Incorporating Control Barrier Functions in Distributed Model Predictive Control for Multirobot Coordinated Control

Chao Jiang, *Member, IEEE*, and Yi Guo, *Senior Member, IEEE*

***Abstract*—Multirobot motion planning and control has been investigated for decades and is still an active research area due to the growing demand for both performance optimality and safety assurance. This article presents an optimization-based method for the coordinated control of multiple robots with optimized control performance and guaranteed collision avoidance. We consider a group of differential drive wheeled robots, and design a distributed model predictive control (DMPC) where the team-level trajectory optimization is decomposed into subproblems solved by individual robots via alternating direction method of multipliers (ADMM). Our DMPC design utilizes a discrete-time control barrier functions method to develop control constraints that provide collision avoidance assurance. Compared to existing ADMM-based DMPC methods with Euclidean distance constraint for collision avoidance, our method ensures collision avoidance with minimal compromise of the optimality with respect to primary control objectives. We validated our method in a multirobot cluster flocking problem. The simulation results show effective coordinated control that achieves improved control performance and safety guarantees.**

***Index Terms*—Alternating direction method of multipliers (ADMM), control barrier functions (CBF), distributed model predictive control (DMPC), multirobot coordinated control.**

I. Introduction

MULTIROBOT self-organizing behavior emerges as robots coordinate their motion through local interactions to achieve team-level objectives. A plethora of control protocols have been investigated to facilitate the autonomy of multirobot systems [1]. A major control design paradigm focuses on decentralized feedback control [2], where the control protocols utilize local information acquired by each robot to compute an analytic control input in a reactive manner at each time step. These approaches provide provably correct control protocols that can be executed efficiently in real time. However, such feedback control approaches do not explicitly account for performance optimality, such as minimal control effort, minimal time usage, and minimal travel distance. Therefore, optimization-based approaches that plan robot trajectories for optimized performance ahead of time have been investigated in a large body of work [3].

Trajectory optimization is one of the most investigated trajectory planning methods, which solves for a state-control sequence that satisfies the desired motion and performance metrics specified by an objective function. Distributed multirobot trajectory optimization is a preferable framework that allows robots to concurrently compute their own trajectory based on the information acquired through local sensing and/or communication, and no central planner is needed. Distributed trajectory optimization problems have been approached with mixed-integer linear programming [4], sequential convex programming [5], [6], [7], [8], and distributed model predictive control (DMPC) [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21]. Notably, DMPC has become an appealing control scheme to address various multirobot planning problems owing to its capabilities of predicting future system outputs, handling state/control constraints, and replanning during online execution.

The aforementioned DMPC methods essentially entail solving a distributed optimization problem in an inner loop, where the centralized trajectory optimization at the team level is decomposed into subproblems that can be solved on individual robots in a distributed manner. The fundamental challenge of the decomposition is to address interrobot couplings through dynamics [9], [10], control objective function [11], [12], or operational constraints [13], [14], [20], [21], due to local interaction between robots. Among various decomposition techniques, alternating direction method of multipliers (ADMM) [22] is a well-recognized framework that combines the dual decomposition method with the method of multiplier that utilizes an augmented Lagrangian for better convergence property. ADMM scheme has been successfully used to solve a vast range of DMPC problems with different types of couplings [10], [12], [13], [15]. However, the performance of ADMM in practice is sensitive to the choice of the algorithm parameter associated with the quadratic penalty term that balances the convergence of the feasibility residual and the primal optimality specified by the objective function [23]. In the context of DMPC-based

Manuscript received 29 September 2022; revised 15 May 2023; accepted 23 June 2023. Date of publication 28 June 2023; date of current version 1 March 2024. The work of Chao Jiang was supported in part by the U.S. National Science Foundation under Grant IIS-2024813. The work of Yi Guo was supported in part by the U.S. National Science Foundation under Grant CMMI-1825709. Recommended by Associate Editor L. Pallottino. *(Corresponding author: Chao Jiang.)*

Chao Jiang is with the Department of Electrical Engineering and Computer Science, University of Wyoming, Laramie, WY 82071 USA (e-mail: cjiang1@uwyo.edu).

Yi Guo is with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: yguo1@stevens.edu).

Digital Object Identifier 10.1109/TCNS.2023.3290430

multirobot coordinated control, the convergence of feasibility residual is crucial for retaining the feasibility of original centralized problem and for the assurance of interaction constraints between robots. Prior studies suggested that a better convergence property for the feasibility residual can be attained by increasing the value of the penalty parameter, but at the cost of a loss in the primal optimality [9], [24], [25]. Thus, there is a need to improve the primal optimality while ensuring the convergence of feasibility residual to meet various forms of robot interaction constraints.

Among different robot interaction constraints, collision avoidance is the basic safety requirement for real-world multirobot coordinated control. Traditional DMPC formulations cast collision avoidance as a coupling constraint that confines the minimum Euclidean distance between a robot and its neighbors. The distance constraint under ADMM decomposition is strictly satisfied only if the feasibility residual converges. This is because the distance constraint only contains the robot state variables for which slack variables are introduced for decomposition, and thus, will only be enforced when optimizing for the slack variables. Therefore, a larger penalty parameter is needed so that more emphasis is placed on the residual minimization, leading to compromised optimality with respect to the primary control objectives. A related problem about enhancing feasibility and safety performance at the same time for MPC was investigated in [26], where slack variables with relaxing technique were introduced to find a good tradeoff between feasibility and safety.

Control barrier functions (CBFs) approach [27] is a principled framework broadly exploited to enforce safety constraints, such as collision avoidance for robotic systems [28]. By means of CBFs, one can derive control input conditions that assure a forward invariant set of safe states. These conditions then act as constraints in the synthesis of optimization-based safe controllers. For example, conditions expressed by CBFs have been used as safety filters via constrained quadratic programming [29], [30], [31], unified with control Lyapunonv functions in quadratic programs [32], [33], or incorporated in MPC [26], [34], [35], [36].

This article is concerned with the synthesis of distributed safe controller for a group of differential drive wheeled robots. We aim to achieve both improved primal optimality and guaranteed collision avoidance for multirobot coordinated control using ADMM-based DMPC. To this end, our idea is to impose a constraint on the robots' control input to produce collision-free reference positions over the prediction horizon. The control constraints is developed using discrete-time CBFs [37]. We then design a DMPC-based trajectory optimization method that incorporates the CBF constraints for collision avoidance. Our idea to use CBF constraints in MPC is similar to the aforementioned works [26], [34], [36]. While these works are concerned with single robot problems, in this article we propose an approach that incorporates CBFs in ADMM-based DMPC to enhance both safety and performance optimality in multirobot control. We show that our method yielded better performance optimality compared to traditional DMPC with distance constraint.

The main contributions of our work are twofold. First, we propose to use a discrete-time CBF method to derive the control constraints that ensure interrobot collision avoidance for ADMM-based DMPC methods. Second, unlike existing methods for multirobot collision avoidance [29], [30], [31] where CBFs are used as a safety filter [28] that confines the control input given by a nominal controller (usually a distributed feedback controller), our work employs trajectory optimization and incorporates the CBF constraint as a safety constraint to produce collision-free trajectories. While traditional ADMM-based DMPC methods may strive to balance the tradeoff between performance optimality and safety guarantee, our proposed method guarantees collision avoidance with minimal compromise of the optimality with respect to primary control objectives.

The rest of this article is organized as follows. Section II presents the multirobot coordinated control problem and the centralized MPC formulation. Section III presents the constraint design for collision avoidance using a discrete-time CBF method and the ADMM-based DMPC algorithm for real-time coordinated control. Simulation results and performance evaluation are provided in Section IV. Finally, Section V concludes this article.

II. Problem Formulation

A. Robot Kinematics

We study the coordinated motion control of a group of N nonholonomic robots with a two-wheel differential drive system. For each robot $i \in \{1, \ldots, N\}$, define the robot state as $\boldsymbol{x}_i = [x_i, y_i, \theta_i]^T \in \mathbb{R}^3$, which consists of the position $\boldsymbol{p}_i = [x_i, y_i]^T \in \mathbb{R}^2$ and the orientation $\theta_i \in \mathbb{R}$. The robot control input is defined as $\boldsymbol{u}_i = [v_{il}, v_{ir}]^T \in \mathbb{R}^2$ with v_{il} and v_{ir} being the velocity of the left wheel and right wheel of the ith robot, respectively. The discrete-time kinematics of the differential drive robot is then given by

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \Delta t \cdot \boldsymbol{G}(t) \cdot \boldsymbol{u}_i(t) \tag{1}$$

with Δt being the sampling period, and

$$\boldsymbol{G}(t) = \begin{bmatrix} \frac{\cos\theta_i(t)}{2} & \frac{\cos\theta_i(t)}{2} \\ \frac{\sin\theta_i(t)}{2} & \frac{\sin\theta_i(t)}{2} \\ -\frac{1}{l} & \frac{1}{l} \end{bmatrix}$$

where l is the distance between the robot's left and right wheels. The differential drive robot model (1) is nonlinear and underactuated, and it represents a large class of autonomous mobile robots [38], including three-wheel and four-wheel differential drive robots. Differential drive robots are one of the most popular autonomous mobile robots whose flocking control has broad application significance.

