obtained by the motion planners are tracked by a separate path-tracking control policy that is robust to uncertain dynamics. After traversing the paths to escape the local minima, the robots switch to the original feedback control policy. From this point the robots will either complete their navigation (and thus stop) or get stuck in a new local minima, in which case the process repeats until completion. The considered robots obey 2nd-order dynamics with uncertainties and disturbances unknown to the user, and the proposed control policies exhibit provable robustness against these uncertainties. Our approach allows us to solve *efficiently* and in real-time the multi-robot navigation problem for a large variety of complex workspaces, and in cases that cannot be tackled by the original control policy. The validity and efficiency of the approach is demonstrated with extensive experimental studies in simulated environments based on real-world buildings.

## II. Problem Formulation and Methodology

Consider $N$ robots operating in a workspace $\mathcal{W} \subset \mathbb{R}^n$, with index set $\mathcal{N} := \{1, \ldots, N\}$. 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 $i \in \mathcal{N}$:

$$\ddot{x}_i = u_i + f_i(x_i, \dot{x}_i, t), \tag{1}$$

where $f_i : \mathbb{R}^{2n} \times \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is a smooth *unknown* function modeling disturbances, model uncertainties etc, satisfying $f_i(x_i, 0, t) = 0$, $\forall x_i \in \mathbb{R}^n$, $t \in \mathbb{R}_{\geq 0}$ and $\|f_i(x_i, \dot{x}_i, t)\| \leq \bar{f}_i$, $\forall (x_i, \dot{x}_i) \in \mathbb{R}^{2n}$, $t \in \mathbb{R}_{\geq 0}$, for a positive unknown constant $\bar{f}_i$, $\forall i \in \mathcal{N}$. Moreover, in view of the aforementioned properties, it is reasonable to assume that there exist $\lambda_i$, $\epsilon_i$ such that $\|f_i(x_i, \dot{x}_i, t)\| \leq \lambda_i \|\dot{x}_i\|$, when $\|\dot{x}_i\| \leq \epsilon_i$, $\forall i \in \mathcal{N}$ [17]; $u_i \in \mathbb{R}^n$ is robot $i$'s control input, $\forall i \in \mathcal{N}$.

Moreover, let there be $M$ disjoint obstacles in the workspace $\mathcal{O}_k \subset \mathcal{W}$, $\forall k \in \mathcal{M} := \{1, \ldots, M\}$, which *are not assumed to be spherical or convex*, as in the majority of the feedback control-based related works, e.g., [5], [3], [18]. The goal of the robots is to navigate to predefined goal configurations $g_i, i \in \mathcal{N}$ in the free space while avoiding collisions with each other and the workspace obstacles. The goals are assumed to be sufficiently far from the obstacles as well as each other, i.e., $\|g_i - g_j\| > r_i + r_j + \nu$, $\forall i, j \in \mathcal{N}$, $i \neq j$, and $\min_{k \in \mathcal{M}} \inf_{y \in \mathcal{O}_k} \|g_i - y\| > r_i + \nu_o$, $\forall i \in \mathcal{N}$, $k \in \mathcal{M}$, for positive constants $\nu_o$, $\nu$.

### A. Feedback Control

*1) Artificial Potential Field-based Navigation:* We aim here to design a decentralized feedback control law that aims

at navigating the robots to their goals while avoiding colliding with each other and the workspace obstacles. The control design follows standard potential field-based techniques, adjusted for the 2nd-order uncertain dynamics (1). The shapes of the obstacles might not have a closed-form expression describing them, and hence we approximate obstacle $k$ by $L_k$ spheres, and define $\bar{\mathcal{L}} \coloneqq \{1, \dots, L_1, L_1+1, \dots, \sum_{k\in\mathcal{M}} L_k\}$ 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 $k$th obstacle sphere. Note that the spheres might be intersecting. By defining $\gamma_i \coloneqq x_i - g_i \in \mathbb{R}^n$, $\forall i \in \mathcal{N}$, $d_{ij} \coloneqq \|x_i - x_j\|^2 - (r_i + r_j)^2 - \sigma$, $\forall i, j \in \mathcal{N}$, $i \neq j$, and distances to obstacle spheres as $d^o_{ik} \coloneqq \|x_i - o_k\|^2 - (r_i + r_{o_k})^2 - \sigma_o$, $\forall k \in \bar{\mathcal{L}}$, the control design aims at guaranteeing $\lim_{t\to\infty} \gamma_i(t) = 0$, and $d_{ij}(t) > 0$, $d^o_{ik}(t) > 0$, $\forall t \geq 0$, $i, j \in \mathcal{N}$, $i \neq j$, $k \in \bar{\mathcal{L}}$. (Note, squaring is only needed to ensure the equations are differentiable everywhere.) The constants $\sigma$ and $\sigma_o$ are safety margins that will be used in the subsequent sections.

