2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Scalable distributed algorithms for multi-robot near-optimal motion planning

Guoxiang Zhao, Minghui Zhu

Abstract— **This paper investigates a class of motion planning problems where multiple unicycle robots desire to safely reach their respective goal regions with minimal traveling times. We present a distributed algorithm which integrates decoupled optimal feedback planning and distributed conflict resolution. Collision avoidance and finite-time arrival at the goal regions are formally guaranteed. Further, the computational complexity of the proposed algorithm is independent of the robot number. A set of simulations are conduct to verify the scalability and near-optimality of the proposed algorithm.**

Index Terms— **robotic motion planning, multi-robot optimal coordination, scalability**

I. Introduction

Motion planning of mobile robotic networks is an active research topic, where a set of control policies is identified to command a team of mobile robots from initial states to their respective goal sets, while abiding dynamic constraints and environmental rules. Single-robot motion planning is computationally challenging. In particular, the generalized mover's problem in [1] is shown to be PSPACE-hard in degrees of freedom. Most single-robot planning algorithms fall into three categories: cell decomposition [2] [3], roadmap [4] and potential field [5] [6]. However, these methods suffer from either heavy computations or local minima in potential field in high-dimensional space. Sampling-based algorithms are shown to be effective in addressing computational inefficiency. Rapidly-exploring Random Tree (RRT) algorithm [7] can quickly return a feasible path. However, RRT is not asymptotically optimal [8]; i.e., the length of the shortest path returned by RRT is worse than the global minimum with probability one. Searching for the optimal path is harder than feasible path planning. For example, the shortest path problem in the 3D space is NP-hard in the number of obstacles [9]. Recent paper [8] proposes RRT*, which is both computationally efficient and asymptotically optimal. That is, RRT* only scales constantly with respect to (w.r.t.) RRT in computational complexity but guarantees that the returned solution converges to the optimum almost surely.

Multi-robot motion planning is even more computationally challenging. Centralized planners, e.g., [2], treat all the robots as a single entity. As pointed out in [2], centralized planners are not practical since the worst-case computational complexity grows exponentially as the robot number. Consequently, many alternative planners have been proposed to address the scalability issue. In decoupled planning [10] [11], each robot independently plans its path and a coordination scheme is used to resolve conflicts. In priority planning [12] [13], each robot is assigned with a unique priority such that (s.t.) lower-ranked robots make compromises for higher-ranked ones. These algorithms achieve scalability at the expense of optimality. In [14] and [15], game theory is leveraged to coordinate multiple robots and convergence to Nash equilibria is ensured. However, Nash equilibria may not be socially optimal and the algorithms rely on steering functions, whose solutions are only known for a limited class of dynamic systems.

Distributed control of mobile robots has been extensively studied in recent years [16]–[18]. A widely used idea is to encode the network-wide goal of interest as a team objective function and the control laws of individual robots are derived from their partial derivatives of the team objective function. The synthesized control laws only require local information exchanges and thus are scalable w.r.t. network expansion. This idea has been successfully applied to many multi-robot missions, e.g., consensus [19], formation control [20] [21], vehicle routing [22] and sensor deployment [23] [24]. These algorithms usually use Lyapunov analysis [25]–[27] to guarantee asymptotic convergence to certain critical points of the team objective functions. However, they do not guarantee that aggregate running costs, e.g., travelling times and energy consumptions, are minimized.

Contribution statement: In this paper, we propose a distributed algorithm to coordinate a fleet of unicycle robots. The algorithm integrates decoupled optimal feedback planning and distributed conflict resolution. In particular, each robot individually plans its optimal motion offline and evades other objects within its sensing range in online execution. Since the computational complexity of the decoupled planning is independent of the robot number and conflict resolution is fully distributed, the algorithm is scalable w.r.t. the robot number. In addition, each robot's individual planner is optimal and its motion is rarely interfered in exercise, so the algorithm is also near-optimal; i.e., its loss in optimality is minor compared to benchmark. Safe and finite-time arrival at the goal regions is formally ensured. Simulations confirm the scalability and near-optimality of the proposed algorithm.

II. Problem Formulation

Consider a team of mobile robots, labeled by $\mathcal{V} \triangleq \{1, \dots, N\}$, in an environment equipped with a global coordinate frame with basis (e^x, e^y). Each robot is modeled as a circular disk with radius $d_r > 0$. The state of robot $i \in \mathcal{V}$

This work was partially supported by the grants NSF ECCS-1710859 and NSF CNS 1830390.