B. Proximity Graph

The communication connectivity of the robot team is defined by an undirected proximity graph, denoted by $\mathcal{G} = (V, E(t))$, with V being the set of vertices representing the robots $i \in \{1, \ldots, N\}$ and $E(t)$ being the set of edges representing the communication links at time step t. A communication link between

robot i and j is established if the Euclidean distance between the two robots is equal or less than a predefined communication range limit, denoted by d_c. Thus, the set of edges $E(t)$ is defined as

$$E(t) = \{(i,j) \mid \|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\| \leq d_c \ \forall i,j \in V, i \neq j\}. \quad (2)$$

A pair of robots (i,j) is said to be neighbors at time step t if $(i,j) \in E(t)$ and the set of neighbors of each robot i is defined as $\mathcal{N}_i(t) = \{j \mid (i,j) \in E(t)\}$. It is worth noting that the proximity graph is dynamic as both $E(t)$ and $\mathcal{N}_i(t)$ are time-varying when the robots move around.

C. Collision Avoidance

To ensure interrobot collision avoidance, it is required that each robot keeps a predefined minimal separation distance, denoted by d_s, from its neighboring robots. That is, the following inequality relation should be guaranteed at all times:

$$\|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\| \geq d_s \ \forall i \in V \ \forall j \in \mathcal{N}_i(t). \quad (3)$$

In optimization-based control approaches, collision avoidance defined by (3) can be procured in both state space and control input space [16]. The former usually uses (3) as constraints imposed on the Euclidean distance between two robots. The latter imposes constraints on the control input to keep the robots from being driven to collision-prone states. In this article, we employ a CBF method to derive constraints on control input for collision avoidance and demonstrate the advantage of our method over traditional methods that use the distance constraint (3).

D. Coordinated Control via Model Predictive Control

We consider a cluster flocking control problem of multirobot systems. We assume a reference trajectory consisting of time-varying reference states, defined as $\boldsymbol{x}_r(t)$, is assigned to all robots. The common reference trajectory can be given by a long-horizon motion planner, such as those based on differential dynamic programming or iterative LQR (a computationally efficient variant of DDP), as used in [39]. The rationale for using a reference trajectory is that MPC uses short horizon in a receding fashion to gain more robustness than long-horizon planning, but the resulting trajectory usually loses long-horizon optimality. An intuitive solution is to generate a reference trajectory that achieves good long-horizon performance and track the reference trajectory with receding horizon control, such as MPC [40].

The control objective of the cluster flocking is then for each robot to track the reference trajectory, $\boldsymbol{x}_r(t)$, while guaranteeing the minimal separation distance to avoid collision. Note that the defined cluster flocking behavior does not specify an explicit flocking formation. The cluster flocking control can be formulated as the following finite-horizon optimization problem:

$$\min_{\boldsymbol{X}(t),\boldsymbol{U}(t)} \quad J(t) \triangleq \sum_{i=1}^{N} \left(\sum_{k=1}^{M-1} w_t \|\boldsymbol{x}_i(t+k) - \boldsymbol{x}_r(t+k)\|^2 + w_f \|\boldsymbol{x}_i(t+M) - \boldsymbol{x}_r(t+M)\|^2 + w_u \|\boldsymbol{u}_i(t+k-1)\|^2 \right) \quad (4a)$$

$$\text{s.t.} \quad \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \Delta t \cdot \boldsymbol{G}(t) \cdot \boldsymbol{u}_i(t) \ \forall i \in V \quad (4b)$$

$$\boldsymbol{u}_i(t) \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}] \ \forall i \in V \quad (4c)$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i^0 \ \forall i \in V \quad (4d)$$

$$\|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\| \geq d_s \ \forall i \in V \ \forall j \in \mathcal{N}_i(t) \quad (4e)$$

where M is the prediction horizon and $\{\boldsymbol{X}(t), \boldsymbol{U}(t)\} \triangleq \{\boldsymbol{x}_i(t), \ldots, \boldsymbol{x}_i(t+M), \boldsymbol{u}_i(t), \ldots, \boldsymbol{u}_i(t+M-1)\}_{i=1}^{N}$ is the set of trajectories of all robots, starting from time step t. The first term in the objective function (4a) encapsulates the tracking error between the robot state $\boldsymbol{x}_i(t)$ and the reference state $\boldsymbol{x}_r(t)$; the second objective term is the terminal state tracking error of the fixed-horizon optimization; the last objective term encourages minimal control effort. The parameters w_t, w_f, and w_u are constant weights associated with the respective objective terms.

The optimization problem is subject to the kinematics constraint (4b), the control constraint (4c) representing the robot motors' upper and lower limits (i.e., the constants $\boldsymbol{u}_{\min}$ and $\boldsymbol{u}_{\max}$), and the distance constraint (4e) for interrobot collision avoidance. The robot state at time step t is assumed accessible and is given as $\boldsymbol{x}_i^0$. The MPC scheme exploits the system model (1) to predict future robot trajectories and solves for a state-control sequence for each robot with optimized control performance defined in (4a). At each decision time step t, the MPC measures the current state $\boldsymbol{x}_i(t)$, and compute the optimal trajectories $\{\boldsymbol{X}(t), \boldsymbol{U}(t)\}$, of all robots. Then, the first step of the computed control sequence will be executed to move the robots one step forward and new MPC computation will be performed repeatedly in a receding horizon fashion.

The problem (4) is a centralized formulation due to the coupling constraint (4e). Existing works have exploited ADMM framework to decompose the formulation (4) into subproblems, each of which can be solved by an individual robot in a distributed fashion. However, in traditional DMPC methods with distance constraint, the control performance measured by the objective function (4a) can be compromised for ensuring the distance constraint (4e) when solving the augmented Lagrangian in an ADMM scheme. Thus, in this article we propose a DMPC method with CBF constraints for collision avoidance and show that our method achieves improved control performance.

III. Method

In this section, we first present the constraint design for collision avoidance and the modified MPC method with CBF constraints. Then, we present the ADMM-based DMPC algorithm for decentralized coordinated control.

A. Constraint Design for Collision Avoidance

A safe set of each robot's state can be defined as such that the condition (3) is satisfied for all its neighbors. CBF methods provide a way to derive the conditions for robot control inputs under which a forward invariance of the safe set is guaranteed. Therefore, starting from a collision-free state, a robot is

guaranteed to stay in the safe set by executing the control inputs that satisfy the conditions.

1) CBF Condition Derivation: We define the safe set $\mathcal{S}$ that includes all collision-free states of each robot i as

$$\mathcal{S} = \{\boldsymbol{x}_i(t) \in \mathbb{R}^3 \mid B(\boldsymbol{x}_i(t)) \geq 0\} \tag{5}$$

where $B(\boldsymbol{x}_i(t)) : \mathbb{R}^3 \to \mathbb{R}$ is known as the discrete-time exponential barrier function [37]. To ensure collision avoidance, the control input $\boldsymbol{u}_i(t)$ should be synthesized such that the invariance of the safety set $\mathcal{S}$ is promised along the entire state sequence of robot i (i.e., $\boldsymbol{x}_i(t), \forall t \geq 0$), given that the initial state $\boldsymbol{x}_i(0) \in \mathcal{S}$ (i.e., $B(\boldsymbol{x}_i(0)) \geq 0$). The CBF conditions to ensure the invariance of $\mathcal{S}$ are then given as follows [37].

1) $B(\boldsymbol{x}(0)) \geq 0$.
2) There exists a control input $\boldsymbol{u}(t) \in \mathbb{R}^2$ such that $\Delta B(\boldsymbol{x}(t), \boldsymbol{u}(t)) + \gamma B(\boldsymbol{x}(t)) \geq 0, \forall t \geq 0$, where $\Delta B(\boldsymbol{x}(t), \boldsymbol{u}(t)) := B(\boldsymbol{x}(t+1)) - B(\boldsymbol{x}(t))$ and γ is a constant such that $0 < \gamma \leq 1$.