In order to reduce the local minima configurations, we wish the robots to be affected by other robots and the obstacles (in order 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 function $\beta(\cdot, y) : [0, y] \to [0, \bar{\beta}]$, with

$$\beta(x, y) \coloneqq \begin{cases} \vartheta(x), & 0 \leq x \leq y \\ \bar{\beta}, & y \leq x \end{cases},$$

for positive constants $y, \bar{\beta}$, and an appropriate polynomial $\vartheta(x)$ that guarantees that $\beta(x, y)$ is twice continuously differentiable. We define now $\beta_{ij} \coloneqq \beta_{ij}(d_{ij}) \coloneqq \beta(d_{ij}, \underline{s})$, $\forall i, j \in \mathcal{N}$, $i \neq j$, 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 $\beta_{ij}$'s derivative is zero when $d_{ij} > \underline{s}$) Given $\beta_{ij}$, we define then the barrier-like functions $b_{ij} \coloneqq \beta_{ij}(d_{ij})^{-1}$ 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 $b^o_{ik} \coloneqq (\beta^o_{ik})^{-1}$, where $\beta^o_{ik} \coloneqq \beta^o_{ik}(d^o_{ik}) \coloneqq \beta(d^o_{ik}, \underline{s}_o)$, $\forall i \in \mathcal{N}, k \in \bar{\mathcal{L}}$, for a small positive constant $\underline{s}_o$ respresenting the distances where the robots take the obstacles into account for collision avoidance. Notice that the smaller the region of collision influence is (defined by $\underline{s}$, $\underline{s}_o$), the fewer the local minima configurations are.

We design next the control scheme. Define first the reference velocity signals as

$$v_{r_i} \coloneqq \sum_{j\in\mathcal{N}\setminus\{i\}} \alpha_{ij} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j) - \sum_{k\in\bar{\mathcal{L}}} \frac{\partial b^o_{ik}}{\partial d^o_{ik}}(x_i - o_k) - k_i\gamma_i \tag{2}$$

where $\alpha_{ij} = -1$ if $i > j$, $\alpha_{ij} = 1$ if $i < j$, and $k_i$ are positive constant gains, $\forall i \in \mathcal{N}$. Note that $v_{r_i} \to \infty$ in a collision with an obstacle or another robot. The control law is then designed as

$$u_i = -v_{r_i} + \dot{v}_{r_i} - k_{v_i} e_{v_i}, \tag{3}$$

where $e_{v_i} \coloneqq \dot{x}_i - v_{r_i}$, and $k_{v_i}$ are positive constant gains, $\forall i \in \mathcal{N}$. In order to prove the correctness of the aforementioned control policy, consider the Lyapunov function candidate

$$V = \sum_{i\in\mathcal{N}} \left\{ \frac{k_i}{2}\|\gamma_i\|^2 + \sum_{k\in\bar{\mathcal{L}}} b^o_{ik} + \frac{1}{2}\|e_{v_i}\|^2 \right\} + \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}\setminus\{j\}} b_{ij},$$

which, assuming $d_{ij} > 0$, $d^o_{ik} > 0$ at $t = 0$, is well-defined. The derivative of $V$, after noting that $\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}\setminus\{j\}} \dot{b}_{ij} = -\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}\setminus\{i\}} \alpha_{ij} \frac{\partial b_{ij}}{\partial d_{ij}} (x_i - x_j)^\top \dot{x}_i$, writes

$$\dot{V} = -\sum_{i\in\mathcal{N}} \left\{ v_{r_i}^\top \dot{x}_i - e_{v_i}^\top (u_i - \dot{v}_{r_i} + f_i(x_i, \dot{x}_i, t)) \right\}.$$

By substituting $\dot{x}_i = v_{r_i} + e_{v_i}$ and (2), (3), $\dot{V}$ becomes

$$\dot{V} = -\sum_{i\in\mathcal{N}} \left\{ \|v_{r_i}\|^2 + k_{v_i}\|e_{v_i}\|^2 - e_{v_i}^\top f(x_i, v_i, t) \right\}, \tag{4}$$

and by using $\|f_i(\cdot)\| \leq \bar{f}_i$ and completing the squares,

$$\dot{V} \leq -\sum_{i\in\mathcal{N}} \left\{ \|v_{r_i}\|^2 + \frac{k_{v_i}}{2}\|e_{v_i}\|^2 - \frac{\bar{f}_i}{2k_i} \right\}.$$