G. Zhao and M. Zhu are with School of Electrical Engineering and Computer Science, Pennsylvania State University, University Park, PA 16802. (email: {gxzhao, muz16}@psu.edu)

is denoted by $q_i \triangleq \begin{bmatrix} r_i^T & \theta_i \end{bmatrix}^T \in \mathcal{C}_i \triangleq X_i \times \mathbb{R}$ consisting of the position $r_i \triangleq \begin{bmatrix} x_i & y_i \end{bmatrix}^T$ and the orientation θ_i w.r.t. the global coordinate frame. Each robot is equipped with a body-attached coordinate frame with basis $e_i^x \triangleq \begin{bmatrix} \cos\theta_i & \sin\theta_i \end{bmatrix}^T$ and $e_i^y \triangleq \begin{bmatrix} -\sin\theta_i & \cos\theta_i \end{bmatrix}^T$. The motion of robot i w.r.t. (e^x, e^y) is described as a unicycle and governed by:

$$\dot{q}_i = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \end{bmatrix} u_i, \tag{1}$$

where $u_i \triangleq \begin{bmatrix} v_i & \omega_i \end{bmatrix}^T \in U_i = \mathbb{R}^2$ is the control of robot i consisting of the linear velocity v_i and the angular velocity ω_i. We refer to the vector $v_i e_i^x$ as the velocity of robot i.

The environment is packed with circular obstacles $\mathcal{V}^O$. Obstacle $\ell \in \mathcal{V}^O$ is centered at r_ℓ^O and has radius d_o; i.e., $X_\ell^O \triangleq \{p \in \mathbb{R}^2 | \|p - r_\ell^O\| \leq d_o\}$, where $\|\cdot\|$ is the 2-norm. A robot aims at reaching a goal region while keeping a certain distance away from obstacles and other robots to ensure its safety. The goal region for robot $i \in \mathcal{V}$ is denoted by $X_i^G \subseteq X_i \setminus \bigcup_{\ell \in \mathcal{V}^O} X_\ell^O$ while the set of goal states is denoted by $\mathcal{C}_i^G \triangleq X_i^G \times \mathbb{R}$. The minimum safety distances between two robots and a robot-obstacle pair are denoted by $d_s \triangleq 2d_r$ and $d_{so} \triangleq d_r + d_o$ respectively.

Each robot has limited capabilities to communicate with nearby robots and sense local environment. Specifically, each robot can broadcast messages to nearby robots within a communication range $d_c > d_s$. As in [16], we define the communication graph at position r by $\mathcal{G}^C(r, d_c) \triangleq (\mathcal{V}, \mathcal{E}^C(r, d_c))$, where $r \triangleq \prod_{i \in \mathcal{V}} r_i$ is the collection of all robots' positions, $\mathcal{E}^C(r, d_c) \triangleq \{(i,j) | \|r_i - r_j\| \leq d_c\}$ is the collection of communication edges and each edge (i,j) indicates that the messages of robot i can be transmitted to robot j. The communication range d_c may be omitted when no confusion is caused. Similarly, each robot can measure the states of other robots or the positions of static obstacles within a sensing range $d_{sense} > \max\{d_s, d_{so}\}$. Throughout the paper, we assume the sensing range equals to the communication range.

The problem of interest in this paper is to synthesize distributed controllers s.t. all robots can safely reach their own goal regions and meanwhile their total traveling times are minimized. That is, we search for a set of policies $\prod_{i \in \mathcal{V}} \pi_i$ s.t. the cost functional $\sum_{i \in \mathcal{V}} t_i$ is minimized while for each robot $i \in \mathcal{V}$, the solution of $\dot{q}_i = f(q_i)\pi_i$ satisfies $r_i(t_i) \in X_i^G$ and $d_{ij}(\tau) \geq d_s, \forall j \neq i, \tau \in [0, \min\{t_i, t_j\}]$. We trade small loss in optimality for great scalability in the design of distributed controllers.

III. Notations and Assumptions

Throughout the paper, denote the vector from the center of robot $j \in \mathcal{V}$ to the center of robot $i \in \mathcal{V}$ by $r_{ji} \triangleq r_i - r_j$. Similarly, the vector from the center of obstacle $\ell \in \mathcal{V}^O$ is denoted by $r_{\ell i} \triangleq r_i - r_\ell^O$. The direction of r_{ji} and $r_{\ell i}$ are defined as $e_{\ell i} \triangleq \frac{r_{\ell i}}{\|r_{\ell i}\|}$ and $e_{\ell i} \triangleq \frac{r_{\ell i}}{\|r_{\ell i}\|}$ respectively. Denote the angle between a given vector $p = \begin{bmatrix} p_1 & p_2 \end{bmatrix}^T$ and e^x by $Arg(p) \triangleq \arg\min_{\theta \in [0, 2\pi)} \|\frac{p}{\|p\|} - \begin{bmatrix} \cos\theta & \sin\theta \end{bmatrix}^T\|$.

The set of neighboring robots of robot i is defined by $\mathcal{N}_i(r, d_c) \triangleq \{j \in \mathcal{V} | (j,i) \in \mathcal{E}^C(r, d_c)\}$, whose messages can be received by robot i and whose states can be measured by robot i. Denote the neighboring robots whose indices are larger than robot i by $\mathcal{N}_i^+(r, d_c) \triangleq \{j \in \mathcal{N}_i(r, d_c) | j > i\}$. The second argument may be omitted when no confusion is caused. Define the position set where robot i contacts obstacles by $S_i^O \triangleq \{r_i \in X_i | \exists \ell \in \mathcal{V}^O \text{ s.t. } \|r_\ell^O - r_i\| = d_{so}\}$. Denote the indicator function as $\mathbf{1}_A(a)$, where $\mathbf{1}_A(a) = 1$ if $a \in A$ and $\mathbf{1}_A(a) = 0$ otherwise. The following assumption is imposed throughout the paper.