Given the definition of collision avoidance in (3), we select a barrier function as

$$B := \|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\|^2 - d_s^2 \ \forall j \in \mathcal{N}_i(t). \tag{6}$$

Substituting (6) in $\Delta B(\boldsymbol{x}_i(t), \boldsymbol{u}_i(t)) + \gamma B(\boldsymbol{x}_i(t))$, we have

$$\begin{aligned}
&\Delta B(\boldsymbol{x}_i(t), \boldsymbol{u}_i(t)) + \gamma B(\boldsymbol{x}_i(t)) \\
&\quad = B(\boldsymbol{x}_i(t+1)) - (1-\gamma) B(\boldsymbol{x}_i(t)) \\
&\quad = \|\boldsymbol{p}_i(t+1) - \boldsymbol{p}_j(t+1)\|^2 - d_s^2 - \|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\|^2 \\
&\qquad + d_s^2 + \gamma \left(\|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\|^2 - d_s^2\right) \\
&\quad \geq 0 \ \forall j \in \mathcal{N}_i(t).
\end{aligned} \tag{7}$$

Let $\boldsymbol{p}_i(t) = C \cdot \boldsymbol{x}_i(t)$, with $C = [\boldsymbol{I}_{2\times 2} \quad \boldsymbol{0}_{2\times 1}]$, we have

$$\boldsymbol{p}_i(t+1) = C \cdot \boldsymbol{x}_i(t+1) = C \cdot \boldsymbol{x}_i(t) + C \cdot \boldsymbol{G}(\theta_i(t)) \cdot \boldsymbol{u}_i(t).$$

Then, condition (7) can be written as (the time index t is omitted for simplicity in notation)

$$\begin{aligned}
&\|C \cdot \boldsymbol{x}_i + C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{x}_j - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\|^2 \\
&\quad - \|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2 + \gamma \left(\|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2 - d_s^2\right) \\
&= \|C \cdot \boldsymbol{x}_i - C \cdot \boldsymbol{x}_j + C\boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\|^2 \\
&\quad - \|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2 + \gamma \left(\|\boldsymbol{p}_i - \boldsymbol{p}_j\|^2 - d_s^2\right) \\
&= \|\boldsymbol{p}_{ij} + C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\|^2 \\
&\quad - \|\boldsymbol{p}_{ij}\|^2 + \gamma \left(\|\boldsymbol{p}_{ij}\|^2 - d_s^2\right) \\
&= \|\boldsymbol{p}_{ij}\|^2 + 2\boldsymbol{p}_{ij}^T \left(C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\right) - \|\boldsymbol{p}_{ij}\|^2 \\
&\quad + \|C \cdot \boldsymbol{G}(\boldsymbol{x}_i)\boldsymbol{u}_i - C \cdot \boldsymbol{G}(\boldsymbol{x}_j)\boldsymbol{u}_j\|^2 + \gamma \left(\|\boldsymbol{p}_{ij}\|^2 - d_s^2\right) \\
&= 2\boldsymbol{p}_{ij}^T \left(C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\right) + \gamma \left(\|\boldsymbol{p}_{ij}\|^2 - d_s^2\right) \\
&\quad + \|C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j\|^2 \\
&\geq 0 \ \forall j \in \mathcal{N}_i(t)
\end{aligned} \tag{8}$$

where $\boldsymbol{p}_{ij} = \boldsymbol{p}_i - \boldsymbol{p}_j$ is the relative position between robot i and j. Condition (8) can be used as an inequality constraint on robot i's control input $\boldsymbol{u}_i$, by which $B(\boldsymbol{x}_i(t)) \geq 0$ is satisfied along the entire robot trajectory. The scalar γ is a selectable parameter which dictates the extend to which $B(\boldsymbol{x}_i(t)) \geq 0$ is enforced. A larger γ allows the state to be closer to the boundary of the safe set $\mathcal{S}$.

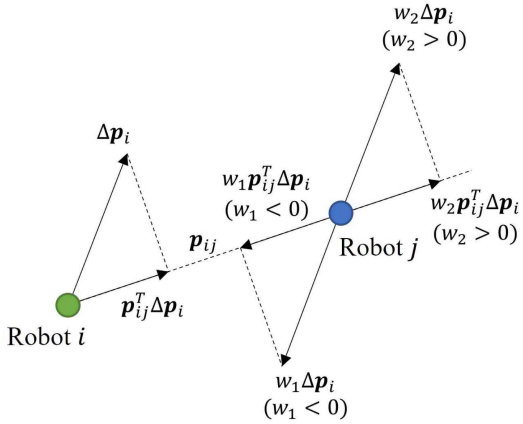


Fig. 1. Relative motion between robot i and robot j.

2) Modified CBF Condition: To evaluate the condition (8), each robot i will need to access the relative positions $\boldsymbol{p}_{ij}$, along with its neighbor's state $\boldsymbol{x}_j$ and control $\boldsymbol{u}_j$ via communication during decentralized control. To reduce communication burden, we consider a modified condition which can be evaluated by each robot without additional communication load.

It is worth noting that, the term $C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j$ in (8) can be interpreted as the displacement of robot j's position by executing the control $\boldsymbol{u}_j$ during the sampling period Δt. Our idea is to replace $C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j$ with $\omega_{ij} \cdot C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i$, i.e., the displacement of robot i during a sampling period, scaled by a factor ω_{ij}. The rationale behind the idea is the fact that the inner product $\boldsymbol{p}_{ij}^T \cdot C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j$ in (8) is the robot j's displacement projected onto the line connecting the positions of robot i and robot j, which captures the "tendency" of robot j to move closer to or further away from robot i.

Reasonable assumptions about the neighbor robots' motion can be made by tailoring the scalar ω_{ij}. Note that ω_{ij} is a real number whose magnitude and sign determines the assumed robot j's motion based on robot i's motion. Fig. 1 shows an example of the relative motion between robot i and robot j, given the assumption about robot j's motion. The robot i's displacement is given by $\Delta\boldsymbol{p}_i = \boldsymbol{p}_i(t+1) - \boldsymbol{p}_i(t)$. As it tends to move closer to robot j, the projection of robot i's displacement on $\boldsymbol{p}_{ij}$, i.e., $\boldsymbol{p}_{ij}^T\Delta\boldsymbol{p}_i$, points toward robot j. A negative ω_{ij} (i.e., ω_1) assumes that robot j will execute a movement such that the magnitude of displacement $\Delta\boldsymbol{p}_j$ is proportional to $\Delta\boldsymbol{p}_i$ and the direction of $\Delta\boldsymbol{p}_j$ is opposite to $\Delta\boldsymbol{p}_i$, showing a tendency to move closer to robot i. In contrast, a positive ω_{ij} (i.e., ω_2) assumes that robot j tends to move away from robot i. In fact, the selection of ω_{ij} reflects how conservative or aggressive robot i's assumption is about its neighbors "contribution" to the collective movement that would lead to a possible collision. Three examples of selecting ω_{ij}, which result in different motion behaviors of robot i, are given as follows.

1) For $\omega_{ij} = 0$, robot i assumes that robot j makes no movement in the direction of $\boldsymbol{p}_{ij}$. In this case, robot i's motion behavior is neither conservative nor aggressive.
2) For $\omega_{ij} = -1$, robot i assumes that the displacement of robot j in the direction of $\boldsymbol{p}_{ij}$ has the same magnitude as that of robot i but points to the opposite direction. In this case, robot i presents a conservative motion behavior when it tends to move toward robot j.
3) For $\omega_{ij} = 1$, robot i assumes that the displacement of robot j in the direction of $\boldsymbol{p}_{ij}$ has the same magnitude and direction as that of robot i. In this case, robot i presents an aggressive motion behavior when it tends to move toward robot j.

By replacing $C \cdot \boldsymbol{G}(\theta_j) \cdot \boldsymbol{u}_j$ with $\omega_{ij} \cdot C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i$, the condition (8) becomes

$$\begin{aligned} &2\boldsymbol{p}_{ij}^T \left(C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - \omega_{ij} C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i\right) + \\ &\|C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i - \omega_{ij} C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i\|^2 + \gamma \left(\|\boldsymbol{p}_{ij}\|^2 - d_s^2\right) \\ &= 2(1-\omega_{ij})\boldsymbol{p}_{ij}^T \cdot C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i + \gamma \left(\|\boldsymbol{p}_{ij}\|^2 - d_s^2\right) \\ &\quad + (1-\omega_{ij})^2 \|C \cdot \boldsymbol{G}(\theta_i) \cdot \boldsymbol{u}_i\|^2 \\ &\geq 0 \; \forall j \in \mathcal{N}_i(t). \end{aligned} \tag{9}$$

The information involving neighboring robots to evaluate (9) now only includes the relative position, $\boldsymbol{p}_{ij}$, thus imposing less communication load compared to (8). Moreover, evaluating (9) could require no interrobot communication if $\boldsymbol{p}_{ij}$ is estimated or measured directly.

B. Model Predictive Control With CBF

The CBF (9) is then used as an inequality constraint for avoiding interrobot collision, and the modified formulation of (4) is defined as

$$\min_{\boldsymbol{X}(t),\boldsymbol{U}(t)} \quad J(t) \triangleq \sum_{i=1}^{N} \left(\sum_{k=1}^{M-1} w_t \|\boldsymbol{x}_i(t+k) - \boldsymbol{x}_r(t+k)\|^2 + w_f \|\boldsymbol{x}_i(t+M) - \boldsymbol{x}_r(t+M)\|^2 + w_u \|\boldsymbol{u}_i(t+k-1)\|^2 \right) \tag{10a}$$

$$\text{s.t.} \quad \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \Delta t \cdot \boldsymbol{G}(t) \cdot \boldsymbol{u}_i(t) \; \forall i \in V \tag{10b}$$

$$\boldsymbol{u}_i(t) \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}] \; \forall i \in V \tag{10c}$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i^0 \; \forall i \in V \tag{10d}$$

$$g(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{x}_j) \leq 0 \; \forall i \in V, j \in \mathcal{N}_i(t) \tag{10e}$$

where $g(\boldsymbol{x}_i, \boldsymbol{u}_i, \boldsymbol{x}_j) \leq 0$ represents the inequality condition (9) derived using the CBF method for collision avoidance.

