IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

Deadlock-Aware Control for Multi-Robot Coordination with Multiple Safety Constraints

Zhenwei Zhang, *Member, IEEE,* Yuhao Zhang, *Member, IEEE,* Xingwei Zhao, Bo Tao, *Member, IEEE,* and Han Ding, *Senior Member, IEEE*

***Abstract*—Multi-robot coordination in shared workspaces is prone to deadlocks, which can compromise operational capabilities and task efficiency. Accurately determining the timing and spatial locations of deadlocks is essential for effective resolution, yet remains challenging due to dynamic robot interactions and growing system complexity. To this end, a distributed deadlock-aware control framework is proposed for robots to detect and avoid deadlocks while maintaining safe task execution. First, deadlocks are characterized by analyzing undesired equilibria in robot dynamics under safety constraints imposed by multiple stacked control barrier functions (CBFs). Our analysis reveals two critical properties: 1) Deadlocks occur at intersections of all active CBF boundaries, and 2) Deadlocks arise when robot stabilizing force are confined within the conical hull formed by active safety forces. These theoretical insights underpin a new detection method that identifies potential deadlocks from conflicts between safety requirements and task objectives. Furthermore, a reactive deadlock avoidance method is designed to help robots escape and prevent entry into potential deadlock regions by adaptively modulating the stabilizing force. A generalized workflow is established to systematically address deadlocks across various multi-robot tasks. Simulation and hardware experiments are conducted on robots collaborating in dense environments to validate the framework's effectiveness in preventing task failures caused by deadlocks.**

***Index Terms*—Deadlock detection and avoidance, control barrier functions, multi-robot systems, reactive motion planning.**

I. Introduction

MULTI-ROBOT coordination leverages local interactions to achieve collective behaviors, enabling a range of advanced applications including collaborative manufacturing [1], wild exploration and rescue [2], and intelligent warehousing [3]. Robotic controllers are designed to fulfill high-level objectives such as area coverage [4] and formation maintenance [5], while strictly enforcing collision avoidance and other safety constraints. However, conflicts between task objectives and safety requirements can severely compromise cooperative performance. For example, in multi-robot material-handling systems, deadlocks arise when robots mutually block each other's paths toward their target locations, leading to operational standstills [6]. If not promptly resolved, such local deadlocks can trigger chain reactions, potentially paralyzing the entire system.

This work was supported by the National Natural Science Foundation of China under Grant 52188102, Grant 62293514, Grand 52422501 and Grant 91948301. *(Corresponding authors: Bo Tao, Xingwei Zhao)*

The authors are with the State Key Laboratory of Intelligent Manufacturing Equipment and Technology, Department of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China. (e-mail:taobo@hust.edu.cn, zhaoxingwei@hust.edu.cn).

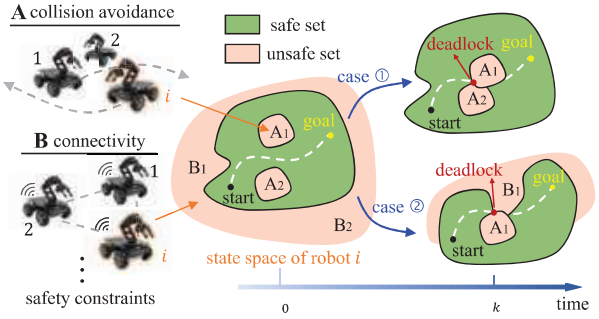


Fig. 1. Deadlocks are defined as undesired equilibria in robot dynamics. (Left) Various safety constraints are mapped to unsafe sets in the robot state space. (Right) The robot converges to the boundaries of the unsafe sets and encounters a deadlock. As robots dynamically interact with each other, these boundaries are constantly changing, making it difficult to detect deadlocks.

Deadlock detection is fundamental to implementing strategies for resolving deadlocks. Existing methods primarily identify deadlocks through the geometric configurations of robots in physical space [7], [8]. For instance, in collision avoidance of multiple mobile robots, directional deadlocks occur when robots block each other's paths due to alignment [9]. Resolving deadlocks requires designing coordination strategies tailored to the enumerated deadlock configurations [10]–[12]. However, such methods have limitations. On the one hand, deadlocks arise from dynamic interactions among multiple robots, often requiring global monitoring to identify when and where they will occur. This approach can be costly in terms of communication and computation, especially as the number of robots increases [13]. On the other hand, when coordination tasks involve diverse safety constraints or robots with complex physical structures, the geometric features of deadlocks become more complicated. For example, when a formation of robots approaches large obstacles, they may split into two groups to pass on either side [14]. This can lead to deadlocks because of the conflicting objectives between moving forward and maintaining formation connectivity. In such cases, the combinatorial complexity arising from different formation shapes and obstacle sizes makes it difficult to fully describe the deadlock configurations in physical space.

To address these challenges, this work defines deadlocks as undesired equilibria in robot dynamics and tackles them in state space. From this perspective, impermissible robot configurations induced by different safety constraints are mapped to unsafe sets in state space, and robots must reach their goal states while remaining within the safe set, as shown in Fig. 1. The concept of undesired equilibria implies a balanced state

where any progress towards their goals would intrude into the unsafe set. For velocity-controlled robots, the only viable option is to halt, resulting in a deadlock. This perspective unifies different forms of deadlocks and converts the detection of collective deadlocks into an analysis of individual dynamics, thereby reducing the combinatorial complexity.

Guided by this idea, we use multiple stacked Control Barrier Functions (CBFs) to encode various safety constraints in multi-robot coordination into unified forward invariance requirements of a state set [15]. To accomplish task goals, we consider two approaches that can be integrated with CBFs. The first involves adding a Control Lyapunov Function (CLF) constraint, guiding the robot to move in the direction of decreasing the CLF [16]–[18]. The second approach uses CBFs as a safety filter, which minimally adjusts a reference control input to maintain safety [19]–[21]. By incorporating CBFs with either CLF constraints or nominal controllers into Quadratic Programming (QP), optimal control that balances safety and performance can be achieved. However, since stability is sacrificed in events that conflict with safety, undesired equilibria occur when any progress toward the goal violates the safety constraints [22].

In order to fully identify deadlocks during robot interactions, we analyze the existence conditions of undesired equilibria in closed-loop robot dynamics using tools from duality theory. The theoretical results show that deadlocks arise from the conical combination relationship between the robot's stabilizing force and the safety forces generated by active CBF constraints. In addition, deadlocks are located at the intersection of the boundaries of these active CBFs. Utilizing these features, robots can evaluate the risk of deadlocks in real time based on local information. To avoid deadlocks, robots dynamically modulate the stabilizing force by altering the shape of CLF. The stabilizing force can be instantaneously adjusted to help robots escape potential deadlock regions or continuously adapted to prevent them from entering such regions. Finally, a general workflow for resolving deadlocks is proposed and validated in different multi-robot coordination tasks through simulations and experiments. The contributions of this article can be summarized as follows:

1) The deadlock characteristics are derived by analyzing the undesired equilibria induced by a QP controller with multiple general CBFs in robot dynamics. The theoretical results identify the conditions and locations of deadlock occurrences in multi-robot coordination, considering different safety constraints and task objectives.
2) An online deadlock detection method is proposed by evaluating the relationship between stabilizing force and the conical hull formed by active safety forces, granting robots deadlock awareness. Using the conical hull, robots can fully evaluate current deadlock risk and identify potential deadlock boundaries based on local information.
3) A reactive deadlock avoidance method is designed in state space to help robots escape from and prevent entry into identified potential deadlock regions by adjusting the stabilizing force. This method can address different types of deadlocks without relying on the geometric properties of deadlock configurations.
4) A deadlock-aware control framework is established by incorporating deadlock detection and avoidance methods as an isolated plugin into CBF-QP controllers. It inherits safety assurance provided by CBF constraints and can address various multi-robot deadlocks in a distributed manner without modifying the underlying theory.

The article is structured as follows. Related work is presented in Section II. Section III provides the preliminaries on CBF and formulates the problems. Section IV analyzes deadlock properties in a QP-based controller with multiple CBFs. Section V presents the deadlock-aware control framework. Section VI outlines a general workflow for deadlock resolution. Section VII compares different methods from both theoretical and simulation perspectives. Section VIII details the experimental results. Finally, Section IX concludes the article. The appendix contains proofs for several lemmas.

II. Related Work

Many researchers have focused on addressing deadlock in multi-robot coordination due to its critical impact on system ability and efficiency [23], [24]. In many applications, the main cause of deadlock is the conflict between collision avoidance and achieving task goals. The robots block each other's paths, forcing them to stop in place. Such deadlocks can be detected by analyzing the relative positions of the robots in physical space. However, as the number of robots increases, the number of geometric arrangements of deadlocks can grow exponentially. To reduce the complexity, many approaches use discrete environmental representations, such as grid maps [25] or topological networks [26]. This discretization converts continuous spatial reasoning into finite cell transitions, effectively reducing computational load by limiting possible robot configurations. Since the robots' motion is also discretized, typical deadlock patterns (e.g., cyclic deadlocks) can be accurately identified [27]. Some methods model robot behavior as discrete state transition systems and detect deadlocks by analyzing the evolution of these systems [12], [28]. These methods are particularly effective in structured environments, such as highways or warehouses, where robot paths can be easily segmented.