Assumption 3.1: For all $t \geq 0$ and $i \in \mathcal{V}$, it holds true that $|\mathcal{N}_i^+(r(t))| + \mathbf{1}_{S_i^O}(r_i) \leq 1$.

Assumption 3.1 indicates that any robot $i \in \mathcal{V}$ at any time $t \geq 0$ at most needs to address either one robot with larger index or static obstacles. Under Assumption 3.1, $\mathcal{N}_i^+(r)$ can be either a singleton or an empty set. For brevity, denote the unique element of $\mathcal{N}_i^+(r)$ by j^* if $|\mathcal{N}_i^+(r)| = 1$. Define the safe position space by $S \triangleq \{r \in \prod_{i \in \mathcal{V}} (X_i \setminus \bigcup_{\ell \in \mathcal{V}^O} X_\ell^O) | \forall i \in \mathcal{V} \text{ and } j \in \mathcal{V} \setminus \{i\}, \|r_i - r_j\| \geq d_s, \text{ and } \|r_i - r_\ell^O\| \geq d_{so}, \forall \ell \in \mathcal{V}^O\}$. Moreover, define the restricted safe position space by $\hat{S}_t(d) \triangleq \{r \in S | |\mathcal{N}_i^+(r(t), d)| + \mathbf{1}_{S_i^O}(r_i) \leq 1, \forall i \in \mathcal{V}\}$. Then Assumption 3.1 refers to the position space $\bigcap_{t \geq 0} \hat{S}_t(d_c)$.

The following lemma shows that Assumption 3.1 is mild.

Lemma 3.1: The function $\Psi(d) \triangleq \mu(S) - \mu(\cap_{t \geq 0} \hat{S}_t(d))$ is monotonically increasing for all $d > d_s$ and $\Psi(d_s) = 0$, where $\mu : 2^{\mathbb{R}^{2N}} \to \mathbb{R}_{\geq 0}$ is the Lebesgue measure.

The above lemma shows that for a communication range d that is sufficiently close to d_s, the corresponding safe position space $\hat{S}(d)$ where Assumption 3.1 holds true will be very close to the entire safe position space S; in other words, the cases where Assumption 3.1 does not hold are negligible.

Another assumption on the reachability of the initial state is imposed. We introduce the concept of reachability before the assumption.

Definition 3.1: Given a system satisfying (1) and $q_i(0) = q_{i,1}$, a state $q_{i,2} \in \mathcal{C}_i$ is reachable from $q_{i,1} \in \mathcal{C}_i$ within finite time if there exists $u_i : [0, t_i] \to U_i$ for some $t_i \in [0, +\infty)$ s.t. $q_i(t_i) = q_{i,2}$ and $\|r_i(\tau) - r_\ell^O\| \geq d_{so}, \forall \ell \in \mathcal{V}^O$ and $\tau \in [0, t_i]$.

Assumption 3.2: For each robot $i \in \mathcal{V}$, $\exists q_i \in \mathcal{C}_i^G$ is reachable from the initial state $q_{i,0} \in \mathcal{C}_i$ within finite time.

Since the trajectory of a unicycle robot is reversible, Assumption 3.2 guarantees that for any time $t \geq 0$, every robot $i \in \mathcal{V}$ can reach its goal states $\mathcal{C}_i^G$ within finite time from any state $q_i(t)$.

IV. Algorithm Statement

In this section, we propose a distributed hybrid controller to solve the problem of interest.

Robot i's controller distinguishes three cases:

- Case 1: it is moving in a robot-free environment;
- Case 2: it detects a neighboring robot with larger index coming from the front or back;
- Case 3: it senses a nearby robot with larger index approaching sideways.

Case 1 indicates that robot i can safely proceed and only needs to avoid static obstacles. Case 2 shows that robot i needs to steer away immediately. Case 3 implies that robot i needs to move away immediately and avoid computational singularity at the same time. Robot i chooses its controller mode by checking out these cases. At the end of the iteration, robot i broadcasts its control to all nearby robots.

A. *Robot-free environment and evasion of static obstacles*

Robot i is associated with attractive controller $\mathcal{A}_i$ which can command robot i to its goal region safely within finite time when no moving obstacles interfere. Notice that $\mathcal{A}_i$ ignores other robots and only solves a single-robot optimal motion planning problem. There are a number of numerical algorithms to synthesize $\mathcal{A}_i$; e.g., [28]. Its synthesis is out of scope of this paper. We express the attractive controller as $(v_i^A, \omega_i^A) = \mathcal{A}_i(q_i)$.