The feasible set of problem (10) is determined by the constraints (10b)–(10e) and is the intersection of the reachable set determined by (10b)–(10d) and the invariant set determined by (10e). The choice of γ in (10e) thus has an effect on the feasiblity of problem (10). A discussion on the effect of γ is given in [41].

C. Distributed MPC via ADMM

The problem (10) is a centralized formulation due to the coupling constraint (10e). We thus exploit the ADMM [22] framework to decompose (10) into subproblems that can be solved locally by each robot in a distributed manner. Various ADMM-based decomposition schemes (e.g., [10], [12], [13], [15]) have been proposed in literature. Our decomposition scheme is similar to that in [13], which is designed to tackle interrobot couplings imposed by operational constraints only. That is, each robot's dynamics and objective function are decoupled from other robots.

First, the coupling constraint (10e) is decoupled by introducing the slack variables, $\tilde{\boldsymbol{x}}_i \forall i \in V$, and $\tilde{\boldsymbol{x}}_{ij} \forall i \in V, j \in \mathcal{N}_i(t)$, and enforcing $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{ij}$ to be equal to $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, respectively. The resulting equivalent problem of (10) is defined as

$$\min_{\boldsymbol{X}(t),\boldsymbol{U}(t)} \quad J(t) \triangleq \sum_{i=1}^{N} \left(\sum_{k=1}^{M-1} w_t \|\boldsymbol{x}_i(t+k) - \boldsymbol{x}_r(t+k)\|^2 + w_f \|\boldsymbol{x}_i(t+M) - \boldsymbol{x}_r(t+M)\|^2 + w_u \|\boldsymbol{u}_i(t+k-1)\|^2 \right) \tag{11a}$$

$$\text{s.t.} \quad \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \Delta t \cdot \boldsymbol{G}(t) \cdot \boldsymbol{u}_i(t) \; \forall i \in V \tag{11b}$$

$$\boldsymbol{u}_i(t) \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}] \; \forall i \in V \tag{11c}$$

$$\boldsymbol{x}_i(t) = \boldsymbol{x}_i^0 \; \forall i \in V \tag{11d}$$

$$g(\tilde{\boldsymbol{x}}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_{ij}) \leq 0 \; \forall i \in V, j \in \mathcal{N}_i(t) \tag{11e}$$

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i \; \forall i \in V \tag{11f}$$

$$\tilde{\boldsymbol{x}}_{ij} = \boldsymbol{x}_j \; \forall i \in V \forall j \in \mathcal{N}_i(t) \tag{11g}$$

The slack variables $\tilde{\boldsymbol{x}}_i(t)$ and $\tilde{\boldsymbol{x}}_{ij}(t)$ can be considered as the robot i's hypothesis about its own state $\boldsymbol{x}_i(t)$ and the robot j's state $\boldsymbol{x}_j(t)$, respectively. An approximate solution to (11) can be found by solving the dual problem of (11). To reduce the duality gap, only the equality constraints (11f) and (11g) are dualized to form the dual problem of (11). The ADMM scheme solves the dual problem with an augmented Lagrangian function which is given by

$$\begin{aligned} \mathcal{L} = \sum_{i=1}^{N} J_i(t) + \sum_{k=1}^{M} \Bigg(&\lambda_i^T (\boldsymbol{x}_i(t+k) - \tilde{\boldsymbol{x}}_i(t+k)) \\ &+ \frac{\rho}{2} \|\boldsymbol{x}_i(t+k) - \tilde{\boldsymbol{x}}_i(t+k)\|^2 \\ &+ \sum_{j \in \mathcal{N}_i(t)} \Big(\lambda_{ij}^T (\boldsymbol{x}_j(t+k) - \tilde{\boldsymbol{x}}_{ij}(t+k)) \\ &+ \frac{\rho}{2} \|\boldsymbol{x}_j(t+k) - \tilde{\boldsymbol{x}}_{ij}(t+k)\|^2 \Big) \Bigg) \end{aligned} \tag{12}$$

where $J_i(t)$ is the primary control objective of each robot defined in (11a), and λ_i, λ_{ij} are the dual variables associated with the

dualized constraints (11f) and (11g). We rewrite (12) with a simplified notation as follows:

$$\begin{aligned}\mathcal{L} &= \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum_{j\in\mathcal{N}_i} \mathcal{L}_{ij}(\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_{ij}, \lambda_{ij}) \\ &= \sum_{i=1}^{N} \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum_{j\in\mathcal{N}_i} \mathcal{L}_{ji}(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_{ji}, \lambda_{ji}) \qquad (13)\end{aligned}$$

with $\mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i)$ being the term involving robot i only and $\mathcal{L}_{ij}(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_{ij}, \lambda_{ij})$ being the term involving both robot i and j. The second equality in (13) holds due to the symmetry of bidirectional interaction [42] with the robot j's hypothesis about the robot i's state $\tilde{\boldsymbol{x}}_{ji}$ and the associated multiplier λ_{ji}. Then, the augmented Lagrangian (12) can be solved by the following ADMM steps recursively.

Step 1:

$$\begin{aligned}\min_{\boldsymbol{x}_i, \boldsymbol{u}_i} \quad & \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum_{j\in\mathcal{N}_i} \mathcal{L}_{ji}(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_{ji}, \lambda_{ji}) \\ \text{s.t.} \quad & \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{G}(t)\cdot \boldsymbol{u}_i(t)\ \forall i \in V \\ & \boldsymbol{u}_i(t) \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}]\ \forall i \in V \\ & \boldsymbol{x}_i(t) = \boldsymbol{x}_i^0\ \forall i \in V \\ & g(\tilde{\boldsymbol{x}}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_{ji}) \le 0\ \forall i \in V\ \forall j \in \mathcal{N}_i(t). \qquad (14)\end{aligned}$$

Step 2:

$$\begin{aligned}\min_{\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_{ij}} \quad & \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum_{j\in\mathcal{N}_i} \mathcal{L}_{ij}(\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_{ij}, \lambda_{ij}) \\ \text{s.t.} \quad & g(\tilde{\boldsymbol{x}}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_{ij}) \le 0\ \forall i \in V\ \forall j \in \mathcal{N}_i(t). \qquad (15)\end{aligned}$$

Step 3:

$$\begin{aligned}&\lambda_i = \lambda_i + \rho(\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i)\ \forall i \in V \\ &\lambda_{ij} = \lambda_{ij} + \rho(\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_{ij})\ \forall i \in V\ \forall j \in \mathcal{N}_i(t). \qquad (16)\end{aligned}$$

In step 1, each robot solves for its own trajectory over the fixed horizon such that the primary control objective $J_i(t)$ as well as the cost of its state $\boldsymbol{x}_i(t)$ deviating from the hypotheses $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{ji}$ are minimized. In step 2, each robot updates its hypotheses $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{ij}$ such that the differences between $\boldsymbol{x}_i$ and $\tilde{\boldsymbol{x}}_i$, $\boldsymbol{x}_j$ and $\tilde{\boldsymbol{x}}_{ij}$ are minimized. The multipliers λ_i and λ_{ij} are updated in step 3. Computations in steps 1–3 are fully decoupled and can be performed simultaneously on each robot in a distributed and parallel manner.

Remark 1 (Comparison with DMPC using distance constraints): In traditional MPC formulations with distance constraint, the centralized problem [e.g., problem (4)] is decomposed using the same ADMM scheme. The resulting ADMM procedure is similar to (14)–(16) except that the distance constraint, $\|C\tilde{\boldsymbol{x}}_i - C\tilde{\boldsymbol{x}}_{ij}\| \ge d_s$, is used in stead of $g(\tilde{\boldsymbol{x}}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_{ij}) \le 0$. Moreover, since the distance constraint, $\|C\tilde{\boldsymbol{x}}_i - C\tilde{\boldsymbol{x}}_{ij}\| \ge d_s$, only contains the slack variables $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{ij}$, it is excluded from (14) and is only incorporated in (15). As a result, the distance constraint is only imposed in solving for the local hypotheses $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{ij}$, and the robot trajectory $\{\boldsymbol{x}_i, \boldsymbol{u}_i\}$ satisfying the distance constraint is ensured by minimizing the deviation from the hypotheses in the ADMM step (14). The selection of the parameter ρ in the augmented Lagrangian (12) is therefore crucial as it should balance minimizing the primary control objective, $J_i(t)$, and minimizing the deviation to conform to the distance constraint. In contrast, our MPC formulation with CBF constraint results in an inequality constraint in the ADMM step (14). The collision avoidance is enforced with a hard constraint on the control input in addition to the minimum deviation objective. This way, the parameter ρ can be set to a smaller number so that the algorithm can focus more on the primary control objective in (12) without violating the constraint, leading to better tracking performance.