Resolving deadlocks usually requires designing coordination strategies for the identified deadlock configurations, which can be categorized into coupled and decoupled methods. The former relies on centralized planning based on knowledge of the environment and the states of all robots. The main methods include protocol-based search [29], conflict-based search [27], and graph-based search [30]. These methods are typically complete, meaning they can simultaneously plan collision-free paths for all robots to reach their targets. However, the computational cost increases exponentially with the number of robots [31]. Decoupled methods are incomplete but computationally faster than coupled methods, as each robot's path is planned individually, and deadlocks are resolved a posteriori [32], [33]. The main techniques, such as priority assignment [34], setting detour points [35], and motion delays [36], coordinate robot trajectories to avoid deadlocks. Since these methods are heuristic, the success in resolving

deadlocks also depends on the quality of the pre-planned paths. In [37], the authors incorporated a deadlock avoidance strategy as a penalty term in multi-robot trajectory optimization to directly generate deadlock-free trajectories. However, these coordination methods depend on synchronized clocks and precise execution in both space and time.

Another decoupled method allows robots to make real-time adjustments based on locally observable states, such as the positions and speeds of neighboring robots. Compared to carefully designed trajectories, this approach reduces coordination complexity and synchronization requirements and is widely used in dynamic and unknown environments [38], [39]. However, without a supervisor, it is difficult to detect deadlocks in advance. As a result, these methods typically address only local collisions and resolving deadlocks requires additional strategies like the right-hand rule [9] or consistent perturbations [40]. To enhance foresight, multi-hop communication is used for robots to acquire the intentions of distant neighboring robots [41]. The work in [42] divided the permissible motion areas of robots into non-overlapping Voronoi cells, allowing deadlocks to be predicted based on the positional relationships of neighboring robots at cell boundaries.

Overall, existing deadlock resolution methods primarily focus on the geometric properties of deadlocks and are specifically d esigned f or m ulti-robot c ollision a voidance. In fact, deadlocks are common in other multi-robot coordination tasks. In [43], deadlocks were observed during experiments on cooperative surveillance tasks, where aerial swarms failed to cover the most valuable areas as they attempted to avoid overlapping viewpoints. Another example is formation navigation, where robots can become deadlocked due to conflicts in maintaining connectivity, avoiding collisions, and reaching designated locations. In [14], an effective solution was to block deadlock-prone areas in the global path planner. These tasks introduce abstract task goals and safety constraints, making it more complicated to detect and avoid deadlocks based on their geometric arrangements in physical space.

Undesired equilibria in robot dynamics provide a more precise interpretation of deadlock. Some prior work has inspired the ideas and methods presented in this work. Reis et al. [22] proved that even with a single CBF constraint, controllers can lead to undesired stable equilibria when the vectors guiding system convergence are collinear with those ensuring safety. Recent work [44] derived similar conditions in robotic tracking applications, where an auxiliary force was introduced to avoid collinear conditions and enhance performance. The concept of region of attraction was introduced in [45]. It represents a region where control can be exerted to ensure the system's safety and stability. Following this idea, Mestres et al. [46] introduced a penalty term to eliminate undesired equilibria and provided an estimation of the region of attraction. Other discussions on undesired equilibria can be found in [47], [48]. These works adjust the parameters in improved controllers to eliminate or limit undesired equilibria.

Despite these works achieving theoretical breakthroughs in avoiding undesired equilibria, they mainly focus on a single CBF constraint. In multi-robot coordination, each robot faces multiple safety constraints simultaneously, such as collision avoidance with neighbors. Consequently, a common practice is to stack multiple CBF constraints within the QP [49], [50]. Compared to single CBF scenarios, this work demonstrates that deadlock characteristics fundamentally differ due to the combined effect of multiple CBFs, which expands the deadlock region rather than relying on collinearity conditions as in [22], [44]. While employing similar analytical tools to Grover et al. [51], [52], their focus was on analyzing the geometric properties of robot deadlock configurations. Therefore, they mainly discussed scenarios with no more than three robots. In contrast, our method derives deadlock characteristics directly from general CBF constraints in the state space, making it applicable to different task types and robot numbers.

III. Preliminaries and Problem Formulation

Notation: We use the notation $\nabla f(x)$ to represent the gradient $\frac{\partial f}{\partial x}$ of a differentiable function $f \in \mathbb{R}$ with respect to x. For a set of functions $\mathcal{F} = \{f_1, \cdots, f_M\}$, we define $\nabla\mathcal{F} = \{\nabla f_i(x) \mid f_i \in \mathcal{F}\}$. The notation $L_f V$ denotes the Lie derivative of a scalar-valued differentiable function V along the vector fields f, that is, $L_f V = \nabla V^T f$. The union of sets $\mathcal{A}_i$ is denoted by $\bigcup_{i=1}^{M} \mathcal{A}_i$, and the intersection is denoted by $\bigcap_{i=1}^{M} \mathcal{A}_i$. The notation $Cone(\mathcal{T})$ represents the conical hull generated by the vectors in the set $\mathcal{T} = \{\tau_1, \tau_2, \cdots\}$, i.e. $Cone(\mathcal{T}) = \{\sum_i \lambda_i \tau_i | \lambda_i \geq 0, \tau_i \in \mathcal{T}\}$. Let $|\mathcal{S}|$ be the cardinality of a set $\mathcal{S}$. The symbol $[M]$, where $M \in \mathbb{N}$, represents the set $\{1, 2, \ldots, M\}$.

In this section, we review CLF-CBF-based QPs for safe task execution, define the concept of deadlock, and outline the problems addressed in this work. Consider a control-affine system that follows dynamics:

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where $x \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u \in \mathcal{U} \subseteq \mathbb{R}^m$ are the state and control input vectors, with $\mathcal{X}$ and $\mathcal{U}$ being the admissible state and input sets. The functions $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are locally Lipschitz continuous. We assume that $g(x)$ is full rank for all values of $x \in \mathcal{X}$.

Definition 1 (CLF [17]): A positive definite function V is a control Lyapunov function (CLF) for system (1) if it satisfies:

$$\inf_{u\in\mathcal{U}} \{L_f V(x) + L_g V(x) u\} \leq -\gamma(V(x)), \tag{2}$$

where $\gamma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a class $\mathcal{K}$ function that is strictly increasing with $\gamma(0) = 0$.

This definition implies the existence of a stabilizing control that ensures the CLF decreases at any point outside the origin. Many robotics applications, such as navigation [16], trajectory planning [53], and formation control [54], can be encoded using CLFs.

The concept of safety is defined by the zero superlevel set of a continuously differentiable function $h : \mathbb{R}^n \to \mathbb{R}$:

$$\mathcal{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\}. \tag{3}$$

We refer to $\mathcal{C}$ as a safe set. The boundary and interior of set $\mathcal{C}$ are defined by $\partial\mathcal{C} = \{x \in \mathbb{R}^n : h(x) = 0\}$ and $\text{Int}(\mathcal{C}) = \{x \in \mathbb{R}^n : h(x) > 0\}$. The system is considered safe as long as the set $\mathcal{C}$ is forward-invariant, i.e. if $x(0) \in \mathcal{C}$ then $x(t) \in \mathcal{C}, \forall t \geq 0$. One way to render the set $\mathcal{C}$ forward-invariant for the system (1) is to use CBFs in the controller design.

Definition 2 (CBF [17]): T he f unction $h : \mathcal{X} \to \mathbb{R}$ i s a control barrier function (CBF) for system (1) if it satisfies

$$\sup_{u\in\mathcal{U}}\{L_f h(x) + L_g h(x)u\} \geq -\alpha(h(x)), \tag{4}$$

where $\alpha : \mathbb{R} \to \mathbb{R}$ is an extended class $\mathcal{K}$ function that is strictly increasing with $\alpha(0) = 0, \lim_{r\to\pm\infty} \alpha(r) = \pm\infty$.

Consider multiple safety constraints encoded by M CBFs $h_i, i \in [M]$. The control inputs set $\mathcal{U}_s$ that guarantee the safety of system (1) is defined by

$$\mathcal{U}_s = \{u \in \mathcal{U} : L_f h_i(x)+L_g h_i(x)u \geq -\alpha_i(h_i(x)), \forall i \in [M]\}. \tag{5}$$

We assume that the M CBFs are compatible, i.e. $\mathcal{U}_s \neq \emptyset$. A safety-critical controller that guides the system to the origin can be implemented by incorporating CLF and multiple CBF constraints into a QP. Additionally, the controller can be synthesized in a minimally invasive manner by using a nominal controller $\bar{u}$ as a reference input.