B. *Evasion of robots when $(e_i^x)^T e_{j^*i} \neq 0$*

Each robot i only considers nearby robots with larger indices $\mathcal{N}_i^+(r)$. When $\mathcal{N}_i^+(r) \neq \emptyset$, a sufficient condition to avoid inter-robot collision between i and $j^* \in \mathcal{N}_i^+(r)$ is that the velocity of robot i along e_{j^*i} equals to that of robot j^*; that is, $(v_i e_i^x)^T e_{j^*i} = (v_{j^*} e_{j^*}^x)^T e_{j^*i}$. The solution is referred to as the critically evasive linear velocity of robot i w.r.t. robot j^*, and denoted by $v^*_{i|j^*} \triangleq \frac{(v_{j^*} e^x_{j^*})^T e_{j^*i}}{(e_i^x)^T e_{j^*i}}$. See Figure 1 for $(e_i^x)^T e_{j^*i} > 0$ and Figure 2 for $(e_i^x)^T e_{j^*i} < 0$.

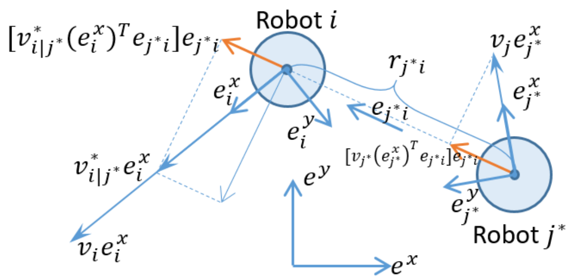

Fig. 1: Robot i evades robot j^* when robot j^* is behind robot i; i.e., $(e_i^x)^T e_{j^*i} > 0$.

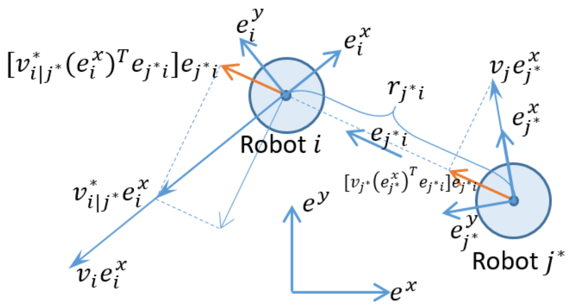

Fig. 2: Robot i evades robot j^* when robot j^* is in front of robot i; i.e., $(e_i^x)^T e_{j^*i} < 0$.

The critically evasive linear velocity $v^*_{i|j^*}$ acts as the boundary of the set of safe linear velocities for robot i. Specifically, if robot j^* shows up behind robot i, i.e., $(e_i^x)^T e_{j^*i} > 0$ as Figure 1, robot i may need to accelerate to avoid rear-end collision, i.e., $v_i \geq v^*_{i|j^*}$; if robot j^* appears in front of robot i, i.e., $(e_i^x)^T e_{j^*i} < 0$ as Figure 2, robot i needs to slow down or even turn around, i.e., $v_i \leq v^*_{i|j^*}$. Then a candidate for v_i is v_i^A, the linear velocity term of $\mathcal{A}_i(q_i)$, which indicates the optimal linear velocity for robot i to reach its goal region at q_i. If choosing v_i^A does not result in collision, choose v_i^A; otherwise, choose the critically evasive linear velocity $v^*_{i|j^*}$. This process is summarized as repulsive controller $\mathcal{R}_i^{R,1}$:

$$\begin{aligned} v_i &= \begin{cases} \max\{v_i^A, v^*_{i|j^*}\}, & \text{if } (e_i^x)^T e_{j^*i} > 0; \\ \min\{v_i^A, v^*_{i|j^*}\}, & \text{if } (e_i^x)^T e_{j^*i} < 0; \end{cases} \\ \omega_i &= k_i(\phi_i^R - \theta_i), \end{aligned} \tag{2}$$

where $\phi_i^R \triangleq Arg(r_{j^*i})$ is the desired steering angle for robot i. When $(e_i^x)^T e_{j^*i} < 0$ and $v^*_{i|j^*} < 0$, v_i is negative, indicating that robot i needs to turn back. See Algorithm 2 for the corresponding pseudo codes.

Notice that computing (2) requires the control of robot j^*. We introduce a procedure $\texttt{receive}_{i,j^*} \in \{\texttt{TRUE}, \texttt{FALSE}\}$ to indicate the reception of the control of robot j^*. If the control is received, $\texttt{receive}_{i,j^*}$ returns $\texttt{TRUE}$; otherwise, it returns $\texttt{FALSE}$. See line 5 in Algorithm 1.

C. *Evasion of robots when $(e_i^x)^T e_{j^*i} = 0$*

Notice that the aforementioned repulsive controller $\mathcal{R}_i^{R,1}$ requires $(e_i^x)^T e_{j^*i} \neq 0$; in other words, when robot j^* is approaching robot i at a direction perpendicular to the orientation of robot i, no matter what control robot i takes, the distance between these two robots $\|r_{ij^*}\|$ keeps decreasing and robot i cannot actively avoid the collision. Hence, a solution is to alert robot i in advance and command it to steer away. Since robot i can instantly evade robot j^* by applying (2) once the condition $(e_i^x)^T e_{j^*i} = 0$ does not hold, robot i can take a linear velocity larger than $(v_{j^*} e^x_{j^*})^T e_i^x$ when the distance of two robots is larger than the safety distance. More specifically, the repulsive controller $\mathcal{R}_i^{R,2}$ fixes robot i's angular velocity $\omega_i = 0$ and commands the linear velocity as