For each robot to perform the ADMM steps (14)–(16), communication between neighboring robots will be required for local information exchange. Specifically, step (14) requires each

Algorithm 1: Distributed MPC via ADMM.

Input : Initial $\boldsymbol{x}_i^0$, $\tilde{\boldsymbol{x}}_i$, $\tilde{\boldsymbol{x}}_{ij}$, λ_i, λ_{ij};
Output: Planned trajectory $\{\boldsymbol{x}_i(t+k), \boldsymbol{u}_i(t+k-1)\}_{k=1}^{M}$ over the MPC horizon M;

while *not converge* **do**
  Communicate with neighboring robots $j \in \mathcal{N}_i$: Send $\tilde{\boldsymbol{x}}_{ij}$, λ_{ij} and receive $\tilde{\boldsymbol{x}}_{ji}$, λ_{ji};
  Compute trajectory starting from current state $\boldsymbol{x}_i^0$ by solving (14): $\{\boldsymbol{x}_i, \boldsymbol{u}_i\} \leftarrow \arg\min_{\boldsymbol{x}_i, \boldsymbol{u}_i} \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum \mathcal{L}_{ji}(\boldsymbol{x}_i, \tilde{\boldsymbol{x}}_{ji}, \lambda_{ji})$;
  Communicate with neighboring robots $j \in \mathcal{N}_i(t)$: Send $\boldsymbol{x}_i$ and receive $\boldsymbol{x}_j$;
  Compute $\tilde{\boldsymbol{x}}_i$, $\tilde{\boldsymbol{x}}_{ij}$ by solving (15): $\{\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_{ij}\} \leftarrow \arg\min_{\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_{ij}} \mathcal{L}_i(\boldsymbol{x}_i, \boldsymbol{u}_i, \tilde{\boldsymbol{x}}_i, \lambda_i) + \sum \mathcal{L}_{ij}(\boldsymbol{x}_j, \tilde{\boldsymbol{x}}_{ij}, \lambda_{ij})$;
  Update λ_i, λ_{ij} using (16): $\lambda_i = \lambda_i + \rho(\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i)$, $\lambda_{ij} = \lambda_{ij} + \rho(\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_{ij}), \forall j \in \mathcal{N}_i(t)$;
end

Algorithm 2: Motion Control via DMPC.

Input : Reference trajectory $[\boldsymbol{x}_r(0), \boldsymbol{x}_r(1), ..., \boldsymbol{x}_r(T)]$;
Output: Robot control $[\boldsymbol{u}_i(0), \boldsymbol{u}_i(1), ..., \boldsymbol{u}_i(T)], \forall i \in V$;

Initialize the states of all robots $\boldsymbol{x}_i(0) = \boldsymbol{x}_i^{\text{init}}$;
for $t = 0, ..., T$ **do**
  for *robot* $i = 1, ..., N$ **do**
    Measure the current robot state $\boldsymbol{x}_i(t)$;
    Solve the fixed horizon optimization (10): $\{\boldsymbol{x}_i(t+k), \boldsymbol{u}_i(t+k-1)\}_{k=1}^{M} \leftarrow$ call Algorithm 1;
    Apply the first step of the computed control sequence $\boldsymbol{u}_i(t)$ at the current state $\boldsymbol{x}_i(t)$;
  end
end

robot i to acquire the hypotheses $\tilde{\boldsymbol{x}}_{ji}$ and the multipliers λ_{ji} from all neighbors $j \in \mathcal{N}_i(t)$, and meanwhile send $\tilde{\boldsymbol{x}}_{ij}$ and λ_{ij} to all neighbors; step (15) requires each robot to acquire the optimized trajectory $\boldsymbol{x}_j$ from all neighbors and send the optimized trajectory $\boldsymbol{x}_i$ to all neighbors. The ADMM-based DMPC algorithm performed on each robot is summarized in Algorithm 1.

During real-time coordinated motion control, Algorithm 1 will be executed by each robot at every time step in a receding horizon manner, and only the first step of the solved control sequence will be executed. The real-time coordinated motion control via DMPC is summarized in Algorithm 2. As an instance of ADMM framework, Algorithm 1 inherits the convergence property of ADMM. However, the convergence to the exact solution of the centralized formulation may require considerable ADMM iterations, imposing computation and communication load for real-time control. Therefore, in practice, the iterations in Algorithm 1 can be terminated if the residual, $\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_{ij}$, is smaller than a predefined tolerance or a maximum number of iterations is reached.

IV. Simulation and Results

A. Simulation Setup and Parameters

The proposed method was validated in Matlab simulations. For the problem defined in (11), the following simulation setup and parameters are chosen. The number of robots $N = 5$. The robots' positions were randomly initialized in an 8-by-8-m square area. The initial positions of any two robots were at least 2-m apart. The initial robot orientations were randomly selected from $[0, 2\pi)$. The communication range limit d_c was set to 2.5 m. The minimal separation distance d_s was set to 0.8 m. The robot control limits were set as $\boldsymbol{u}_{\min} = -1.2$ m/s and $\boldsymbol{u}_{\max} = 1.2$ m/s. The parameter l in (1) was set to 0.4 m. Simulation update rate was set to 5 Hz, i.e., $\Delta t = 0.2$ s. All simulation experiments were run for 120 time steps, i.e., $T = 120$.

The algorithm parameters were selected as follows. The prediction horizon $M = 10$. The weights in the objective function were set as $w_t = 1$, $w_f = 10$, $w_u = 0.5$, respectively. The Lagrangian multipliers λ_i and $\lambda_{ij} \forall i, j \in V, i \neq j$ were initialized to 0.1. The penalty parameter was chosen as $\rho = 0.6$. The CBF parameter ω_{ij} and γ were set to -0.5 and 0.8, respectively. The number of ADMM iterations K was set to 2 to lessen the computational burden. For all simulation experiments, the constrained nonlinear programming of the ADMM algorithm was solved using MATLAB Nonlinear Optimization toolbox (fmincon). All simulation experiments were run on a computer with an Intel Core i7-12700 K (3.6 GHz) CPU and a RAM of 16 GB.

B. Simulation Results

An example result of robot trajectories for the multirobot coordinated control is shown in Fig. 2, along with the robot positions at $t = 0$ s, $t = 8$ s, $t = 16$ s, and $t = 24$ s. The dotted lines represent the communication connection between robots. We can see that the robots successfully tracked the reference trajectory (represented by the black dashed line) and kept the minimal separation distance to their neighbors. Fig. 3(a) shows the time history of tracking error $\|\boldsymbol{x}_i(t) - \boldsymbol{x}_r\|$ for each robot and the average tracking error of all robots. One can observe that all tracking errors converged and the average tracking error, as shown by the solid black trace, converged to 0.8 m. To verify that the interrobot collision avoidance is guaranteed, we show the time history of interrobot distance in Fig. 3(b). It can be seen that the interrobot distances were all greater than the predefined minimal separation distance (as shown by the red dashed line) at all times, indicating that no collision occurred during the control process. The black dashed line represents the communication range limit. The time history of robot control inputs is shown in Fig. 3(c). All robot motor speed control inputs converged to the speed determined by the reference trajectory after about 14 s. To show the convergence of the feasibility residual $\boldsymbol{x}_i - \tilde{\boldsymbol{x}}_i$ and $\boldsymbol{x}_j - \tilde{\boldsymbol{x}}_{ij}$ of the ADMM algorithm, we define $c(t)$ that measures the residual at each time step t as

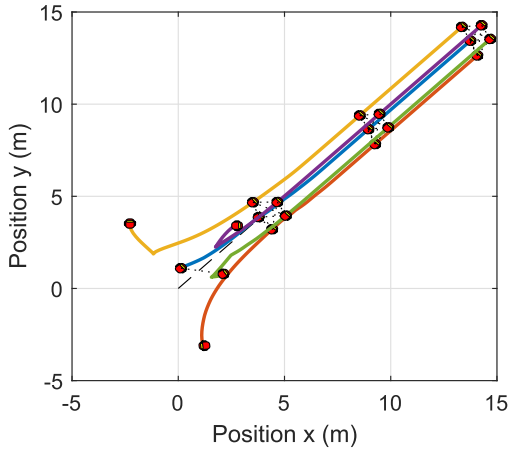


Fig. 2. Robot trajectories with robot positions at $t = 0$ s, 8 s, 16 s, and 24 s. The dashed black line represents the reference trajectory and the dotted lines represent inter-robot communication links.

$$c(t) = \frac{1}{M}\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{M}\left(\|\boldsymbol{x}_i(t+k) - \tilde{\boldsymbol{x}}_i(t+k)\|^2 + \sum_{j\in\mathcal{N}_i(t)}\|\boldsymbol{x}_j(t+k) - \tilde{\boldsymbol{x}}_{ij}(t+k)\|^2\right). \quad (17)$$

The residual $c(t)$ was averaged over the number of robots N and the prediction horizon M, and the residual after two ADMM iterations (as the maximum ADMM iterations K at each time step was set to 2) at each time step was recorded. Fig. 3(d) shows the time history of $c(t)$ on a logarithmic scale. One can see that the residual converged to 0.046 m^2. Our approach optimized the tracking error [as shown in Fig. 3(a)] while ensuring the convergence of the feasibility residual.