Definition 3 (CLF-MCBF-QP controller): The convergence and multiple safety constraints of system (1) can be simultaneously achieved through an optimal control u^* obtained from the following CLF-MCBF-QP:

$$\min_{(u,\delta)\in\mathbb{R}^m\times\mathbb{R}} \quad \frac{1}{2}\|u - \bar{u}(x)\|^2 + \frac{1}{2}p\delta^2 \tag{6a}$$

$$s.t. \quad L_f V(x) + L_g V(x)u + \gamma(V(x)) \leq \delta \tag{6b}$$

$$L_f h_i(x) + L_g h_i(x)u + \alpha_i(h_i(x)) \geq 0 \tag{6c}$$

$$i \in [M],$$

where relaxation variable δ in the CLF constraint (6b) relaxes convergence objective and ensures the feasibility of the QP.

The control $u^*(x)$ is designed to maintain safety while guiding the system towards convergence. However, if a conflict arises between the convergence objective and safety constraints, the system may halt to prevent any safety violations. Consider the closed-loop system

$$\dot{x} = f_{cl}(x) := f(x) + g(x)u^*(x). \tag{7}$$

The equilibrium condition of the closed-loop system (7) is $f_{cl}(x_{eq}) = 0$. From a control perspective, deadlock occurs when the system converges to an undesired equilibrium.

Definition 4 (Deadlock): An equilibrium x_{eq} is a deadlock point of the system (7) if $f_{cl}(x_{eq}) = 0$ and $V(x_{eq}) \neq 0$.

An example illustrates the deadlock phenomenon in which an integrator is controlled to the origin using the CLF-MCBF-QP controller but becomes trapped in a deadlock point, as depicted in Fig. 2. A potential deadlock region, surrounding the deadlock point, can be observed, comprising states that ultimately converge to the deadlock point. While this case involves static unsafe regions, in multi-robot coordination, the unsafe regions for each robot dynamically change and vary due to their movements toward different tasks. Consequently, the potential deadlock regions that robots may encounter are also dynamic and distinct. To reduce the complexity of this problem, this work aims to endow each robot with a deadlock-aware capability so that collective deadlocks can be resolved through individual reactive interactions. Specifically, this work addresses the following set of problems:

Problem 1 (deadlock detection): Given system in (1) and the CLF-MCBF-QP controller in (6), detect potential deadlock regions in real-time based on the system's current state, $x(t)$.

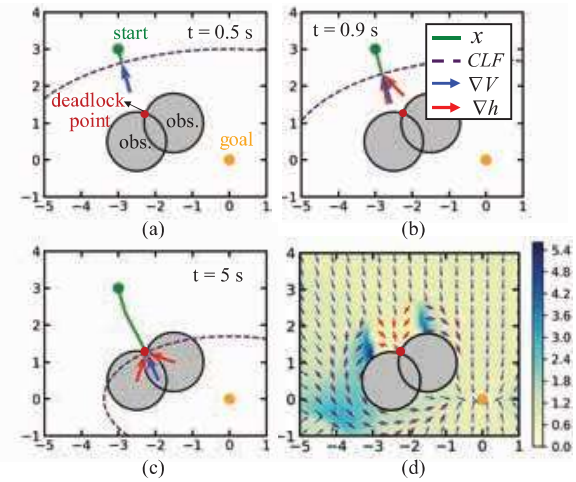


Fig. 2. Illustration of deadlock with a 2-D integrator controlled by a CLF-MCBF-QP controller in the presence of two unsafe sets. The simulation setup is given in Section IV-A. (a-c) Trajectories of the system leading to a deadlock point. (d) Velocity vector diagram of the system at different states. Different levels of deadlock risk are marked by vectors of different colors, following the method in Section IV-B. The red arrow indicates *weak deadlock*, the blue arrow indicates *completely deadlock-free*, and the purple arrow indicates *restricted deadlock-free*.

Problem 2 (deadlock avoidance): Given the system in (1) and multiple dynamic CBFs, generate control inputs that prevent the system from stabilizing at deadlock points while adhering to the CBF constraints.

Problem 3 (multi-robot deadlock resolution): Based on the proposed deadlock detection and avoidance methods, analyze and resolve deadlock issues in different multi-robot coordination tasks with multiple safety constraints.

IV. Deadlock Analysis in CBF-Based Controller

In this section, we analyze the characteristics of deadlock under multiple safety constraints. Building on this analysis, we propose a new deadlock detection method and discuss feasible deadlock avoidance strategies.

A. Characteristics of Deadlocks

We now present key conclusions on deadlock by analyzing the undesired equilibria in the closed-loop dynamics governed by the CLF-MCBF-QP controller (6). When any of the inequalities (6b) or (6c) becomes an equality, the corresponding constraint is considered active, and the associated CLF or CBF is said to be active. Let the set of all active CBFs be denoted by $\mathcal{H}$. Let $\mathcal{E}$ represent the set of deadlock points. Without loss of generality, assume that the system's goal is the origin and lies inside the safety set. Moreover, assume that the reference input $\bar{u}$ guides the system to converge towards the goal along the CLF, without considering safety constraints. Specifically, this implies $L_f V(x)+L_g V(x)\bar{u} \leq 0$, with equality holding if and only if $x = 0$. For the sake of simplicity, the dependency on state x is omitted in the following formulation.

Theorem 1: The deadlocks in the closed-loop dynamics resulting from applying the controller (6) to system (1) satisfy the following properties:

i. A necessary condition for deadlocks to occur is that the negation of stabilizing force F_V must belong to the conical hull formed by the active safety force F_{h_i}, i.e.

$$\underbrace{f + g\bar{u} - p\gamma(V)G\nabla V}_{F_V} + \sum_i \lambda_i \underbrace{G\nabla h_i}_{F_{h_i}} = 0, \quad (8)$$

where $G = gg^T \in \mathbb{R}^{n\times n}$, $\lambda_i \geq 0$ and $h_i \in \mathcal{H}$.

ii. The deadlocks are located at the intersection of all active CBF boundaries, i.e., $\mathcal{E} \subseteq \bigcap \partial\mathcal{C}(h_j)$, for $h_j \in \mathcal{H}$.

Proof: We can use the Karush-Kuhn-Tucker (KKT) conditions to analyze the properties of system (1) in deadlock. The Lagrangian associated to the controller (6) is

$$\begin{aligned}\mathcal{L} =& \frac{1}{2}\|u-\bar{u}\|^2 + \frac{1}{2}p\delta^2 + \lambda_0\left(L_fV + L_gVu + \gamma(V) - \delta\right) \\ &- \sum_{i=1}^{M}\lambda_i\left(L_fh_i + L_gh_iu + \alpha_i(h_i)\right),\end{aligned} \quad (9)$$

where $\lambda_0, \lambda_1, \cdots \lambda_M$ are the KKT multipliers. Then, the KKT conditions are deduced as follows:

$$\frac{\partial\mathcal{L}}{\partial u} = u - \bar{u} + \lambda_0 L_g V^T - \sum_{i=1}\lambda_i L_g h_i^T = 0 \quad (10)$$

$$\frac{\partial\mathcal{L}}{\partial \delta} = p\delta - \lambda_0 = 0 \quad (11)$$

$$\lambda_0\left(L_fV + L_gVu + \gamma(V) - \delta\right) = 0 \quad (12)$$

$$\lambda_i\left(L_fh_i + L_gh_iu + \alpha_i(h_i)\right) = 0 \quad (13)$$

$$i = 1, 2, \cdots, M,$$

with $\lambda_0, \lambda_1, \cdots, \lambda_M \geq 0$. To solve the optimal primal-dual solution, we have a categorical discussion.

Case 1: If the CLF constraint is inactive, we have $L_fV + L_gVu + \gamma(V) - \delta < 0$ and $\lambda_0 = 0$. Given the equilibrium condition $\dot{x} = f + gu = 0$ and $\delta = 0$ from (11), it follows that $\gamma(V) < 0$. This contradicts the fact that $V(x)$ is a positive definite function. Therefore, no equilibria exist.

Case 2: Consider the case where the CLF constraint is active and all CBF constraints are inactive, i.e. $L_fV + L_gVu + \gamma(V) - \delta = 0$ and $L_fh_i + L_gh_iu + \alpha_i(h_i) > 0$, with $\lambda_0 > 0$ and $\lambda_i = 0, i \in [M]$. The condition (10) changes to $u - \bar{u} + \lambda_0 L_g V^T = 0$. Solving the KKT conditions, we have

$$u^* = \bar{u} - \frac{L_fV + L_gV\bar{u} + \gamma(V)}{p^{-1} + \|L_gV\|^2}L_gV^T. \quad (14)$$

Given the equilibrium condition, we have

$$f + g\bar{u} - \frac{L_fV + L_gV\bar{u} + \gamma(V)}{p^{-1} + \|L_gV\|^2}G\nabla V = 0. \quad (15)$$

The condition for (15) to have a solution is $f + g\bar{u} = \kappa G\nabla V$ for some constant κ. Solving for κ, (15) reduces to

$$f + g\bar{u} = p\gamma(V)G\nabla V. \quad (16)$$

By left-multiplying equation (16) by ∇V^T, we obtain $L_fV + L_gV\bar{u} = p\gamma(V)\|L_gV\|^2$. Since $L_fV + L_gV\bar{u} \leq 0$ and $p\gamma(V)\|L_gV\|^2 \geq 0$, it follows that the equilibrium must coincide with the goal. Therefore, deadlock cannot occur.