$$v_i = (v_{j^*} e^x_{j^*})^T e_i^x + \delta_i (v_{j^*} e^x_{j^*})^T e_i^y, \tag{3}$$

where $\delta_i \triangleq 1/\sqrt{(\frac{\|r_{j^*i}(T_i)\|}{d_s})^2 - 1} + 1$ is the piecewise constant evading coefficient and T_i is the last time when robot i begins to apply (3). In the second term, δ_i ensures that the distance from robot j^* to the ray of $v_i e_i^x - v_{j^*} e^x_{j^*}$ is larger than the safety distance d_s. See Figure 3 for illustration. Denote the angle between $v_i e_i^x - v_{j^*} e^x_{j^*}$ and $-e_{j^*i}$ by ψ and the distance from robot j^* to $v_i e_i^x - v_{j^*} e^x_{j^*}$ by $D > d_s$. Then δ_i should satisfy that $\sin\psi = \frac{\|\delta_i (v_{j^*} e^x_{j^*})^T e_i^y\|}{\|v_i e_i^x - v_{j^*} e^x_{j^*}\|} = \frac{D}{\|r_{j^*i}(T_i)\|} > \frac{d_s}{\|r_{j^*i}(T_i)\|}$, where the strict inequality is for theoretic completeness.

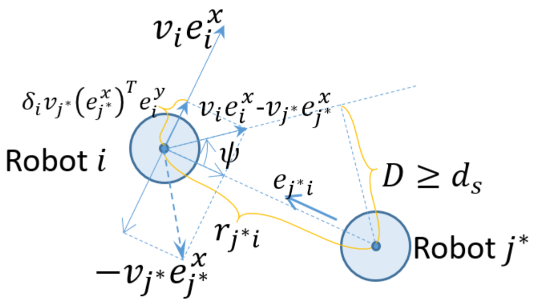

Fig. 3: Robot i evades robot j^* when $(e_i^x)^T e_{j^*i} = 0$.

The corresponding pseudo codes are shown in Algorithm 3. In the algorithm, a variable singularity$\in$ {TRUE, FALSE} indicates whether the condition $(e_i^x)^T e_{j^*i} = 0$ is activated. When robot i executes controller $\mathcal{R}_i^{R,2}$, it checks the value of singularity. If singularity==FALSE, i.e., the condition $(e_i^x)^T e_{j^*i} = 0$ is not active before, $\mathcal{R}_i^{R,2}$ calculates δ_i and sets singularity as TRUE for record; otherwise, i.e., the condition is already active, $\mathcal{R}_i^{R,2}$ simply keeps the last δ_i and proceeds to apply (3). This variable is set FALSE as long as condition $(e_i^x)^T e_{j^*i} = 0$ is invalid.

Algorithm 1 Distributed optimal controller for robot $i \in \mathcal{V}$

```
1:  while $r_i \notin X_i^G$ do
2:      Measure the states of robots $\mathcal{N}_i(r)$;
3:      $(v_i^A, \omega_i^A) = \mathcal{A}_i(q_i)$;
4:      if $\mathcal{N}_i^+(r) \neq \emptyset$ then
5:          if receive$_{i,j^*}(u_{j^*})$==TRUE then
6:              if $(e_i^x)^T e_{j^*i} \neq 0$ then
7:                  singularity=FALSE;
8:                  $(v_i, \omega_i) = \mathcal{R}_i^{R,1}(q_i, v_i^A, j^*, q_{j^*}, u_{j^*})$;
9:              else
10:                 $(v_i, \omega_i) = \mathcal{R}_i^{R,2}(q_i, j^*, q_{j^*}, u_{j^*})$;
11:             end if
12:         end if
13:     else
14:         singularity=FALSE;
15:         $(v_i, \omega_i) = (v_i^A, \omega_i^A)$;
16:     end if
17:     Broadcast $u_i$ to $\mathcal{N}_i(r)$;
18: end while
```

D. Discussion

Algorithm 1 is distributed. On one hand, the attractive controllers solve single-robot optimal motion planning problems and are synthesized in parallel. On the other hand, each robot makes decisions based on states and controls of nearby robots with larger indices and obstacles, which can be obtained locally. The distributed nature contributes to the scalability of Algorithm 1 and removes the dependency of computations on robot number.

V. Analysis

In this section, we show that the proposed algorithm in Section IV guarantees collision avoidance and finite-time arrivals at goal regions. Due to the space limitation, only the statements of theorems are be provided. The following theorem ensures collision avoidance for each robot.