Hence, $\dot{V} < 0$ when $\|v_{r_i}\| > \sqrt{\frac{\bar{f}_i}{2k_{v_i}}}$ or $\|e_{v_i}\| > \sqrt{\frac{\bar{f}_i}{k^2_{v_i}}}$, which implies that $v_{r_i}(t)$ and $e_{v_i}(t)$ remain bounded, $\forall t \geq 0$. From continuity, and the fact that the obstacles are not placed at infinity, this implies that $\frac{\partial b_{ij}}{\partial d_{ij}}$ and $\frac{\partial b^o_{ik}}{\partial d^o_{ik}}$ also remain bounded, guaranteeing thus obstacle and inter-robot collision avoidance. Moreover, according to [17, Th. 4.18], the system will converge to a set around the equilibrium $(v_{r_i}, e_{v_i}) = (0, 0)$, $\forall i \in \mathcal{N}$, where it holds $\dot{x}_i = 0$, $k_i\gamma_i = \sum_{j\in\mathcal{N}\setminus\{i\}} \alpha_{ij} \frac{\partial b_{ij}}{\partial d_{ij}}(x_i - x_j) - \sum_{k\in\bar{\mathcal{L}}} \frac{\partial b^o_{ij}}{\partial d^o_{ik}}(x_i - o_k)$, $\forall i \in \mathcal{N}$. By selecting large enough gains $k_{v_i}$, this set can be shrunk to achieve $\|\dot{x}_i\| \leq \epsilon$, where $\|f_i(\cdot)\| \leq \lambda_i\|\dot{x}_i\|$ holds. Hence, in that set, (4) becomes, after using $\dot{x}_i = e_{v_i} + v_{r_i}$ and completing the squares

$$\dot{V} \leq -\sum_{i\in\mathcal{N}} \left\{ \frac{1}{2}\|v_{r_i}\|^2 + \frac{1}{2}\left(k_{v_i} - 2\lambda - \lambda^2\right)\|e_{v_i}\|^2 \right\},$$

and hence, by choosing large enough $k_{v_i}$ so that $k_{v_i} > \lambda^2 + 2\lambda$, we can guarantee by Barbalat's Lemma [17] that the system will converge to the set $\mathbb{S} \coloneqq \{x \in \mathbb{R}^{Nn} : v_{r_i} = e_{v_i} = 0, \forall i \in \mathcal{N}\} = \{x \in \mathbb{R}^{Nn} : \dot{x}_i = v_{r_i} = 0, \forall i \in \mathcal{N}\}$. Note that, by sufficiently reducing $\underline{s}_o$ (in the definition of $\beta^o_{ik}$), we can achieve $\|g_i - o_k\| > r_{o_k}$, $\forall i \in \mathcal{N}, k \in \bar{\mathcal{L}}$, i.e., there's no influence from the obstacles at the goal configurations (note that this is feasible due to the assumption on the goal distances). In that case, the configurations in the set $\{x \in \mathbb{R}^{Nn} : d_{ij} > \underline{s}, \forall i, j \in \mathcal{N}, i \neq j\} \cap \mathbb{S}$ are the desired equilibria, since $v_{r_i} = k_i\gamma_i = 0$, $\forall i \in \mathcal{N}$, in that set. Except for the desired configuration, $\mathbb{S}$ contains also several

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

undesired local minima configurations. In the next section, we will devise a higher-level sampling-based algorithm, that overcomes these configurations.

*2) Path Tracking Control Scheme:* We design here a simple control scheme that achieves tracking of a collision-free path $x_{\mathrm{d}}$ within certain bounds, using the Prescribed Performance Control (PPC) methodology [19]. This path will be the output of the geometric sampling-based motion planner of 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 $x_{\mathrm{d}}$ 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. Nevertheless, the design can be trivially extended to simultaneous tracking by more than one robot (of respective paths), since it doesn't interact with the other robots. Moreover, we consider that $x_{\mathrm{d}}(0) = x_i^*(0)$, which can be guaranteed by the sampling-based planner of the next section.

Prescribed Performance Control achieves evolution of an error state in prescribed bounds, regardless of potentially unknown dynamic terms (here $f_i(\cdot)$ in (1)). More specifically, PPC aims at guaranteeing $|e_\ell(t)| < \rho$, $\forall \ell \in \{1, \dots, n\}$, $t \geq 0$, where $e_\ell$ is the $\ell$th component of the error metric $e \coloneqq x_i^* - x_{\mathrm{d}}$, and $\rho > 0$ is a *prescribed* positive constant[2] to bound $e(t)$. This $\rho$ will be 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 $x_{\mathrm{d}}$.

The potential field control policy from the previous subsection guarantees that, at this local minima configuration, it will hold that $\|x_i^* - x_j^*\|^2 > (r_i + r_j)^2 + \sigma$, $\forall i, j \in \mathcal{N}$, $i \neq j$, and $\|x_i^* - o_k\|^2 > (r_i + r_{o_k})^2 + \sigma_o$, $\forall i \in \mathcal{N}$, $k \in \bar{\mathcal{L}}$. Since the spheres are overapproximations of the robots and the obstacles, it will also hold that

$$\inf_{y \in \mathcal{A}_j(x_j^*)} \|x_i^* - y\| > \widetilde{\sigma}, \forall i, j \in \mathcal{N}, i \neq j \tag{5a}$$