1) Effect of Parameter γ: We conducted simulations with varying selections of the parameter γ to show its effect on the control performance. The parameter was set as $\gamma = 1$, $\gamma = 0.8$, $\gamma = 0.5$, $\gamma = 0.25$, respectively. For each parameter selection, 200 simulation runs with randomized initial robot states were

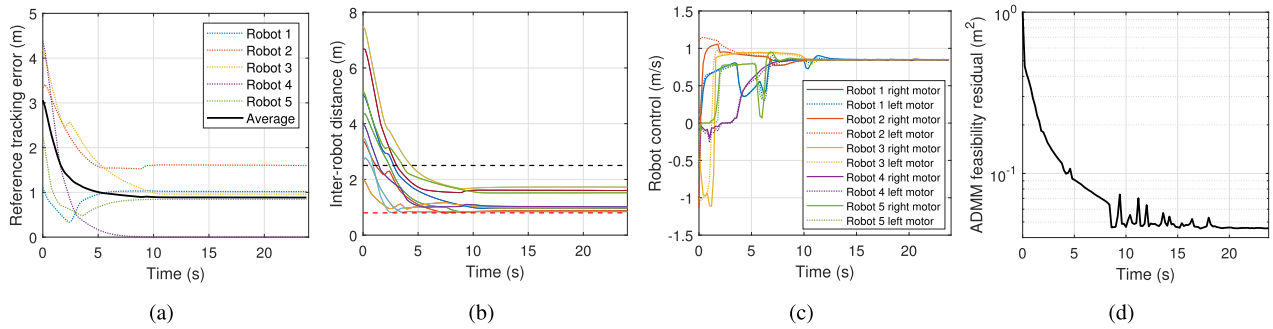


Fig. 3. Sample result of the multirobot coordinated control using our DMPC method with CBF constraint for collision avoidance. (a) Tracking error; (b) inter-robot distance; (c) robot control input; and (d) ADMM feasibility residual $c(t)$ on a logarithmic scale. The red and black dashed lines in (b) represent the minimal separation distance, d_s, and the communication range limit, d_c, respectively.

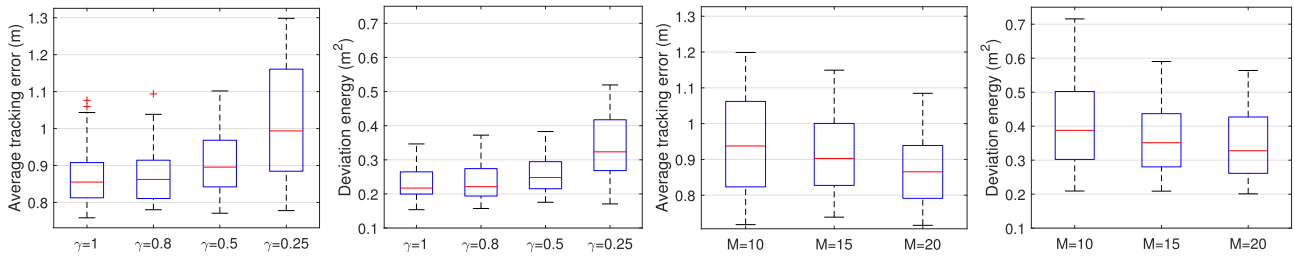


Fig. 4. Statistic results for the comparison between the DMPC method with CBF constraint and the DMPC method with distance constraint for different parameters selections. (a) and (b) Average tracking error and the deviation energy, respectively, of the DMPC method with CBF constraint. (c) and (d) Average tracking error and the deviation energy, respectively, of the DMPC with distance constraint. The comparison results show that our DMPC-CBF method achieved better tracking accuracy than the DMPC-DC method when γ was appropriated selected. The DMPC-DC method could achieve comparable tracking performance with our method using a larger prediction horizon M. (a) DMPC-CBF. (b) DMPC-CBF. (c) DMPC-DC. (d) DMPC-DC.

performed to obtain the statistic results. We use two metrics to evaluate the control performance. The first metric measures the average tracking error of all robots, i.e.,

$$\bar{e}(t) = \frac{1}{N}\sum_{i\in V}\|\boldsymbol{x}_i(t) - \boldsymbol{x}_r(t)\|. \tag{18}$$

The minimization of (18) is enforced by the first two terms in the objective function (4a). The second metric is the deviation energy [43] which is defined as

$$e_d(t) = \frac{1}{E(t)+1}\sum_{i\in V}\sum_{j\in\mathcal{N}_i}(\|\boldsymbol{p}_i(t) - \boldsymbol{p}_j(t)\| - d_s)^2. \tag{19}$$

The deviation energy measures how far the interrobot distance of neighboring robots deviates from the minimal separation distance d_s. It should be noted that keeping the minimal separation distance is necessary but not sufficient for successfully minimizing the average tracking error of the group.

The statistical results of average tracking error and deviation energy with different values of γ are shown in Fig. 4(a) and (b). Each data point in the statistical results is the average tracking error or deviation energy of all robots at the final simulation step, i.e., $t = 24$ s. The red line in each box marks the median, the box edges represent the 25th and 75th percentiles, respectively. The dashed vertical lines extend to the max/min, and the cross markers are outliers.

We can see from Fig. 4(a) that $\gamma = 1$ yielded best tracking performance. Then, the tracking error increased as γ became smaller, and deteriorated dramatically when $\gamma = 0.25$. From Fig. 4(b) we observe that the deviation energy increased as γ decreased from 1 to 0.25. These observations align with the analysis in Section III that a smaller γ creates a smaller feasible region, resulting in control inputs that keep the robots apart with larger interrobot distances. Although $\gamma = 1$ yielded the best tracking performance, it is worth mentioning that it is advised to choose a slightly smaller value (e.g., $\gamma = 0.8$) in practice. This is because the constraint with $\gamma = 1$ explores the boundary of the safe set $\mathcal{S}$ and its forward invariance may be violated due to uncertainties in real systems.

C. Comparison With Euclidean Distance Constraint

In this subsection, we present the performance comparison between our DMPC method with CBF (DMPC-CBF) and the DMPC method with distance constraint (DMPC-DC). The DMPC-DC method solves the centralized problem defined in (4) by decomposing it into subproblems using the same ADMM decomposition scheme as [13]. Unlike [13], however, the DMPC-DC for comparison uses Euclidean distance rather than

separating hyperplane for collision avoidance and does not adopt spline parameterization for robot trajectories. The equivalent problem after decomposition for DMPC-DC is similar to (11) except that the constraint (11e) is replaced by the distance constraint (4e). As mentioned in Section III-C, the distance constraint is only incorporated in the ADMM step (15) and is excluded from step (14).

Since the performance of DMPC-DC and DMPC-CBF is affected by multiple algorithm parameters, we first compare the two methods without considering the effect of the ADMM parameter ρ (which is discussed in Section IV-C1). We implemented the DMPC-DC method with three different values of prediction horizon M, i.e., $M = 10$, $M = 15$, and $M = 20$ and our DMPC-CBF method with $\gamma = 1$, $\gamma = 0.8$, $\gamma = 0.5$, and $\gamma = 0.25$. The penalty parameter ρ was set to 1 for the DMPC-DC method and 0.6 for our DMPC-CBF method. The other parameters were the same for both methods. It should be noted that we chose different values of ρ for DMPC-DC and DMPC-CBF, respectively, in order to obtain the best results of the two methods for comparison. As presented in Section IV-C1, we found from empirical evaluations that $\rho = 1$ and $\rho = 0.6$ yielded best performance for DMPC-DC and DMPC-CBF, respectively, under the metrics (18) and (19) without violating the safety constraint. The DMPC-DC and DMPC-CBF methods with each selection of M or γ were simulated 200 times with varying initial states. The performance was compared using the metrics defined in (18) and (19).