Algorithm 2 Repulsive controller $\mathcal{R}_i^{R,1}$ when $(e_i^x)^T e_{j^*i} \neq 0$

Input: Robot index $i \in \mathcal{V}$ and state q_i, the optimal linear velocity v_i^A, robot j^* and its state q_{j*} and control u_{j^*}
Output: Control u_i

```
1: $v_{i|j^*}^* = \frac{v_{j^*}(e_{j^*}^x)^T r_{j^*i}}{(e_i^x)^T r_{j^*i}}$;
2: if $(e_i^x)^T r_{j^*i} > 0$ then
3:     $v_i = \max\{v_i^A, v_{i|j^*}^*\}$;
4: else
5:     $v_i = \min\{v_i^A, v_{i|j^*}^*\}$;
6: end if
7: $\phi_i^R = Arg(r_{j^*i})$;
8: $\omega_i = k_i(\phi_i^R - \theta_i)$;
```

Algorithm 3 Repulsive controller $\mathcal{R}_i^{R,2}$ when $(e_i^x)^T e_{j^*i} = 0$

Input: Robot index $i \in \mathcal{V}$ and state q_i, the nearest superior robot j^* and its state q_{j*} and control u_{j^*}
Output: Control u_i

```
1: if singularity==FALSE then
2:     $\delta_i = 1/\sqrt{(\frac{\|r_{j^*i}(T_i)\|}{d_s})^2 - 1} + 1$;
3:     singularity=TRUE;
4: end if
5: $v_i = (v_{j^*} e_{j^*}^x)^T e_i^x + \delta_i (v_{j^*} e_{j^*}^x)^T e_i^y$;
6: $\omega_i = 0$;
```

Theorem 5.1: If Assumption 3.1 holds and $\forall i \neq j$, $\|r_{ij}(0)\| > d_s$, then for every $i \in \mathcal{V}$, Algorithm 1 guarantees collision avoidance between robot i and other objects.

The following theorem formally guarantees the finite-time arrivals of all robots at their goal regions.

Theorem 5.2: If Assumptions 3.1 and 3.2 hold true, then every robot can reach its goal region safely within finite time without causing any collisions.

VI. Simulations

In this section, we demonstrate the scalability and near-optimality of Algorithm 1 through simulations.

The simulation environment consists of a single circular obstacle with radius $d_o = 1$ positioned at the origin. At the beginning of each simulation, N unicycle robots modeled as disks with radii $d_r = 0.25$ are uniformly distributed on a circle with radius 10 centered at the origin and the orientation of each robot points to the origin. Each robot can communicate with other robots and sense objects within range $d_c = 0.55$ and is desired to reach another robot's initial position s.t. the total displacement spans over $2\pi/3$ counter-clockwise w.r.t. the origin. Once a robot arrives at its goal region, it is immediately removed from the scene. Five different scenarios are considered, where the only controlled variable is the number of robots $N \in \{5, 10, 15, 20, 25\}$.

A. Implementation of Algorithm 1

Algorithm 1 requires the attractive controller $\mathcal{A}_i$ to solve a single-robot optimal motion planning problem. Essentially, this is a constrained optimal control problem of a nonlinear system, and no analytical solution can be derived. Hence, numerical algorithms are necessary; e.g., the algorithm in Section 3.2.4 of [28], where a continuous-time dynamic system in a continuous state space is approximated by a set-valued dynamic system over a discrete grid. Since each attractive controller $\mathcal{A}_i$ is independent of others, we compute them parallelly. The maximum linear velocity and angular velocity generated by the attractive controller are restricted by 0.5 to satisfy the assumptions imposed in [28]. A regular grid of $77 \times 77 \times 20$ nodes with uniform spatial resolution 0.3 is adopted for each robot. Throughout the computations, the discrete grid and state transitions are shared among robots.

Collision avoidance in [28] is theoretically guaranteed if allowable computation times are infinite or sufficiently large. In contrast, allowable computation times in practical scenarios are always finite, dynamic and uncertain. To address this practical issue, we slightly revise the control strategy when a robot contacts a static obstacle.

The repulsive controller $\mathcal{R}_i^O$ treats the static obstacle $\ell^* \in \mathcal{V}^O$ as a stationary robot and follows the idea of $\mathcal{R}_i^{R,1}$ in (2) to compute the evasive linear velocity when robot i senses the static obstacle ℓ^* nearby; i.e., $\|r_{\ell^*} - r_i\| \le d_{no}$ for some $d_{no} > d_{so}$. This procedure is summarized below:

$$v_i = \begin{cases} \max\{v_i^A, v_{eo}\}, & \text{if } (e_i^x)^T e_{\ell^* i} > 0; \\ \min\{v_i^A, -v_{eo}\}, & \text{if } (e_i^x)^T e_{\ell^* i} < 0; \\ v_i^A, & \text{if } (e_i^x)^T e_{\ell^* i} = 0, \end{cases}$$
$$\omega_i = \omega_i^A,$$

where $v_{eo} > 0$ is the minimum evasive velocity from the obstacle so as to avoid oscillation and $(v_i^A, \omega_i^A) = \mathcal{A}_i(q_i)$ is the control of attractive controller. Compared to the original attractive controller in Section IV-A, the revised control strategy differs in the sensing range d_{no} and the lower bound of the evasive linear velocity v_{eo}.

B. Two competitors

To verify the performance of Algorithm 1, two competitors are constructed.

Centralized coupled algorithm (CC): This type of algorithms, e.g. Algorithm 1 in [29], treats the entire robot team as an entity and computes a central controller to command all the robots. CC is only tested for $N = 2$ robots and adopts a coupled regular grid of $538,752$ nodes with spatial resolution 1.5 since CC is heavy in computations.