$$\inf_{y \in \mathcal{O}_k} \|x_i^* - y\| > \widetilde{\sigma}_o, \forall i \in \mathcal{N}, k \in \mathcal{M}, \tag{5b}$$

where $\widetilde{\sigma}$, $\widetilde{\sigma}_o$ are constants that can be derived from $\sigma$ and $\sigma_o$, respectively and the radii of the approximating spheres. Assume now that $x_{\mathrm{d}}(t)$ satisfies the following for robot $i$:

$$\begin{aligned} |e_\ell| \leq \min\{\widetilde{\sigma}, \widetilde{\sigma}_o\} \Rightarrow \\ \mathcal{A}_i(x_i(t)) \cap \mathcal{A}_j(x_j^*) = \mathcal{A}_i(x_i(t)) \cap \mathcal{O}_k = \emptyset, \end{aligned} \tag{6}$$

$\forall \ell \in \{1, \dots, n\}$, $j \in \mathcal{N} \backslash \{i\}, k \in \mathcal{M}$ and $t \in [0, t_f]$. In other words, the path $x_{\mathrm{d}}$ is chosen such that, if robot $i$ tracks it sufficiently close (defined by $\min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$), it won't collide with other robots or obstacles. Note that such a choice for $x_{\mathrm{d}}(t)$ is feasible due to (5) and will be enforced in the motion planner of the next subsection. We will choose the constant $\rho$ such that $\rho < \min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$. Then, since the following control design will satisfy $-\rho < e_\ell(t) < \rho$, $\forall i \in [0, t_f]$, $\ell \in \{1, \dots, n\}$, collisions between robot $i$ and other robots/obstacles is provably avoided. The developed control design, described subsequently, follows closely the traditional PPC methodology (e.g., [19]).

We design first a reference velocity vector as as

$$v_r \coloneqq \dot{x}_{\mathrm{d}} - k r_\varepsilon \varepsilon, \tag{7}$$

where $k$ is a positive constant gain, and $\varepsilon \coloneqq [\varepsilon_1, \dots, \varepsilon_n]^\top$, $r_\varepsilon \coloneqq \mathrm{diag}\{[r_{\varepsilon_\ell}]_{\ell \in \{1,\dots,n\}}\}$, with $\varepsilon_\ell \coloneqq \ln \frac{1 + \frac{e_\ell}{\rho}}{1 - \frac{e_\ell}{\rho}}$, $r_{\varepsilon_\ell} \coloneqq \frac{2\rho^2}{\rho^2 - e_\ell^2}$, $\forall \ell \in \{1, \dots, n\}$. The intuition here is that we aim to keep $\varepsilon$ bounded in a *compact* set, which would then imply that $-\rho < e_\ell(t) < \rho$, $\forall \ell \in \{1, \dots, n\}, t \geq 0$. We define the velocity error $e_v = \dot{x}_i - v_r$, and design the control input as

$$u_i = \dot{v}_r - \rho^{-1} r_\varepsilon \varepsilon - k_v e_v, \tag{8}$$

for a positive constant gain $k_v$. To show the correctness of the aforementioned policy, consider the Lyapunov function

$$V = \frac{1}{2}\|\varepsilon\|^2 + \frac{1}{2}\|e_v\|^2,$$

which is well-defined at $t = 0$ since $x(0) = x_{\mathrm{d}}(0)$ implying that $-\rho < e_\ell(0) < \rho$, $\ell \in \{1, \dots, n\}$. Differentiating $V$ yields

$$\begin{aligned} \dot{V} =& \varepsilon^\top r_\varepsilon \rho^{-1}(\dot{x}_i - \dot{x}_{\mathrm{d}}) + e_v^\top (u_i - \dot{v}_r + f_i(x_i, \dot{x}_i, t)) \\ =& -k\rho^{-1}\|r_\varepsilon \varepsilon\|^2 - k_v \|e_v\|^2 + e_v^\top f_i(x_i, \dot{x}_i, t) \end{aligned}$$

and by using $\|f_i(x_i, \dot{x}_i, t)\| \leq \bar{f}_i$ as well as completing the squares, we obtain

$$\dot{V} \leq -k\rho^{-1}\|r_\varepsilon \varepsilon\|^2 - \frac{k_v}{2}\|e_v\|^2 + \frac{\bar{f}^2}{2k_v}$$