The statistical results of the DMPC-DC method are shown in Fig. 4(c) and (d). We observe from Fig. 4(c) that the DMPC-DC method with a larger prediction horizon M yielded better tracking performance. This is because a larger prediction horizon allows the robots to coordinate their motions for an optimal configuration that achieves minimum overall tracking error well ahead of time before collision becomes imminent. As shown in Fig. 4(d), the deviation energy also tended to decrease when the prediction horizon became larger. Comparing the average tracking errors shown in Fig. 4(a) and (c), we can see that with the same prediction horizon, $M = 10$, our DMPC-CBF method outperformed the DMPC-DC method when γ was set to 0.5 or larger. The DMPC-DC method requires a larger prediction horizon to achieve comparable tracking performance to our DMPC-CBF method. For example, the DMPC-DC with $M = 15$ achieved similar median tracking error to our DMPC-CBF method with $\gamma = 0.5$, but with higher variance. However, a larger prediction horizon M imposes more computational load for each MPC computation iteration.

1) Effect of Parameter ρ: To verify the analysis on the effect of the penalty parameter ρ in Section III-C, we evaluated both methods with three values of the penalty parameter: $\rho = 0.6$, $\rho = 0.8$, and $\rho = 1$. For comparison, the centralized MPC (4) with distance constraint and the centralized MPC (10) with CBF constraint were implemented as baseline methods. Each centralized method and decentralized method with a selected value of ρ was simulated 200 times with random initial states. We compared their performance using average tracking error, deviation energy, and success rate, i.e., the percentage of successful runs out of 200. A successful run is defined as such that the minimum separation distance d_s is guaranteed during the entire simulation.

The evaluation results are summarized in Table I, where the mean tracking error and the mean deviation energy with a standard deviation of all successful runs are shown. One can see that both centralized MPC methods yielded similar performance that outperformed the decentralized methods, DMPC-CBF and DMPC-DC. Both decentralized methods with $\rho = 1$ successfully ensured the minimum distance d_s for all 200 runs and achieved comparable mean tracking error. When ρ was reduced to 0.8, the DMPC-DC method only achieved a success rate of 80%, while our DMPC-CBF method still achieved a success rate of 100%. Moreover, with a smaller value of ρ, the performance of our DMPC-CBF was improved with a smaller mean tracking error. When ρ was further reduced to 0.6, none of the DMPC-DC runs was successful, while our DMPC-CBF method could obtain further improved tracking performance with a success rate of 98%. The 2% runs of the DPMC-CBF method for $\rho = 0.6$ were unsuccessful as violations of the minimum interrobot distance constraint occurred at some time steps. The violations at those time steps were a result of large feasibility residuals after the maximum ADMM iterations were reached. The evaluation results validate that the traditional DMPC based on ADMM needs to select an appropriate value for ρ such that the robot trajectories strictly conform to the distance constraint. However, the tracking performance is sacrificed when choosing a larger ρ to satisfy the constraint. In contrast, our DMPC-CBF method can obtain better tracking performance by choosing a smaller ρ with high success rates of guaranteeing the minimum interrobot distance.

D. Discussion

1) Relation Between CBF Constraint and Distance Constraint: With centralized implementation, the MPC method with CBF constraint yielded similar performance to the MPC method with distance constraint when the parameter γ approaches the upper bound of 1 (as indicated by the results in the last two columns of Table I). Indeed, as explained in [37] and [41], when $\gamma = 1$ the CBF constraint and the distance constraint are almost the same except that the CBF constraint is imposed to the next prediction step rather than the current one. Similar performance of the two methods was also observed in their decentralized implementation when the penalty parameter $\rho = 1$. However, the DMPC-DC method failed to satisfy the distance constraint when ρ became smaller. Therefore, using the CBF constraint gains advantages in the DMPC schemes as it allows us to choose a smaller value of ρ to reduce the primal optimality gap while ensuring the constraint.

2) Primal Suboptimality: Our DMPC-CBF method underperformed its centralized counterpart with regard to the tracking error, as shown in Table I. This is expected considering that 1) a limited number of ADMM iterations was performed per time step to lessen the computation load, but at the cost of losing the optimality of the resulting solutions; 2) the modified CBF condition (9) may make conservative assumption of the neighboring robots' movement by selecting a smaller ω_{ij} to ensure the minimal separation distance. As a result, the robots

TABLE I
PERFORMANCE COMPARISON BETWEEN DMPC-CBF AND DMPC-DC WITH DIFFERENT VALUES OF ρ

	$\rho = 1$		$\rho = 0.8$		$\rho = 0.6$		Centralized MPC-CBF (baseline)	Centralized MPC-DC (baseline)
	DMPC -CBF	DMPC -DC	DMPC -CBF	DMPC -DC	DMPC -CBF	DMPC -DC		
Success rate (%)	100	100	**100**	80	**98**	0	100	100
Tracking error (m)	**0.95 ±0.10**	0.96 ±0.12	**0.90 ±0.08**	0.93 ±0.14	**0.87 ±0.07**	N/A	0.64 ±0.03	**0.63 ±0.02**
Deviation energy (m^2)	**0.36 ±0.09**	0.41 ±0.15	**0.27 ±0.08**	0.35 ±0.12	**0.24 ±0.05**	N/A	0.13 ±0.03	0.13 ±0.03

* N/A represents cases where results are not available as no successful run was obtained.
Values in bold typeface are the best result between DMPC-CBF and DMPC-DC, or between centralized MPC-CBF and centralized MPC-DC on each metric.

tended to keep a larger distance from their neighbors, thus increasing the overall tracking error of the group. The parameter ω_{ij} determines the robots' collision avoidance behavior (i.e., aggressive or conservative) [44] and needs to be selected to balance the tradeoff between tracking performance and safety assurance. Both control performance and safety would benefit from adjusting ω_{ij} adaptively at runtime based on prediction of neighboring robots' motion so that an appropriate ω_{ij} is always selected at each time step.

3) Extension to High-Order Dynamics: The proposed DMPC-CBF method relies on the robot dynamics (1) whose relative degree [45] with respect to the CBF (6) is one. For high-order system dynamics, the relative degree of the system would be greater than one. In this case, the control input $\boldsymbol{u}_i$ does not appear in the CBF condition (8), and thus, safe control can not be synthesized. To extend our approach to systems of higher relative degree, the discrete-time high-order CBF method (a generalization of high-order CBFs for continuous-time systems [46]) proposed in [47] could be exploited. One challenge in solving MPC with high-order CBF constraints is guaranteeing feasibility. Therefore, a feasibility enforcement method [48] needs to be designed to produce feasible solutions while satisfying safety constraints.

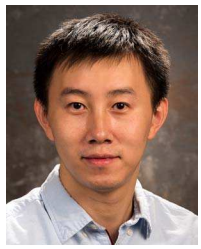
V. CONCLUSION

In this work, we proposed an ADMM-based DMPC for coordinated motion control of multirobot system with collision avoidance. Unlike existing DMPC methods that use Euclidean distance constraints, our method leverages the discrete-time CBFs method to develop the constraints that confine the robot control for collision avoidance. The proposed method allows us to choose a smaller penalty parameter for the ADMM algorithm to focus more on the primal optimality without violating the constraints for collision avoidance, and thus improves the performance with respect to the primary control objectives. We validated and evaluated our method in a multirobot cluster flocking problem and the simulation results show effective coordinated control that achieves improved control performance and safety guarantees. In our future work, we will extend our method to more complex dynamics, such as high-order dynamics, as discussed in [33] and [48], or general nonlinear dynamics.

REFERENCES

[1] Y. Guo, *Distributed Cooperative Control: Emerging Applications*. Hoboken, NJ, USA: Wiley, 2017.
[2] J. Cortés and M. Egerstedt, "Coordinated control of multi-robot systems: A survey," *SICE J. Control, Meas., Syst. Integration*, vol. 10, no. 6, pp. 495–503, 2017.
[3] L. E. Beaver and A. A. Malikopoulos, "An overview on optimal flocking," *Annu. Rev. Control*, vol. 51, pp. 88–99, 2021.
[4] Y. Kuwata and J. P. How, "Cooperative distributed robust trajectory optimization using receding horizon MILP," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 2, pp. 423–431, Mar. 2011.
[5] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 1917–1922.
[6] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 5954–5961.
[7] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot navigation in formation via sequential convex programming," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 4634–4641.
[8] D. Morgan, G. P. Subramanian, S.-J. Chung, and F. Y. Hadaegh, "Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming," *Int. J. Robot. Res.*, vol. 35, no. 10, pp. 1261–1285, 2016.
[9] S. K. Mulagaleti, R. Van Parys, and G. Pipeleers, "Distributed model predictive control of multiple vehicles transporting a flexible payload," in *Proc. Eur. Control Conf.*, 2018, pp. 1417–1422.
[10] W. Tang and P. Daoutidis, "Distributed nonlinear model predictive control through accelerated parallel ADMM," in *Proc. Amer. Control Conf.*, 2019, pp. 1406–1411.
[11] W. B. Dunbar and R. M. Murray, "Distributed receding horizon control for multi-vehicle formation stabilization," *Automatica*, vol. 42, no. 4, pp. 549–558, 2006.
[12] T. H. Summers and J. Lygeros, "Distributed model predictive consensus via the alternating direction method of multipliers," in *Proc. Annu. Allerton Conf. Commun., Control, Comput.*, 2012, pp. 79–84.
[13] R. Van Parys and G. Pipeleers, "Distributed MPC for multi-vehicle systems moving in formation," *Robot. Auton. Syst.*, vol. 97, pp. 144–152, 2017.
[14] L. Dai, Q. Cao, Y. Xia, and Y. Gao, "Distributed MPC for formation of multi-agent systems with collision avoidance and obstacle avoidance," *J. Franklin Inst.*, vol. 354, no. 4, pp. 2068–2085, 2017.
[15] Y. Lyu, J. Hu, B. M. Chen, C. Zhao, and Q. Pan, "Multivehicle flocking with collision avoidance via distributed model predictive control," *IEEE Trans. Cybern.*, vol. 51, no. 5, pp. 2651–2662, May 2021.
[16] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 604–611, Apr. 2020.