Centralized decoupled algorithm (CD): As Section IV-A, CD contructs an attractive controller for each robot which ignores all other robots. Different from Algortihm 1, CD sequencially computes decoupled controllers with a centralized computer and directly implements the computed controllers without coordination among the robots. In the simulations, the settings of attractive controllers follow those in Section VI-A.

C. Scalability

Figure 4 exhibits the computation times for five scenarios of Algorithm 1 and CD. The computation times of CD are linear w.r.t. the number of robots while those of Algorithm 1 are independent of the number of robots. Notice that the computation time for CC exceeds the computation time limit $6,000$ seconds and is not shown in Figure 4.

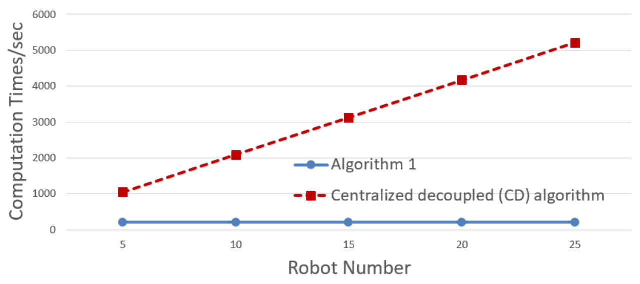

Fig. 4: Computation times of Algorithm 1 and CD w.r.t. different numbers of robots.

D. Near-optimality

To verify the optimality of Algorithm 1, the ideal benchmark is CC, whose optimality and collision avoidance are simultaneously guaranteed. However, this algorithm is not scalable as it takes over $6,000$ seconds to compute for two robots, and therefore cannot be used for a large number of robots. Instead, CD is chosen as the benchmark. Notice that the total traveling times of CD are less than those of CC.

Throughout the simulations, Assumption 3.1 is never violated and Algorithm 1 can successfully drive all robots to their respective goal regions without causing collisions. The travelling times are summarized in Figure 5, where the total traveling times of Algoritm 1 are higher than those of CD, but the differences are small; both traveling times are almost linear w.r.t. the number of robots.

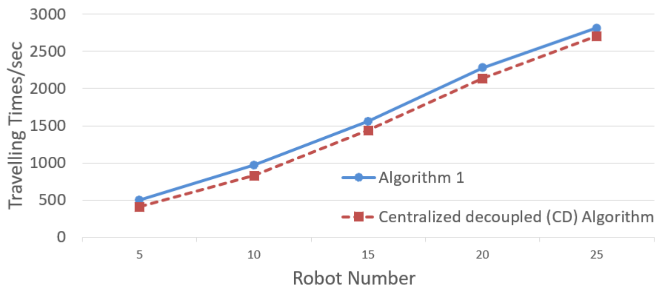

Fig. 5: The total traveling times under the guidance of Algorithm 1 and CD.

In particular, the trajectories of $N = 25$ robots driven by Algorithm 1 are shown in Figure 6. In this simulation, evasions of the static obstacle and moving robots are activated and 10 out of 25 robots apply more than one mode of Algorithm 1. The corresponding minimum distances between any two robots and any pair of a robot and the obstacle over time are shown in Figure 7. It is shown that the minimum distances are always beyond the safety distances and thus the collision avoidance is accomplished.

VII. Conclusion

In this paper, a distributed multi-robot coordination algorithm is proposed. The algorithm formally guarantees collision avoidance and goal region arrival. A set of simulations are conducted to assess the scalability and near-optimality.

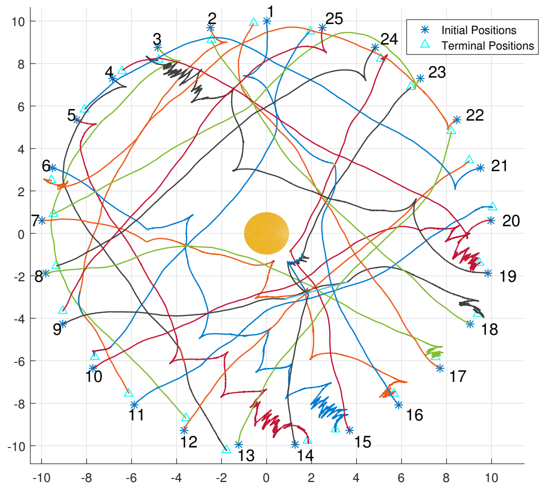

Fig. 6: Trajectories of 25 robots under the guidance of Algorithm 1.

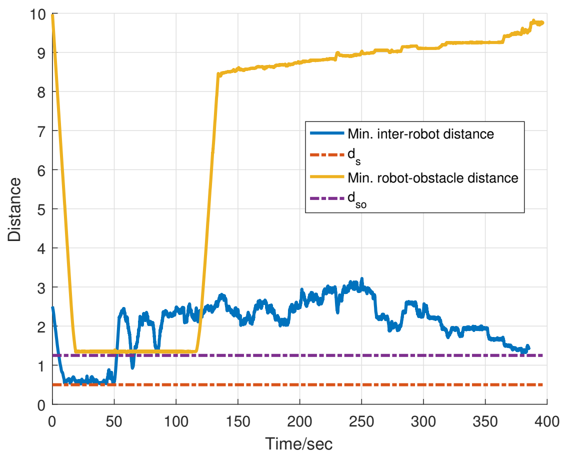

Fig. 7: Minimum distances between two robots and any robot-obstacle pair.