Hence, $\dot{V} < 0$ when $\|e_v\| > \frac{\bar{f}}{k_v^2}$ or $\|r_\varepsilon \varepsilon\| > \bar{f}\sqrt{\frac{\rho}{2kk_v}}$, which is satisfied when $\|\varepsilon\| > \sqrt{\bar{f}}\sqrt[4]{\frac{\rho}{2kk_v}}$, since it can be verified that$\|r_\varepsilon\| > \|\varepsilon\|$. We conclude then by [17, Th. 4.18] that $e_v$ and $\varepsilon$ are ultimately bounded in compact sets, as $\|e_v(t)\| \leq \bar{e}_v$, $\|\varepsilon(t)\| \leq \bar{\varepsilon}$ for positive and *unknown* constants $\bar{e}_v$, $\bar{\varepsilon}$. Therefore, by using the inverse logarithm, we obtain

$$-1 < \frac{\exp(-\bar{\varepsilon}) - 1}{\exp(-\bar{\varepsilon}) + 1} \leq \frac{e_\ell(t)}{\rho} \leq \frac{\exp(\bar{\varepsilon}) - 1}{\exp(\bar{\varepsilon}) + 1} < 1,$$

implying that $-\rho < e_\ell(t) < \rho$, $\forall \ell \in \{1, \dots, n\}$, $t \geq 0$, achieving thus tracking of $x_{\mathrm{d}}(t)$ with prescribed performance, without requiring knowledge of $f_i(\cdot)$ or its bound.

### *B. Application of Motion Planner*

Rather than attempting to compute a motion plan before moving, we apply the potential field-based control policy to the real robots and only pause to compute plans when a local minima is detected. By default, the robots move in real-time according to the potential field-based controller (Alg. 1 line: 2). At any time step, if we detect that the multi-robot system is stuck in a local-minimum, we call a motion planner on a subset of robots. Once a path that escapes the local minimum has been found, it is tracked using the path-tracking control

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

policy of Section II.A.2). Once path-tracking completes, we return to the original multi-robot policy in the current state.

We detect local minima by testing whether the velocity for any robot is above a small threshold $\delta$ (Alg. 1 line: 3). If no robot has velocity greater than $\delta$ and if any robot has yet to reach its goal, then the potential field policy is at a local minima. Once this state is detected, we call the motion planner (Alg. 1 line: 8) with a specified timeout value (parameter *MPTimeout*) as well as a limit on the number of robots to consider for simultaneous motion planning (*NRobMP*). The motion planner is to return paths with a clearance *MinClearance* $= \min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$, as required by the path tracking controller (see (6)).

Initially *NRobMP* is set to 1 (Alg. 1 line: 6) such that only a single robot is chosen for motion planning. All robots are considered, one at a time. Later *NRobMP* increases. In general, one philosophy is that we attempt to escape local minima by moving as few robots simultaneously as possible. This allows us to avoid the curse of dimensionality as much as we can. To preserve completeness, we may need to invoke motion planning on all robots (even robots that have already reached their goal). Additionally, we need to increase the motion planning time. Our heuristic to determine which subset of robots should be passed to the motion planner is discussed below. We increment *MPTimeout* and *NRobMP* until the motion planner returns a plan. The path tracking policy follows the trajectory and drives the robots to the new state. Then we switch back to the potential field-based controller and follow the control policy again. This repeats until the problem is solved or the total timeout is reached.

The pseudocode describing the application of the motion planner (function call at Alg. 1 line: 8) is shown in Algorithm 2. Here $\mathcal{A}$ refers to the set of all robots and $\mathcal{R}_s$ refers to robot $s$. Let $\mathcal{A}_{arrived} \subset \mathcal{A}$ be the set of robots that have reached their goals, and $\mathcal{A}_{unarrived} = \mathcal{A} \setminus \mathcal{A}_{arrived}$ the set of robots that have not reached their goals.

The function $getSubsetsOfRobots(\mathcal{A}, \mathcal{A}_{arrived}, n_s)$ returns a list of subsets of up to $n_s$ robots. The list is ordered such that robots in $\mathcal{A}_{unarrived}$ are prefered. Of these, subsets are ordered based on the smallest summed Euclidean distance to the respective goals of the robots. Once all robots in $\mathcal{A}_{unarrived}$ are selected, we consider subsets with additional robots, ordered such that robot $i_s \in \mathcal{A}_{arrived}$ is included for $i_s = \text{argmin}_{j \in Arr \setminus \mathcal{R}_s} \sum_{k \in \mathcal{R}_s} \|x_j^* - x_k^*\|$. We iterate through these subsets in order and attempt motion planning on each.

Other heuristics could be applied. Because we consider all subsets, the heuristic only affects the order of consideration. As one final note, in the case where a robot's goal is within constant $\min\{\widetilde{\sigma}, \widetilde{\sigma}_o\}$ of an obstacle, clearly no path with *MinClearance* to this goal exists. We then use a trajectory that travels to the closest point with *MinClearance* clearance from all obstacles and rely on the potential field controller to move the robot the remaining distance.

Note that because our method achieves probabilistic completeness by increasing *MPTimeout* and *NRobMP*, the underlying motion planner must be probabilistically complete.

*Remark 1:* It is worth noting that the motion planner can handle non-spherical robots, providing less conservative solutions. Moreover, observe that the control policies can run locally by each robot, since the variables $\underline{s}$, $\underline{s}_o$ can resemble a local sensing/communication radius. We plan to consider these in future work.