Case 3: Consider the case where the CLF constraint is active, and the condition (12) becomes

$$L_fV + L_gVu + \gamma(V) - \delta = 0, \lambda_0 \geq 0. \quad (17)$$

Without loss of generality, we assume that the first K CBF constraints are active and the next $M - K$ CBF constraints are inactive, where $0 < K \leq M$. Hence, we have

$$\begin{aligned}L_fh_k + L_gh_ku + \alpha_k(h_k) = 0, \quad \lambda_k \geq 0, \quad k \in [K] \\ L_fh_j + L_gh_ju + \alpha_j(h_j) > 0, \quad \lambda_j = 0, \quad j \in [M]\setminus[K].\end{aligned} \quad (18)$$

Substituting the KKT conditions (10)-(11) into (17) and (18), we obtain the following linear system in $\lambda_0, \lambda_1, \cdots, \lambda_k$:

$$\lambda_0(\|L_gV\|^2 + p^{-1}) - \sum_{i=1}^{K}\lambda_i L_gV L_gh_i^T = L_fV + L_gV\bar{u} + \gamma(V) \quad (19)$$

$$\lambda_0 L_gh_k L_gV^T - \sum_{i=1}^{K}\lambda_i L_gV L_gh_i^T = L_fh_k + L_gh_k\bar{u} + \alpha_k(h_k). \quad (20)$$

Substituting the KKT conditions (10) into the equilibrium condition yields

$$f + g\bar{u} - \lambda_0 G\nabla V + \sum_i^K \lambda_i G\nabla h_i = 0. \quad (21)$$

Left-multiplying the equation (21) by ∇V^T, we have

$$L_fV + L_gV\bar{u} - \lambda_0\|L_gV\|^2 + \sum_i^K \lambda_i L_gV L_gh_i^T = 0. \quad (22)$$

Substituting (22) into (19), we find $\lambda_0 = p\gamma(V)$ at the equilibrium state. Hence, the equilibrium condition (21) becomes

$$f + g\bar{u} - p\gamma(V)G\nabla V + \sum_i^K \lambda_i G\nabla h_i = 0.$$

Note that $F_V = f + g\bar{u} - p\gamma(V)G\nabla V$ characterizes the force guiding the system towards the goal, while $F_{h_i} = G\nabla h_i$ signifies the safety maintenance force. The condition shows that the vector $-F_V$ lies within the conical hull formed by the vectors F_{h_i}, thereby proving Property i.

Then, left-multiplying the equation (21) by ∇h_k^T, we have

$$\lambda_0 L_gh_k L_gV^T - \sum_{i=1}^{K}\lambda_i L_gh_k L_gh_i^T = L_fh_k + L_gh_k\bar{u}. \quad (23)$$

Comparing equations (20) and (23), we find that $\alpha_k(h_k) = 0$. Since the goal is not on the boundary of the safe set, this implies that deadlocks must occur at the intersection of all active CBF boundaries, proving Property ii. ■

Remark 1: Theorem 1 demonstrates that deadlock arises from an undesired equilibrium between stabilizing and safety forces, with the safety forces generated exclusively when CBF constraints are active. The conical hull formed by multiple safety forces can balance the stabilizing force within a certain range, increasing the likelihood of deadlock. Notably, while the resultant force in a QP can be directly computed through its KKT multipliers, this approach implicitly assumes static safety forces and captures only instantaneous system states. However, as the system converges to the deadlock point, the KKT multipliers dynamically adapt until the safety forces fully balance the stabilizing forces. Therefore, the conical hull is used to characterize the variable magnitudes of safety forces, where the parameter λ_i in Eq. (8) can be any nonnegative number. This approach can identify force balancing trends before reaching the deadlock point, thereby facilitating proactive

deadlock detection at an earlier stage.

Remark 2: A subtle yet crucial point in Theorem 1 is that only active CBFs contribute to deadlock, whereas inactive CBFs do not form the conical hull. Furthermore, the safety forces defining t he c onical h ull b oundary a re t he primary sources of deadlock. Consequently, focusing on these safety constraints can significantly simplify the identification of deadlock configurations. T he c onical h ull a lso c haracterizes the boundaries of potential deadlock regions, providing a useful reference for deadlock avoidance.

To illustrate Theorem 1 more intuitively, a 2-D integrator example is given.

Example 1: Consider a simple system $\dot{x} = u, x \in \mathbb{R}^2$. Let there be two stationary and circular obstacles centered at $x_{c_1} = [-1.5, 1]^T$ and $x_{c_2} = [-2.5, 0.5]^T$ with a radius of 0.8. We use two CBF $h_i = 0.5\|x - x_{c_i}\|^2 - 0.32, i = \{1, 2\}$ and the CLF $V = 0.5x_1^2 + 2x_2^2$. No nominal controller is applied, i.e. $\bar{u} = 0$. The integrator is regulated to the origin using the CLF-MCBF-QP controller given in (6). Due to the dynamics of the integrator, the deadlock condition (8) reduces to $\lambda_0 \nabla V = \sum_{i=1}^{K} \lambda_i \nabla h_i$, where $\lambda_i \geq 0$.

Fig. 2 (a)-(c) show the trajectory of the system starting from the initial state $x_0 = [-3, 3]^T$ and getting trapped at a deadlock point. The system's stabilizing force is indicated by blue arrows, and the red arrow indicates the safety force when the CBF constraint is active. The system starts from a sufficiently safe state where all CBF constraints are inactive, and the control input is determined only by the CLF, as given in (14). At $t = 0.9$ s, two CBF constraints become active, and the stabilizing force falls within the conical hull formed by the active safety force, satisfying the deadlock condition (8). By $t = 5$ s, the system settles at the intersection of the boundaries of the two CBF functions.

B. Deadlock Detection

Theorem 1 suggests that two conditions must be met for deadlock to occur: 1) the robot is at the intersection of all active CBFs, and 2) the robot's stabilizing force lies within the conical hull formed by the safety forces. Based on these conditions, a deadlock detection method is proposed to evaluate the robot's current risk of deadlock.

Let $\mathcal{H}_\rho$ be a set of active CBFs that generate safety forces:

$$\mathcal{H}_\rho = \{h_i \mid h_i \leq \rho_i, L_f h_i + L_g h_i u^* + \alpha_i(h_i) = 0\}, \quad (24)$$

where $\rho_i > 0$ is the lookahead value for the ith CBF. A larger value allows the system to detect deadlocks caused by this constraint earlier. The set of safety force generated by each CBF is denoted by $\mathcal{F}_{\mathcal{H}_\rho} = \{G\nabla h_i | h_i \in \mathcal{H}_\rho\}$ and let $Cone(\mathcal{F}_{\mathcal{H}_\rho})$ be the conical hull formed by $\mathcal{F}_{\mathcal{H}_\rho}$. We can categorize the risk of deadlock into four levels based on the relationship between F_V and $Cone(\mathcal{F}_{\mathcal{H}_\rho})$:

1) *Completely deadlock-free* state occurs when $|\mathcal{H}_\rho| = 0$. In this case, the system can converge without limit, guided by the stabilizing forces F_V.
2) *Restricted deadlock-free* state occurs when $|\mathcal{H}_\rho| \neq 0$ and $-F_V \notin Cone(\mathcal{F}_{\mathcal{H}_\rho})$. In this case, the stabilizing force F_V runs the risk of falling into the conical hull.

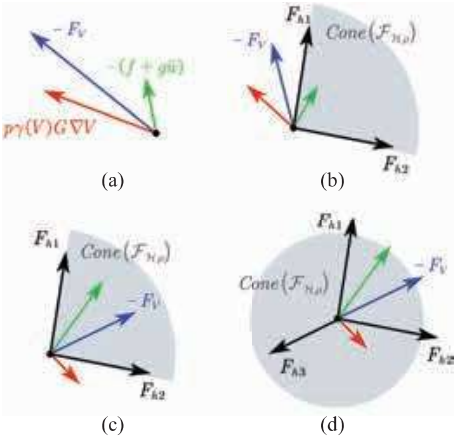


Fig. 3. Deadlock risk levels: (a) Completely deadlock-free. (b) Restricted deadlock-free. (c) Weak deadlock. (d) Strong deadlock.

3) *Weak deadlock* state occurs when $-F_V \in Cone(\mathcal{F}_{\mathcal{H}_\rho})$, but $\exists \Delta F_V \in \mathbb{R}^n$ such that $-(F_V + \Delta F_V) \notin Cone(\mathcal{F}_{\mathcal{H}_\rho})$. In this case, there exists a half-space $H(n_c) = \{y \in \mathbb{R}^n | n_c^T y \geq 0\}$ such that $Cone(\mathcal{F}_{\mathcal{H}_\rho}) \subseteq H(n_c)$, and F_V can escape the conical hull by modulating its direction.
4) *Strong deadlock* state occurs when $Cone(\mathcal{F}_{\mathcal{H}_\rho})$ covers the entire $\mathbb{R}^n$ space, making it impossible for F_V to escape the conical hull.