References

[1] J. H. Reif, "Complexity of the mover's problem and generalizations," in *20th Annu. Symp. Found. Comput. Sci.*, Oct. 1979, pp. 421–427.

[2] J. T. Schwartz and M. Sharir, "On the piano movers problem. II. general techniques for computing topological properties of real algebraic manifolds," *Adv. Applied Math.*, vol. 4, no. 3, pp. 298–351, Sep. 1983.

[3] R. A. Brooks and T. Lozano-Perez, "A subdivision algorithm in configuration space for findpath with rotation," *IEEE Trans. Syst., Man, Cybern.*, no. 2, pp. 224–233, Mar. 1985.

[4] J. Canny, *The complexity of robot motion planning*. MIT Press, 1988.

[5] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous robot vehicles*. Springer, 1986, pp. 396–404.

[6] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Proc. 1991 IEEE Int. Conf. Robot. and Automat.*, 1991, pp. 1398–1404.

[7] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *Int. J. Robot. Res.*, vol. 20, no. 5, pp. 378–400, May 2001.

[8] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *Int. J. Robot. Res.*, vol. 30, no. 7, pp. 846–894, 2011.

[9] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Proc. 28th Annu. Symp. Found. Comput. Sci.*, 1987, pp. 49–60.

[10] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 72–89, 1986.

[11] T. Siméon, S. Leroy, and J.-P. Lauumond, "Path coordination for multiple mobile robots: A resolution-complete algorithm," *IEEE Trans. Robot. Autom.*, vol. 18, no. 1, pp. 42–49, 2002.

[12] S. J. Buckley, "Fast motion planning for multiple moving robots," in *Proc. 1989 IEEE Int. Conf. Robot. and Automat.*, May 1989, pp. 322–326 vol.1.

[13] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," *Algorithmica*, vol. 2, no. 1-4, pp. 477–521, 1987.

[14] M. Zhu, M. Otte, P. Chaudhari, and E. Frazzoli, "Game theoretic controller synthesis for multi-robot motion planning part I: Trajectory based algorithms," in *Proc. 2014 IEEE Int. Conf. on Robot. and Automat.*, May 2014, pp. 1646–1651.

[15] D. K. Jha, M. Zhu, and A. Ray, "Game theoretic controller synthesis for multi-robot motion planning part II: Policy-based algorithms," in *5th IFAC Workshop Distrib. Estimation and Control in Netw. Syst.*, vol. 48, no. 22, 2015, pp. 168–173.

[16] F. Bullo, J. Cortés, and S. Martìnez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[17] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 12, p. 399, 2013.

[18] M. Zhu and S. Martínez, *Distributed Optimization-Based Control of Multi-Agent Networks in Complex Environments*. Springer, 2015.

[19] M. Cao, A. S. Morse, and B. D. Anderson, "Reaching a consensus in a dynamically changing environment: A graphical approach," *SIAM J. Control and Optim.*, vol. 47, no. 2, pp. 575–600, 2008.

[20] W. Ren and R. W. Beard, *Distributed consensus in multi-vehicle cooperative control*. Springer, 2008.

[21] D. Panagou and V. Kumar, "Cooperative visibility maintenance for leader–follower formations in obstacle environments," *IEEE Trans. Robot.*, vol. 30, no. 4, pp. 831–844, 2014.

[22] E. Frazzoli and F. Bullo, "Decentralized algorithms for vehicle routing in a stochastic time-varying environment," in *Proc. 43rd IEEE Conf. Decis. and Control*, vol. 4, 2004, pp. 3357–3363.

[23] J. Cortés, S. Martínez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. and Automat.*, vol. 20, no. 2, pp. 243–255, 2004.

[24] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, adaptive coverage control for networked robots," *Int. J. Robot. Res.*, vol. 28, no. 3, pp. 357–375, 2009.

[25] D. V. Dimarogonas, S. G. Loizou, K. J. Kyriakopoulos, and M. M. Zavlanos, "A feedback stabilization and collision avoidance scheme for multiple independent non-point agents," *Automatica*, vol. 42, no. 2, pp. 229–243, 2006.

[26] R. Olfati-Saber and R. M. Murray, "Distributed cooperative control of multiple vehicle formations using structural potential functions," in *the 15th IFAC World Congr.*, Jun. 2002, pp. 495–500.

[27] D. Panagou, "A distributed feedback motion planning protocol for multiple unicycle agents of different classes," *IEEE Trans. Autom. Control*, vol. 62, no. 3, pp. 1178–1193, 2017.

[28] P. Cardaliaguet, M. Quincampoix, and P. Saint-Pierre, "Set-valued numerical analysis for optimal control and differential games," in *Stochastic and Differ. Games*. Springer, 1999, pp. 177–247.

[29] G. Zhao and M. Zhu, "Pareto optimal multi-robot motion planning," *arXiv:1802.09099*, 2018.