**Algorithm 1:** Control Policy with Motion Planner

**Input:** $\mathcal{A} = \langle \mathcal{A}_1(x_1), ..., \mathcal{A}_n(x_n) \rangle$
**Input:** timeout
**Output:** $P$ // Control Policy
1 **repeat**
2  $u_{r_i}, v_{r_i} \leftarrow controlPolicy(\mathcal{A})$ // Potential field control policy
3  **if** $v_{r_i} < \delta, \forall i \in \mathcal{N}$ // Detect deadlock **then**
4  **if** *not* $allArrived(\mathcal{A})$ **then**
5  $MPT \leftarrow 2$ // Motion planning timeout
6  $NRobMP \leftarrow 1$ // Max number of robots for motion planning
7  **repeat**
8  $\mathcal{T} \leftarrow callMP(\mathcal{A}, \mathcal{A}_{arrived}, MPT, NRobMP)$ // Call motion planner to resolve deadlock
9  $MPT \leftarrow MPT + 2$
10  $NRobMP \leftarrow NRobMP + 1$
11  **until** $\mathcal{T}$;
12  $\mathcal{A} \leftarrow followPaths(\mathcal{T})$ // Use path-tracking controller
13  **else**
14  **return** $P$
15  **else**
16  $\mathcal{A} \leftarrow applyControl(u_{r_i})$
  // Follow the control policy
17 **until** *timeout*;

## III. Experimental Evaluation

We present a comparison of our new approach that combines the potential field control policy of Section II-A1) with a motion planner to base line approaches of the same control policy without a planner and of the same motion planner without a control policy. Note that because we consider uncertain disturbances in the term $f_i(\cdot)$ (see 1), this is not a "standard" motion planning problem.

We use one artificial example (Fig. 2) and four realistic scenes (Fig. 3) in $\mathbb{R}^2$ for experiments. The realistic scenes are obtained from scans of real buildings found in several works [1] [20] [21]. The start and goal locations of the robots are mostly randomized for the realistic scenes, but several of the locations are manually selected to achieve more difficult settings (e.g. the cluttered initial and goal locations in the narrow passage in the scene 2; 3 goal locations close to the corridor between the small obstacle and the wall in scene 5).

**Algorithm 2:** CallMP(A, $\mathcal{A}_{arrived}$, MPT, NRobMP)

**Input:** $\mathcal{A}$, $\mathcal{A}_{arrived}$, $MPT, NRobMP$
**Output:** $\mathcal{T}$ // Trajectory

1 **foreach** $n_s \in [1, NRobMP]$ **do**
2  $RobotsList \leftarrow$ $getSubsetsOfRobots(\mathcal{A}, \mathcal{A}_{arrived}, n_s)$
3  $Obs = Obs \cup \mathcal{A}$
4  **foreach** $\mathcal{R}_s \in RobotsList$ **do**
5  $Obs = Obs \cap \overline{\mathcal{R}_s}$
6  $\mathcal{T} = EST(\mathcal{R}_s, Obs, MPT)$
7  **if** $dist(finalState(\mathcal{T}), \mathcal{G}) < dist(\mathcal{A}, \mathcal{G})$ **then**
8  **return** $\mathcal{T}$
9 **return** *null*

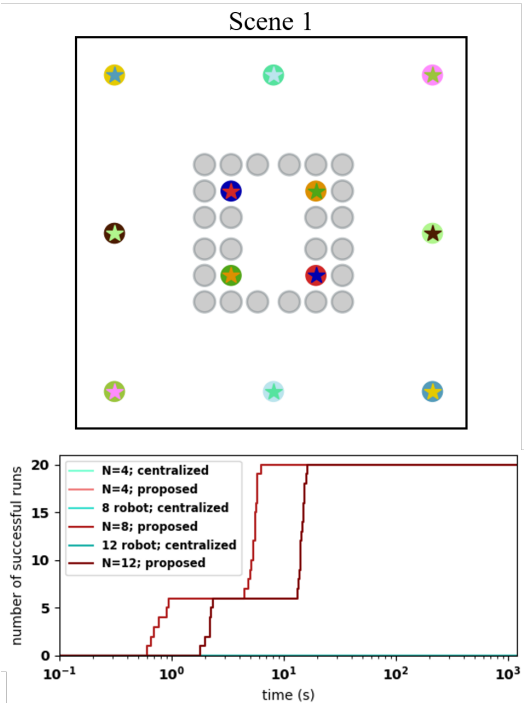


Fig. 2. A typical artificial scene with twelve robots that must swap locations diagonally. The gray circles represent obstacles and the colored circles represent initial positions of the robots. The goal positions are marked as stars with color coded to show the corresponding robot. The figure below the scene is a cactus plot showing the number of successful runs (y-axis) as a function of motion planning time (x-axis. Note the log-scale). Our method in 4-robot setting is not shown, because the control policy solved the problem without calling the motion planner.