Fig. 3 provides a visual representation of the four deadlock types. The gray sector denotes the conical hull $Cone(\mathcal{F}_{\mathcal{H}_\rho})$, and the colored arrows depict different force vectors. In the *restricted deadlock-free* state, although the robot is outside the potential deadlock region, dynamic variations in stabilizing or safety forces may still drive it into this region. In such cases, keeping the stabilizing force outside the conical hull can prevent deadlock. The *weak deadlock* state indicates that the robot is within the potential deadlock region and approaching a deadlock point. While motion is still possible, the system will eventually settle at the deadlock point. However, by actively adjusting the direction of the stabilizing force, it can be transformed into a *restricted deadlock-free* state. In contrast, the *strong deadlock* state offers no room for active adjustment, leaving safety assurance as the only viable option.

Remark 3: In Theorem 1, the stabilizing force generated by the controller (6) consists of three components: (i) the force from the system's dynamics, (ii) the force from the reference control input $\bar{u}$ and (iii) the force from the CLF constraints. Importantly, if the CLF constraint is removed from controller (6), the result of Theorem 1 still holds, with $F_V = f + g\bar{u}$. Hence, the proposed deadlock detection method remains valid for the broader class of CBF-based controllers.

The risk of deadlock for the system in Example 1 is evaluated across different states using our proposed method, with the results presented in Fig. 2 (d). The lookahead value is set to $\rho = 3$. The proposed method can accurately identify the potential deadlock region, as indicated by the red arrows. Outside this region, near the CBF boundary, the system is in a *restricted deadlock-free* state, shown by the purple arrows. When the system is far from the CBF boundary, it is in a *completely deadlock-free* state, marked by the blue arrows.

C. Feasible Strategies for Deadlock Avoidance

The deadlock analysis results suggest that keeping the system's stabilizing force outside the conical hull is key to avoiding deadlock. Observing deadlock condition (8), there are three strategies to achieve this goal.

1) Additional nominal control input: We can introduce an additional control input, $\bar{u}_a$, to steer the stabilizing force F_V out of the conical hull. This modification adjusts the objective function (6a), which now takes the form:

$$\min_{(u,\delta)\in\mathbb{R}^m\times\mathbb{R}} \frac{1}{2}\|u-(\bar{u}+\phi_a\bar{u}_a)\|^2+\frac{1}{2}p\delta^2, \tag{25}$$

where ϕ_a is a logical selection function that activates when the system approaches the deadlock point ($\phi_a = 1$) or remains inactive otherwise ($\phi_a = 0$). Consequently, the deadlock condition (8) becomes:

$$f+g(\bar{u}+\phi_a\bar{u}_a)-p\gamma(V)G\nabla V+\sum_i^K \lambda_i G\nabla h_i=0. \tag{26}$$

This approach offers the advantage of allowing users to freely choose $\bar{u}_a$ to modulate F_V. However, the convergence guarantee of the CLF-MCBF-QP controller is compromised because $\bar{u}_a$ can be considered a disturbance. Therefore, careful design of $\phi_a(t)$ and $\bar{u}_a$ is necessary to ensure the stability of the system.

2) Narrowing deadlock range: We can reduce the coverage of $Cone(\mathcal{F}_{\mathcal{H}_\rho})$ to make F_V escape the potential deadlock region indirectly. Inspired by [44], we can introduce auxiliary control inputs into each CBF constraint to modulate the constituent vectors F_{h_i} of $Cone(\mathcal{F}_{\mathcal{H}_\rho})$ as follows:

$$L_f h_i + L_g h_i(u+\dot{\varphi}r)+\alpha(h_i)\geq 0, \tag{27}$$

where $\dot{\varphi}\in\mathbb{R}$ denotes the magnitude of the auxiliary control force, and $r\in\mathbb{R}^m$ indicates its direction. New QP controller can be formulated as (6a), (6b), and (27). Therefore, the deadlock condition (8) becomes:

$$F_V+\sum_{i=1}^K \lambda_i(G\nabla h_i+p^{-1}\nabla h_i^T gr\cdot gr)=0. \tag{28}$$

One can see from (28) that the auxiliary control $\dot{\varphi}r$ will modulate all of the component functions within the conical hull. The authors in [44] used this way to avoid deadlock in the single CBF case by adding a vector that is not perpendicular to ∇h. However, for multiple CBFs, it is difficult to find an auxiliary control to allow for consistent changes in the conical-shaped deadlock region.

3) Shaping control Lyapunov function: This strategy involves modifying the shape of CLF so that F_V varies with ∇V. In [22], a virtual control signal acting as an angular velocity is introduced to rotate a positive-definite, non-radial CLF, enabling the system to avoid the collinearity deadlock condition. Compared to the other two strategies, the continuous modulation of ∇V allows for a continuous variation of F_V, which can be leveraged to design a feedback controller that prevents deadlocks. Additionally, preserving the CLF constraint can help to maintain system stability. However, the changes in ∇V generated by simply rotating the CLF are limited and may not be sufficient to guarantee escape from the conical-shaped potential deadlock region. In the next section, we will address this limitation to accommodate more complex deadlock scenarios arising from multiple safety constraints.

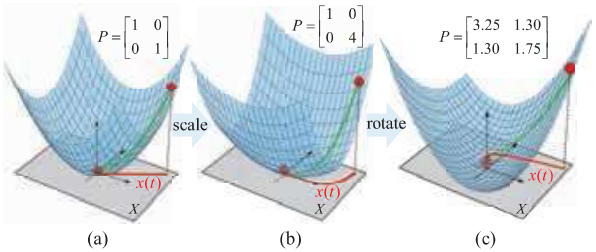


Fig. 4. System trajectories under different CLF shapes. By scaling and rotating the CLF, the system converges to the origin from various directions.

V. Deadlock Avoidance Using Lyapunov Shaping

In Section IV-B, the deadlock detection method allows the robot to identify potential deadlock regions using local information. Here, we introduce a new reactive deadlock avoidance method based on the Lyapunov shaping strategy. The core idea is to escape potential deadlock regions by instantaneously modulating the CLF shape and to prevent entry into such regions through a virtual CLF shape controller. By integrating deadlock detection and avoidance methods as a stand-alone module within the CLF-MCBF-QP controller, we establish a deadlock-aware control framework.

A. Parameterizing Control Lyapunov Functions

Consider a quadratic control Lyapunov function:

$$V(x)=x^TPx, \tag{29}$$

where $P\in\mathbb{R}^{n\times n}$ is a positive-definite matrix and x represents the control error of the system, with a slight abuse of notation. The matrix P defines the weighting relationships among different elements, which can influence the system's convergence trajectory. See Fig. 4 for illustrative examples. For any matrix P, spectral decomposition yields $P=QSQ^T$, where $Q\in\mathbb{R}^{n\times n}$ is an orthogonal matrix and $S=diag([s_1,s_2,\cdots,s_n])$ is a diagonal matrix with positive diagonal elements. In a geometric sense, $Q\in\mathcal{SO}(n)$ is a rotation state for the CLF and S is a scaling state. Since P remains positive definite during the transformation, once shape is fixed, the convergence guarantee of the CLF constraint holds. Specially, define $V_I=x^Tx$ as the identity CLF with $P=I$ and $Q=I$. The gradient of V with respect to x is

$$\nabla V=QSQ^Tx. \tag{30}$$

Note that ∇V can be modulated to any vector within the half-plane defined by $\{\nabla V \mid \nabla V^T\nabla V_I>0\}$.

To reshape the CLF continuously, define a virtual angular velocity $\omega\in\mathbb{R}^{\frac{1}{2}n(n-1)}$ to control the rotation of the CLF level sets around the origin. Similarly, define a rate $\dot{s}\in\mathbb{R}^n$ to control the scaling of the CLF level set along the rotation axis. The dynamics of shape matrices Q and S are

$$\dot{Q}=Q\hat{\omega} \tag{31a}$$

$$\dot{S}=diag(\dot{s}), \tag{31b}$$

where $\hat{\omega}\in\mathfrak{so}(n)$ is isomorphic to $\mathbb{R}^{\frac{1}{2}n(n-1)}$. For example, for $n=3$, $\hat{\omega}\in\mathbb{R}^{3\times3}$ represents the cross product operator.

B. Escaping Potential Deadlock Region

As Q and S are virtual shape parameters, ∇V can be changed instantaneously to escape the potential deadlock region. To emphasize the contribution of ∇V in stabilizing force, the deadlock condition (8) can be reformulated as

$$-q_1 + \nabla V = \sum_{h_i \in \mathcal{H}_\rho} \lambda_i \nabla h_i, \tag{32}$$

where $q_1 = p^{-1}\gamma(V)^{-1}G^{-1}(f + g\bar{u})$. For convenience of expression, write $q_2 = -q_1 + \nabla V$.