[17] J. Zhan and X. Li, "Flocking of multi-agent systems via model predictive control based on position-only measurements," *IEEE Trans. Ind. Inform.*, vol. 9, no. 1, pp. 377–385, Feb. 2013.
[18] H.-T. Zhang, Z. Cheng, G. Chen, and C. Li, "Model predictive flocking control for second-order multi-agent systems with input constraints," *IEEE Trans. Circuits Syst. I, Regular Papers*, vol. 62, no. 6, pp. 1599–1606, Jun. 2015.
[19] F. Rey, Z. Pan, A. Hauswirth, and J. Lygeros, "Fully decentralized ADMM for coordination and collision avoidance," in *Proc. Eur. Control Conf.*, 2018, pp. 825–830.
[20] M. Farina and S. Misiano, "Stochastic distributed predictive tracking control for networks of autonomous systems with coupling constraints," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 3, pp. 1412–1423, Sep. 2018.
[21] I. Notarnicola and G. Notarstefano, "Constraint-coupled distributed optimization: A relaxation and duality approach," *IEEE Trans. Control Netw. Syst.*, vol. 7, no. 1, pp. 483–492, Mar. 2020.
[22] S. Boyd et al., "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.
[23] S. Lee, N. Chatzipanagiotis, and M. M. Zavlanos, "A distributed augmented Lagrangian method for model predictive control," in *Proc. IEEE Annu. Conf. Decis. Control*, 2017, pp. 2888–2893.
[24] Y. Ouyang, Y. Chen, G. Lan, and E. Pasiliao Jr., "An accelerated linearized alternating direction method of multipliers," *SIAM J. Imag. Sci.*, vol. 8, no. 1, pp. 644–681, 2015.
[25] S. Novoth, Q. Zhang, K. Ji, and D. Yu, "Distributed formation control for multi-vehicle systems with splitting and merging capability," *IEEE Control Syst. Lett.*, vol. 5, no. 1, pp. 355–360, Jan. 2021.
[26] J. Zeng, Z. Li, and K. Sreenath, "Enhancing feasibility and safety of nonlinear model predictive control with discrete-time control barrier functions," in *Proc. IEEE Conf. Decis. Control*, 2021, pp. 6137–6144.
[27] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. Eur. Control Conf.*, 2019, pp. 3420–3431.
[28] F. Ferraguti et al., "Safety and efficiency in robotics: The control barrier functions approach," *IEEE Robot. Automat. Mag.*, vol. 29, no. 3, pp. 139–151, Sep. 2022.
[29] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 661–674, Jun. 2017.
[30] T. Ibuki, S. Wilson, J. Yamauchi, M. Fujita, and M. Egerstedt, "Optimization-based distributed flocking control for multiple rigid bodies," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1891–1898, Apr. 2020.
[31] Y. Chen, A. Singletary, and A. D. Ames, "Guaranteed obstacle avoidance for multi-robot operations with limited actuation: A control barrier function approach," *IEEE Control Syst. Lett.*, vol. 5, no. 1, pp. 127–132, Jan. 2021.
[32] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. Autom. Control*, vol. 62, no. 8, pp. 3861–3876, Aug. 2017.
[33] G. Wu and K. Sreenath, "Safety-critical control of a planar quadrotor," in *Proc. Amer. Control Conf.*, 2016, pp. 2252–2258.
[34] K. P. Wabersich and M. N. Zeilinger, "Predictive control barrier functions: Enhanced safety mechanisms for learning-based control," *IEEE Trans. Autom. Control*, vol. 68, no. 5, pp. 2638–2651, May 2023.
[35] Z. Li, J. Zeng, A. Thirugnanam, and K. Sreenath, "Bridging model-based safety and model-free reinforcement learning through system identification of low dimensional linear models," in *Proc. Robot.: Sci. Syst.*, 2022.
[36] A. Thirugnanam, J. Zeng, and K. Sreenath, "Safety-critical control and planning for obstacle avoidance between polytopes with control barrier functions," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 286–292.
[37] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Proc. Robot., Sci. Syst.*, vol. 13, 2017, pp. 1–10.
[38] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. Cambridge, MA, USA: MIT Press, 2011.
[39] C. Jiang and Y. Guo, "Multi-robot guided policy search for learning decentralized swarm control," *IEEE Control Syst. Lett.*, vol. 5, no. 3, pp. 743–748, Jul. 2020.
[40] T. Zhang, G. Kahn, S. Levine, and P. Abbeel, "Learning deep control policies for autonomous aerial vehicles with MPC-guided policy search," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 528–535.
[41] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *Proc. Amer. Control Conf.*, 2021, pp. 3882–3889.
[42] R. Van Parys and G. Pipeleers, "Online distributed motion planning for multi-vehicle systems," in *Proc. Eur. Control Conf.*, 2016, pp. 1580–1585.
[43] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, Mar. 2006.
[44] U. Borrmann, L. Wang, A. D. Ames, and M. Egerstedt, "Control barrier certificates for safe swarm behavior," *IFAC-PapersOnLine*, vol. 48, no. 27, pp. 68–73, 2015.
[45] M. Sun and D. Wang, "Initial shift issues on discrete-time iterative learning control with system relative degree," *IEEE Trans. Autom. Control*, vol. 48, no. 1, pp. 144–148, Jan. 2003.
[46] W. Xiao and C. Belta, "High-order control barrier functions," *IEEE Trans. Autom. Control*, vol. 67, no. 7, pp. 3655–3662, Jul. 2022.
[47] Y. Xiong, D.-H. Zhai, M. Tavakoli, and Y. Xia, "Discrete-time control barrier function: High-order case and adaptive case," *IEEE Trans. Cybern.*, vol. 53, no. 5, pp. 3231–3239, May 2023.
[48] S. Liu, J. Zeng, K. Sreenath, and C. A. Belta, "Iterative convex optimization for model predictive control with discrete-time high-order control barrier functions," in *Proc. Amer. Control Conf.*, 2023, pp. 3368–3375.



Chao Jiang (Member, IEEE) received the B.S. degree in measuring and control technology and instrumentation from Chongqing University, Chongqing, China, in 2009, and the Ph.D. degree in electrical engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2019.

He was a Research Assistant with the Key Laboratory of Optoelectronic Technology and Systems, Chongqing University, Ministry of Education, China, from 2009 to 2012. In 2019, he joined the Department of Electrical Engineering and Computer Science, University of Wyoming, Laramie, WY, USA, where he is currently an Assistant Professor. His research interests include autonomous robots, multiple and distributed robotic systems, human–robot interaction, robotic learning, deep reinforcement learning.

Dr. Jiang was the recipient of the Innovation and Entrepreneurship Doctoral Fellowship at Stevens Institute of Technology, from 2012 to 2016, and the Outstanding Doctoral Dissertation Award in Electrical Engineering at Stevens Institute of Technology in 2019.

Yi Guo (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from the Xi'an University of Technology, Xi'an, China, in 1992 and 1995, respectively, and the Ph.D. degree in electrical engineering from the University of Sydney, Camperdown, NSW, Australia, in 1999.

She was a Postdoctoral Research Fellow with the Oak Ridge National Laboratory, Oak Ridge, TN, USA, from 2000 to 2002, and a Visiting Assistant Professor with the University of Central Florida, Orlando, FL, USA, from 2002 to 2005. Since 2005, she has been with the Stevens Institute of Technology, Hoboken, NJ, USA, where she is currently the Thomas E. Hattrick Chair Professor. She has authored one book (Wiley) and edited one book (Springer) and authored or coauthored more than 130 peer-reviewed journal and conference papers. Her research interests include autonomous mobile robotics, human–robot interaction, and distributed sensor networks.

Dr. Guo currently serves as the Editor-in-Chief for the IEEE ROBOTICS AND AUTOMATION MAGAZINE. She is Distinguished Lecturer of the IEEE Robotics and Automation Society. She has served in organizing committees of the IEEE International Conference on Robotics and Automation, in 2006, 2008, 2014, 2015, and 2023, and the IEEE/RSJ International Conference on Intelligent Robots and Systems, in 2019 and 2020.