We consider robot dynamics of the form (1), with a disturbance of the form $f_i = A_i \|v_i\| \sin(\omega_i t + \phi_i)[q_i, p_i]^\top$, with $A_i$, $\omega_i$, $\phi_i$, $q_i$, $p_i$ random constants, $\forall i \in \mathcal{N}$. Because we allow some bounded disturbance captured by the $f_i$ term, the path-following controller from Section II.A.2) is used to move robots along trajectories output by the motion planner. The control gains of Section II-A are chosen as $k_i = 0.01$, $k_{v_i} = 10$, $k = k_v = 0.5$, $\forall i \in \mathcal{N}$.

We note that the potential field control policy consists of closed-form equations to be applied to a physical robot at any state in real time. The motion planner, however, outputs a single trajectory to be followed without accounting for the disturbance induced by $f_i(\cdot)$. Hence, a distinct comparison of these two methods is not possible. Rather, we compare our new approach to the original closed-form control policy to demonstrate that our new approach can solve more scenes (indeed, our new approach is probabilistically complete assuming a path with clearance *MinClearance* exists from any reachable state). We also compare our new approach to the centralized motion planner to demonstrate that we spend less total time planning.

As our motion planner, we choose EST [22] for both the centralized planner and our method. Any probabilistically complete motion planner could be used. We tested EST and RRT [23], and found EST worked better for our scenarios.

### A. Comparison to potential field

The potential field control policy is unable to fully solve any scenario presented here except Scene 1 with 4 robots. The policy can make significant progress, but gets trapped in local minima. In general, our policy works well for open worlds without narrow passages. We leverage this to make progress in real-world scenarios where the world is mostly open.

### B. Comparison to a Centralized Motion Planner

To compare to a centralized EST motion planner, we record the total time spent in $CallMP(\mathcal{A}, MPT, NRobMP)$ (Alg. 2, line 8). We compare this to the time spent running the centralized EST planner on the problem. We plot these as cactus plots that show the number of successful runs (y-axis) as a function of the total planning time (x-axis). Note that this includes only the time taken by the motion planner, since the feedback control policies are run on-line in real time. All scenarios are run 20 times.

For scene 1 (fig. 2), we tested scenarios with 4, 8 and 12 robots (to maintain symmetry). Our method succeeded on all 20 runs of each scenario with total planning time less than 20 seconds. The centralized motion planner failed to solve the 4 robot scenario within the timeout (1200 seconds).

For the remaining scenes, we tested scenarios with 5, 10 and 15 robots. Fig. 3 shows that our method outperforms the centralized motion planner in terms of both motion planning time and success rate. Our method takes less than 100 seconds for most of the experiment runs, while the centralized planner mostly fails to find a solution within the timeout of 1200 seconds. The only scenario where the centralized planner generally succeeds is in Scene 3 with 5 robots. Here the centralized motion planner takes about 10 to 300 seconds, while our method takes less than 1 second.

In summary, our method outperforms the potential field-based controller by solving many scenes that violate the policy's assumptions. It significantly outperforms a centralized motion planner in terms of planning time. On particularly challenging scenes, our method still occasionally times out (see Fig. 3 Scene 2).

## IV. Discussion and Future Work

This paper integrates feedback control policies with sampling-based motion planning techniques to tackle the