Escaping potential deadlock regions essentially involves finding a new gradient $\nabla V'$ such that $q_2' = -q_1 + \nabla V'$ lies outside the $Cone(\nabla\mathcal{H}_\rho)$. To tackle this, the reverse idea is first determining q_2' and subsequently calculating $\nabla V'$, i.e. $\nabla V' = q_2' + q_1$. Hence, the new shape matrices of CLF can be obtained by

$$Q^*, S^* = \arg\min \|QSQ^T x - \nabla V'\|, \tag{33}$$

The key idea behind selecting q_2' is to identify an appropriate boundary of $Cone(\nabla\mathcal{H}_\rho)$ and guide the variation of q_2 in the direction that moves away from the boundary as quickly as possible. Define the boundary set $\nabla\mathcal{H}_d$ of the conical hull $Cone(\nabla\mathcal{H}_\rho)$ by

$$\nabla\mathcal{H}_d = \{\nabla h_i | \nabla h_i \notin Cone(\nabla\mathcal{H}_\rho \backslash \nabla h_i), h_i \in \mathcal{H}_\rho\}, \tag{34}$$

Fig. 5 (a) illustrates a typical case of weak deadlock. Note that the boundary set $\nabla\mathcal{H}_d$ may have multiple elements, and we need to select valid boundaries as the reference for q_2'. Let $\nabla\mathcal{H}_c \leftarrow \nabla\mathcal{H}_d$ be the candidate boundaries. Recall that $\nabla V'$ can be modulated to any vector within the half-space $\{\nabla V \mid x^T\nabla V \geq 0\}$, and the force q_1 typically guides the system to converge. Therefore, all feasible q_2' vectors lie within the half-space $\{q_2 \mid q_2^T x \geq -q_1^T x\}$, as depicted in the light blue region in Fig. 5 (a). Clearly, to ensure the feasibility of q_2', we need to discard invalid boundaries ∇h_c that satisfy $\nabla h_c^T x \leq 0$, i.e. $\nabla\mathcal{H}_c \leftarrow \nabla\mathcal{H}_c \setminus \{\nabla h_c | \nabla h_c^T x \leq 0, \nabla h_c \in \nabla\mathcal{H}_c\}$.

For the rest of the boundaries in $\nabla\mathcal{H}_c$, we can construct multiple candidate $q_{2,c}$ that constitute the set $\mathcal{Q}_c$. Let δ_d be the distance vector from q_2 to the boundary ∇h_c, and we have

$$\delta_d = \mathcal{P}_{\nabla h_c} q_2 - q_2, \tag{35}$$

where $\mathcal{P}_v = I_n - vv^T/\|v\|$ is a matrix representation for the orthogonal projection operator defined over a vector $v \in \mathbb{R}^n$. Therefore, a candidate $q_{2,c}$ can be determined by

$$q_{2,c} = \mathcal{P}_{\nabla h_c} q_2 + D_0 n(\delta_d), \tag{36}$$

where $n(\delta_d)$ is the unit vector of δ_d. Note that $q_{2,c}$ obtained from (36) may not be in the half-plane $\{q_2 | x^T q_2 \geq -x^T q\}$, see the blue dashed vector in Fig. 5 (b). To handle this, we amplify $q_{2,c}$ by

$$q_{2,c} \quad \max\{-\mu q_1^T x / q_{2,c}^T x, 1\} \cdot q_{2,c}, \tag{37}$$

where $\mu > 1$ is a scale factor. Finally, we choose the vector that minimizes the change in q_2:

$$q_2' = \arg\min_{q_{2,c} \in \mathcal{Q}_c} \|q_2 - q_{2,c}\|. \tag{38}$$

Fig. 5 (c) shows the new gradient $\nabla V'$ generated according to the rule (38). However, for deadlocks arising from robots moving toward each other, if instantaneously generated q_2' are on the same side, the trajectories of the robots may obstruct each other. To address this, we can require that q_2' prioritize changing in a specific direction. Let $\zeta \in \mathbb{R}^n$ be a reference direction. Therefore, before executing (38), we can further filter $\mathcal{Q}_c$ by

$$\mathcal{Q}_c \quad \mathcal{Q}_c \setminus \{q_{2,c} | sign((q_2 \otimes q_{2,c}) \cdot \zeta) > 0, q_c \in \mathcal{Q}_c\}. \tag{39}$$

Fig. 5 (d) illustrates the result of generating q_2' with the introduction of unidirectional variation. The reference direction vector points outward from the paper, so q_2 is required to change counterclockwise.

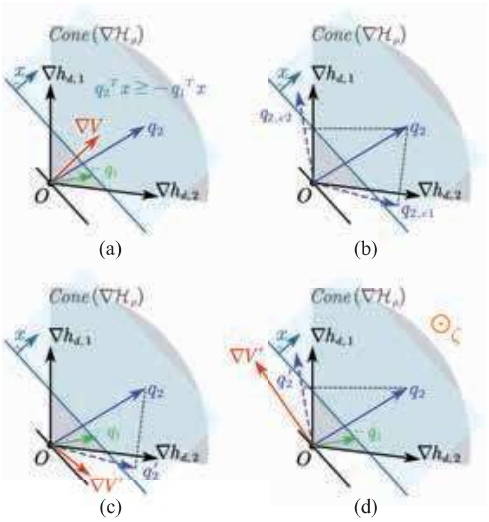


Fig. 5. Schematic of avoiding weak deadlocks. The gray sector represents the conical hull $Cone(\nabla\mathcal{H}_\rho)$, and the light blue region represents all feasible half-spaces formed by q_2'. (a) A typical weak deadlock case. (b) Two candidate vectors $q_{2,c}$ for avoiding deadlocks, where $q_{2,c2}$ is deflated by (37). (c) Stabilizing force q_2' after modulation by the principle of minimum change. (d) Stabilizing force q_2' after introducing the principle of unidirectional variation.

C. Preventing Entry into Potential Deadlock Region

Once the robot escapes potential deadlock regions, it should closely follow the user-defined behavior. For instance, if there are no obstacles, the robot is expected to move in a straight line towards its goal. At the same time, the robot needs to avoid entering potential deadlock regions to prevent weak deadlocks. Therefore, a virtual controller is designed to control the CLF to rebound to its desired shape while maintaining a safe distance from potential deadlock regions.

Define $\omega_r(t)$ and $\dot{s}_r(t)$ as reference virtual control inputs that drive the shape of the CLF to converge to the user-specified parameters Q_r and S_r. When the system is in a *restricted deadlock-free* state, activate a virtual CBF constraint to maintain a safe distance D_0 from $Cone(\nabla\mathcal{H}_\rho)$. In *restricted deadlock-free* state, the distance vector δ_d from q_2 to $Cone(\nabla\mathcal{H}_\rho)$ is defined by

$$\delta_d = \arg\min_{\delta_c} \|\delta_c\| \tag{40a}$$

$$s.t. \quad \delta_c = q_2 - \mathcal{P}_{\nabla h_d} q_2 \tag{40b}$$

$$\nabla h_d \in \nabla\mathcal{H}_d \wedge \nabla h_d^T q_2 \geq 0, \tag{40c}$$

where the mask $\nabla h_d^T q_2 \geq 0$ implies that the safety force is preventing the system from converging to the goal. The

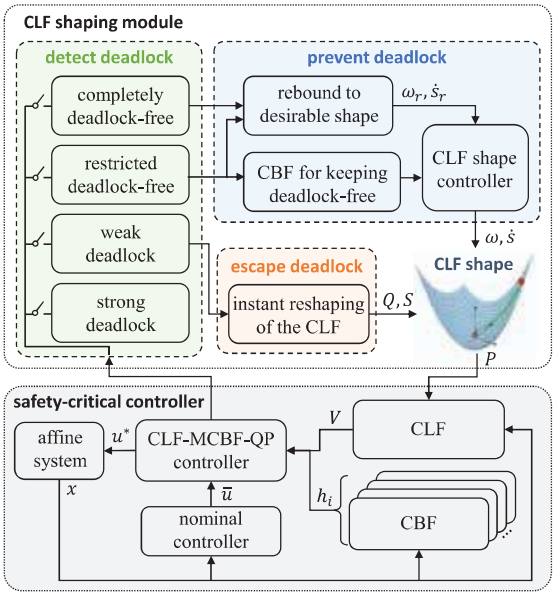


Fig. 6. The scheme of the proposed deadlock-aware control framework. A CLF shaping module serves as a plug-in to the CLF-MCBF-QP controller, enabling it to resolve deadlocks while maintaining safety.

distance D to the $Cone(\nabla\mathcal{H}_\rho)$ is defined by $D = \|\delta_d\|$. Hence, the virtual CBF constraint can be formulated as

$$\dot{D} \geq -\beta(D - D_0), \tag{41}$$

where $D_0 > 0$ is the safe distance for deadlocks and $\beta(\cdot)$ is an extended class $\mathcal{K}$ function. Since the virtual control only influences the shape matrices of CLF, we can consider D as a function of Q and S, independent of the system state x. The time derivative of D can be expressed as:

$$\dot{D} = \frac{\partial D}{\partial \nabla V}\frac{\partial \nabla V}{\partial S}\dot{S} + \frac{\partial D}{\partial \nabla V}\frac{\partial \nabla V}{\partial Q}\dot{Q}. \tag{42}$$

Note that the gradient $\frac{\partial D}{\partial \nabla V}$ represents the direction of the fastest increase of the distance function D along ∇V. Geometrically, we can interpret $\frac{\partial D}{\partial \nabla V} \approx \delta_d$. Hence, integrating the constraint (41) and referred shaping rates, the virtual controller can be formulated as

$$\min_{\omega,\dot{s}} \quad \|\omega - \omega_r\| + \|\dot{s} - \dot{s}_r\| \tag{43a}$$

$$s.t. \quad \delta_d^T \frac{\partial \nabla V}{\partial S}\dot{S} + \delta_d^T \frac{\partial \nabla V}{\partial Q}\dot{Q} \leq -\beta(D - D_0). \tag{43b}$$

The controller prevents the system from entering the potential deadlock region when constraint (43b) is active; otherwise, it ensures that the CLF rebounds to the user-defined shape.