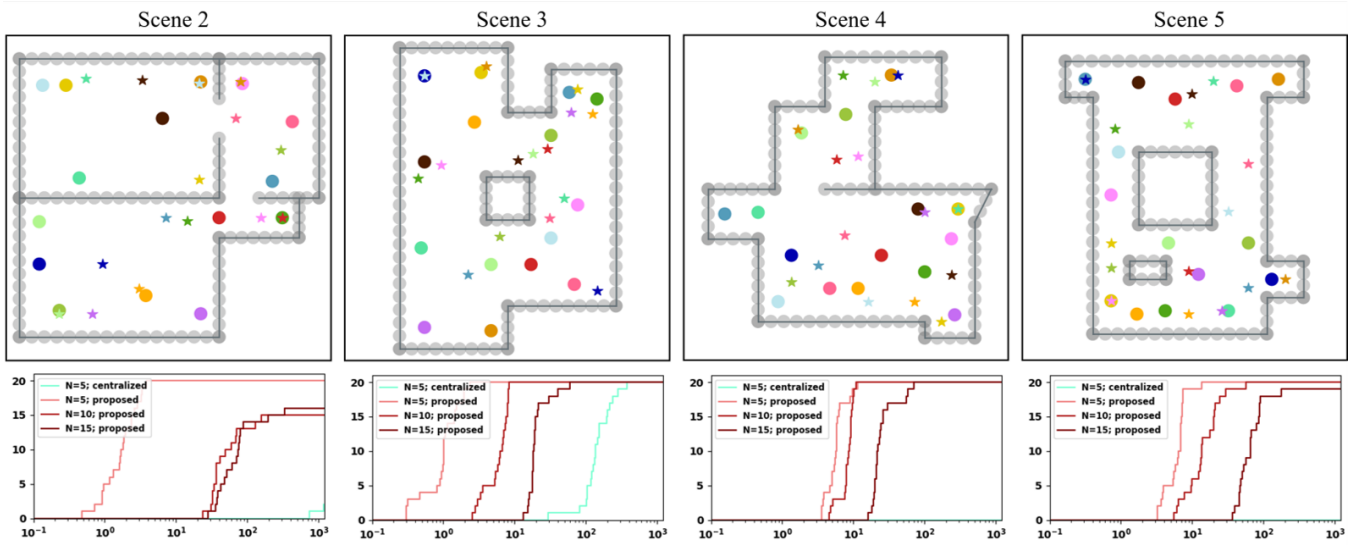


Fig. 3. In the top row, we show four realistic scenes from scans of buildings. We use circular obstacles (gray) to approximate the obstacles in the scene (walls and tables). The initial positions of the robots are colored circles and goal positions are stars of matching color. On the bottom row, we show cactus plots of the number of successful runs (out of twenty) as a function of motion planning time (x-axis. Note log-scale). An omitted line shows that no runs succeeded.

multi-robot safe navigation problem in obstacle-cluttered workspaces with uncertain robot dynamics. We design a closed-form potential field-based control policy that the robots can execute in real-time, and call a motion planner that outputs a geometric path that frees them from potential local minima configurations. This path is tracked by a separate feedback control policy. The robots are assumed to obey 2nd-order uncertain dynamics and both control schemes are robust with respect to those uncertainties. Future efforts will be devoted towards decentralizing the proposed framework as well as generalizing it to arbitrary robot shapes.

## References


[1] B. Krose, R. Bunschoten, S. T. Hagen, B. Terwijn, and N. Vlassis, "Household robots look and learn: environment modeling and localization from an omnidirectional vision system," *IEEE Robotics Automation Magazine*, vol. 11, no. 4, pp. 45–52, Dec 2004.

[2] D. Panagou, D. M. Stipanović, and P. G. Voulgaris, "Distributed coordination control for multi-robot networks using lyapunov-like barrier functions," *IEEE Transactions on Automatic Control*, vol. 61, no. 3, pp. 617–632, 2015.

[3] 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.

[4] O. Arslan, D. P. Guralnik, and D. E. Koditschek, "Coordinated robot navigation via hierarchical clustering," *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 352–371, 2016.

[5] S. G. Loizou and K. J. Kyriakopoulos, "A feedback-based multiagent navigation framework," *International Journal of Systems Science*, vol. 37, no. 6, pp. 377–384, 2006.

[6] 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.

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

[8] 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*, pp. 1–15, 2018.

[9] D. E. Koditschek and E. Rimon, "Robot navigation functions on manifolds with boundary," *Advances in applied mathematics*, vol. 11, p. 412, 1990.

[10] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *Departmental Papers (ESE)*, p. 323, 1992.

[11] P. Vlantis, C. Vrohidis, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Robot navigation in complex workspaces using harmonic maps," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1726–1731, 2018.

[12] H. Choset *et al.*, *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[13] 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," *CoRR*, vol. abs/1305.2889, 2013. [Online]. Available: http://arxiv.org/abs/1305.2889

[14] A. Krontiris, R. Luna, and K. Bekris, "From feasibility tests to path planners for multi-agent pathfinding," *Proceedings of the 6th Annual Symposium on Combinatorial Search, SoCS 2013*, pp. 114–122, 01 2013.

[15] D. Le and E. Plaku, "Multi-robot motion planning with dynamics via coordinated sampling-based expansion guided by multi-agent search," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1868–1875, April 2019.

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

[17] H. K. Khalil and J. W. Grizzle, *Nonlinear systems*. Prentice hall Upper Saddle River, NJ, 2002, vol. 3.

[18] S. Paternain, D. E. Koditschek, and A. Ribeiro, "Navigation functions for convex potentials in a space with convex obstacles," *IEEE Transactions on Automatic Control*, vol. 63, no. 9, pp. 2944–2959, 2017.

[19] 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.

[20] H.-C. Lee, S.-H. Lee, M. H. Choi, and B. H. Lee, "Probabilistic map merging for multi-robot rbpf-slam with unknown initial poses." *Robotica*, vol. 30, no. 2, pp. 205–220, 2012. [Online]. Available: http://dblp.uni-trier.de/db/journals/robotica/robotica30.html#LeeLCL12

[21] Contributors, "Surveyor," https://github.com/gloomyandy/surveyor, 2017.

[22] D. Hsu, J. C. Latombe, and R. Motwani, "Path planning in expansive configuration spaces," in *Proceedings of International Conference on Robotics and Automation*, vol. 3, April 1997, pp. 2719–2726 vol.3.

[23] S. Lavalle and J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," *Algorithmic and computational robotics: New directions*, 01 2000.