D. Properties of Proposed Control Scheme

We encapsulate the proposed methods into a CLF shaping module and integrate it as a plugin within the CLF-MCBF-QP controller, forming a deadlock-aware control framework, as shown in Fig. 6. The pseudocode for the CLF shaping module is presented in Algorithm 1. In each control cycle, the module operates as follows: it first evaluates the system's deadlock risk based on the information from local controller (6). If the system is in *weak deadlock* state, the module instantaneously reshapes the CLF to escape the potential deadlock region

Algorithm 1: CLF shaping module

Input: system dynamics: f, g, x; CBFs/CLF: $\mathcal{H}$, V; parameters: ρ, Q_r, S_r; $\bar{u}$ (optional).
Output: New shape parameters Q, S of CLF.

1 Calculate $\mathcal{H}_\rho$, $\nabla\mathcal{H}_\rho$, q_2, $\nabla\mathcal{H}_d$;
2 **switch** *identified deadlock state* **do**
3 **case** *weak deadlock* **do**
4 $\nabla\mathcal{H}_c \quad \nabla\mathcal{H}_d$;
5 $\nabla\mathcal{H}_c \quad \nabla\mathcal{H}_c \backslash \{\nabla h_c | \nabla h_c^T x < 0, \nabla h_c \in \nabla\mathcal{H}_c\}$;
6 $\mathcal{Q}_c \leftarrow \varnothing$;
7 **for** $\nabla h_c \in \nabla\mathcal{H}_c$ **do**
8 $\delta \leftarrow \mathcal{P}_{\nabla h_c} q_2 - q_2$;
9 $q_2' \leftarrow \mathcal{P}_{\nabla h_c} q_2 + D_0 n(\delta)$;
10 $q_2' \leftarrow \max\{-\mu q_1^T x / q_2'^T x, 1\} \cdot q_2'$;
11 $\mathcal{Q}_c \quad \mathcal{Q}_c \cup q_2'$
12 $\mathcal{Q}_c \leftarrow \mathcal{Q}_c \setminus \{q_c | sign((\zeta \otimes q_c) \cdot q_c) > 0, q_c \in \mathcal{Q}_c\}$ (Optionally);
13 $q_2'^* \quad \arg\min \|q_c - q_2\|, q_c \in \mathcal{Q}_c$;
14 $\nabla V' \leftarrow q_2'^* - q_1$;
15 Calucate Q, S from (33).
16 **case** *strong deadlock* **do**
17 No operation.
18 **case** *deadlock-free* **do**
19 Calculate reference rebound inputs s_r, ω_r;
20 **if** *restricted deadlock-free* **then**
21 $\mathcal{D} \leftarrow \{q_2 - \mathcal{P}_{q_c} q_2 | q_c \in \mathcal{Q} \wedge q_2^T q_c \geq 0\}$;
22 **if** $|D| \neq \varnothing$ **then**
23 $\delta \quad \arg\min \|\delta_c\|_2, \delta_c \in \mathcal{D}$;
24 Activate virtual CBF constraint (41)
25 Calculate CLF shaping rates $\omega, \dot{s}$ from (43);
26 Update CLF shape matrices Q, S via (31).

(Algorithm 1, lines 3-15). If the system is free from deadlock, the module continuously adjusts the CLF shape to the user-specified value while preventing the system from entering potential deadlock regions (Algorithm 1, lines 18-26). Overall, the proposed control scheme has the following properties.

1) Almost weak deadlock-free: When the robot enters a weak deadlock, the potential deadlock region $Cone(\nabla\mathcal{H}_\rho)$ always lies within a half-space. Recall that the CLF shaping module confines the stabilizing force to $\{q_2 | q_2^T x \geq -q_1 x\}$, so its feasible region generally overlaps $Cone(\nabla\mathcal{H}_\rho)$. However, in the degenerate case where $Cone(\nabla\mathcal{H}_\rho)$ exactly equals a half-space $H(n_c)$ whose normal n_c aligns with the control error x, the limit of the modulated stabilizing force is tangent to the boundary of the deadlock region, and the robot cannot escape from weak deadlock. A detailed analysis is provided in Appendix A. Except for this case, the CLF shaping module can always adjust its shape parameters to drive the robot out of the weak deadlock, as demonstrated in the following lemma.

Lemma 1: Consider a robot following the kinematic model (1) with $f = 0$ and controlled by controller (6) with $\bar{u} = 0$. If the robot is in *weak deadlock*, i.e. $q_2 \in Cone(\nabla\mathcal{H}_\rho) \subseteq H(n_c)$ and assuming that the control error $x \nparallel n_c$, then Algorithm 1 can always determine new shape matrices Q and S such that the deadlock condition (8) is violated, i.e.,

$\exists \nabla V' = QSQ^T x$ such that $-q_1 + \nabla V' \notin Cone(\nabla \mathcal{H}_\rho)$.

Proof: See Appendix B. ■

Remark 4: In the case of a robot being in a strong deadlock, the CLF shaping module remains inactive, as no permissible solutions can enable the robot to escape from the deadlock condition. However, the strong deadlock of an individual robot does not imply a system-wide deadlock. In an open workspace, severe congestion is resolved progressively, starting with robots in weak deadlock. As these robots regain mobility and move away, those in strong deadlock will transition into weak deadlock, allowing the congestion to be alleviated gradually.

2) Preservation of Safety and Convergence: The proposed CLF shaping module modulates only the CLF constraint (6b) in the CLF-MCBF-QP controller, leaving the CBF constraints intact. This design guarantees that the control input continues to enforce safety, as guaranteed by Nagumo's Theorem [17]. Furthermore, since the CLF undergoes only rotation and scaling transformations, its positive definiteness is inherently preserved. When CBF constraints are active, any equilibrium indicates a deadlock state. In this case, Algorithm 1 dynamically adjusts the CLF shape to avoid the necessary conditions for deadlock, as proven in Lemma 1. When no CBF constraints are active, the CLF shape controller reshapes the CLF to a predefined value, allowing the system to converge toward the target (see Case 2 in the proof of Theorem 1). Consequently, the proposed framework maintains the convergence guarantees inherent to CLF constraints.

3) Distributed: The proposed framework requires two types of information. The first type enables the CBF-based controller to ensure safety during task execution. As shown in numerous studies, CBF controllers can effectively utilize local information to achieve safe task completion [43], [55]. The second type of information is required by the CLF shaping module. As detailed in algorithm 1, each robot depends solely on its own system dynamics, locally computed CBFs and CLF, and predefined parameters to detect and avoid deadlocks. Since the CLF shaping module operates without requiring global information, the proposed framework inherits the distributed nature of the underlying CLF-MCBF-QP controller.

4) Computationally efficient: We analyze the computational complexity of proposed method. A commonly used method for solving a QP is the interior-point method [56], which has a computational complexity of $\mathcal{O}(N_v M_c^3 + N_v^2 M_c^2 + N_v^3 + M_c^3)$, where N_v denotes the dimension of the optimization variables, and M_c denotes the number of constraints. In (6), we have $N_v = m+1$ and $M_c = M+1$, where m is the dimension of the control input and M is the number of CBFs. In Algorithm 1, the computational cost is dominated by solving the shape parameters (33) and the virtual controller (43). Problem (33) can be converted into a QP with $N_v = n(n-1)/2 + n$ variables, giving it a computational complexity of approximately $\mathcal{O}(N_v^3)$. Similarly, solving the QP (43), which has one linear constraint, also has a computational complexity of $\mathcal{O}(N_v^3)$. Compared to the CLF-MCBF-QP controller, the additional computational load introduced by the CLF shaping module is minimal. Hence, the deadlock-aware control framework is computationally efficient.

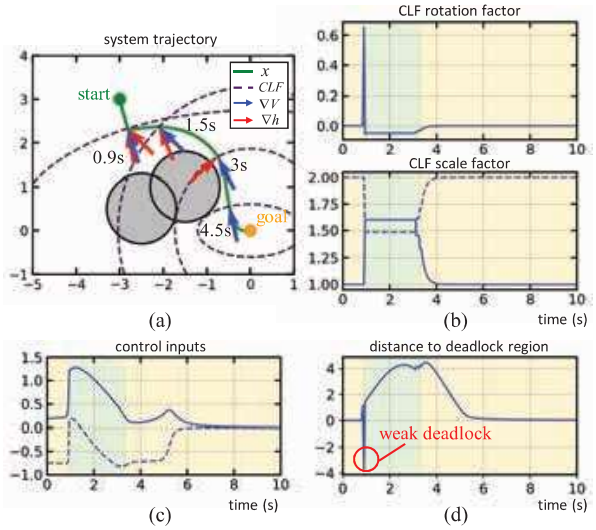


Fig. 7. Simulation results obtained by deploying deadlock-aware control framework in Example 1. (a) System's trajectory. The shape of the CLF varies with the state of the system. (b) The evolution of CLF shape parameters. Deadlock risk levels at different times are marked by different colored backgrounds: light yellow represents the *completely deadlock-free* state and light green represents the *restricted deadlock-free* state. (c) The evolution of control inputs. (d) The evolution of the distance to the deadlock region, where a negative value indicates a *weak deadlock* state.

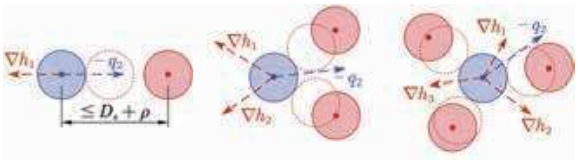
E. Deadlock Avoidance in a Toy Example

A toy example is utilized to verify the theoretical correctness of the proposed method. The simulation setup is similar to Example 1, but includes a virtual CLF shaping module. In the 2D case, the CLF shape parameter Q can be characterized by a rotation variable θ and a scale parameter $S = diag([s_1, s_2])$. The lookahead value is $\rho = 1.5$, the scale factor is $\mu = 1.1$, and the safe distance for deadlocks is $D_0 = 1.5$. Additionally, we quantify the risk of deadlock in the system using the distance D to the deadlock region. Specifically, when the system is in *weak deadlock* state, $D = -\|\delta_d\|$, where δ_d is given by (35). When the system is deadlock-free, $D = \|\delta_d\|$, where δ_d is given by (40).

Fig. 7 shows the process by which the system avoids deadlock, with the CLF shape adapting to varying levels of deadlock risk. At $t = 0.9$ s, the system detects a *weak deadlock* state and instantaneously adjusts the CLF shape. From $t = 0.9$ s to $t = 3.6$ s, the virtual CBF constraint maintains the CLF shape, ensuring the system stays at a safe distance from potential deadlock regions. After $t = 3.6$ s, the system reaches a *completely deadlock-free* state, and the CLF rebounds to its user-defined shape. Comparing results in Fig. 2, the integrator successfully avoids the deadlock and reaches the goal in a smooth trajectory.

VI. Controller Synthesis for Deadlock-Free Multi-Robot Coordination

In this section, we establish a generalized workflow to systematically analyze and resolve deadlocks across various multi-robot tasks. With a focus on multi-robot navigation and leader-follower formation tasks, we design specific controllers and analyze deadlock configurations.



(a) (b) (c)

Fig. 8. Deadlock configurations in multi-robot navigation. The solid disk represents a robot's location where deadlock can be detected in advance, and the dashed disk indicates the location when deadlock occurs. (a) Special weak deadlocks occur when two robots are aligned. (b) General weak deadlocks occur when multiple robots move symmetrically in opposite directions. (c) Strong deadlocks occur when the robot is surrounded by other robots.

A. Workflow for Multi-Robot Deadlock Resolution

A generic workflow for synthesizing deadlock-free and safe controllers for multi-robot coordination is outlined as follows:

Step 1: Design a CLF-MCBF-QP controller for multi-robot coordination, where the CLF and CBF encode the task objectives and safety constraints of the robots, respectively.

Step 2: Identify deadlock configurations of robots. Theorem 1 provides two necessary conditions for deadlocks. From this point forward, we have the following lemma.

Lemma 2: If the robot follows the kinematic model $\dot{x} = g(x)u^*$, where u^* is from controller (6), and the robot reaches the intersection of the boundaries of K active CBFs, i.e. $h_j(x) = 0$ for $j \in [K]$, and satisfies the deadlock condition:

$$\bar{u} - p\gamma(V)g^T\nabla V + \sum_j k_j g^T \nabla h_j = 0, \tag{44}$$

where $k_j > 0$, then the robot is in deadlock, i.e., $u^* = 0$.

Proof: See appendix C. ■

Note that in (44), multiple CBFs can contribute to deadlock, but the key terms are those that form the boundary of the conical hull, as discussed in Section IV-B. Therefore, when identifying deadlock configurations, it is sufficient to consider only the CBF constraints corresponding to the boundary. Additionally, due to the variety of CBF constraint types, combinations of these types must be examined to comprehensively identify all possible deadlock configurations.

Step 3: Insert the CLF shaping module (Algorithm 1) into the CLF-MCBF-QP controller and adjust the parameters accordingly. For different types of safety constraints, we can allocate different lookahead value ρ in (24) to provide early deadlock warnings without making the system overly sensitive.

B. Deadlock Configuration in Multi-Robot Navigation

We first identify and resolve deadlocks in multi-robot navigation using the proposed workflow. Consider a system of N mobile robots, where each robot i is represented by $\dot{p}_i = u_i$, with $p_i = [p_{x,i}, p_{x,j}]^T$ as its position, $u_i = [v_x, v_y]^T$ as its velocity, and $i \in [N]$. The goal for robot i is to reach a designated position p_{d_i} while avoiding collisions with every other robot. For collision avoidance, we formulate the inter-robot CBF $h_{ij} = \|\Delta p_{ij}\|^2 - D_s^2$, and the CBF constraints are constructed as $(p_i - p_j)^T u_i \geq \frac{\alpha}{4} h_{ij}$. Since each robot only needs to avoid collisions with nearby robots, we define the the neighbor set of robot i as $\mathcal{N}_i$, which can be determined by a predefined sensing radius. To reach the target, we construct a CLF $V_i = \|p_i - p_{d_i}\|^2$ for each robot instead of relying on a nominal controller. Then the CLF-MCBF-QP controller for robot i takes the form:

$$\begin{aligned} u_i^* = &\arg\min \frac{1}{2}\|u_i\| + \frac{1}{2}p\delta^2 \\ s.t. \quad &(p_i - p_{d_i})^T u_i \leq -\gamma(V_i) + \delta \qquad (45a) \\ &-(p_i - p_j)^T u_i \leq \frac{\alpha}{4} h_{ij}, j \in \mathcal{N}_i. \qquad (45b) \end{aligned}$$

To identify deadlock configurations, we substitute the terms from the controller (45) into the condition (44) and have

$$p_i - p_{d_i} = \sum_j k_j (p_i - p_j). \tag{46}$$

The deadlock configurations described by (46) are visualized in Fig. 8. For two robots, the deadlock condition is simplified to $p_i - p_{d_i} = k_j(p_i - p_j)$, indicating a *weak deadlock* when the robots are located on the connecting line between their respective targets, as shown in Fig. 8 (a). Fig. 8 (b) illustrates a general *weak deadlock* situation with multiple robots moving in opposite directions. A stable weak deadlock occurs when all robots satisfy this condition in a symmetric configuration. Fig. 8 (c) shows a case of *strong deadlock*, where robot i is surrounded by robot j in relative motion. In this case, the conical hull covers the entire plane and no actions taken by the robot can avoid deadlock. Note that although Fig. 8 only illustrates deadlock configurations among disk-shaped robots, similar configurations also arise with polygonal robots, which can be analyzed using the same workflow.

C. Deadlock Configuration in Leader-Follower Formation

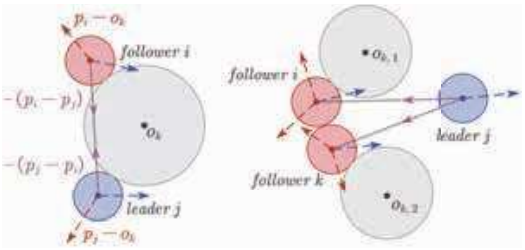
Deadlock configurations in multi-robot formation tasks arise from different types of safety constraints. Consider a team of N robots that navigate through an obstructed environment in a predefined formation. To achieve this, each robot coordinates with neighbors according to the communication topology $\mathcal{G} = (\mathcal{V}, E, A)$, where $\mathcal{V} = [N]$ represents the robot nodes, $E \subset \mathcal{V} \times \mathcal{V}$ defines the communication links between robots, and A is the adjacency matrix with $a_{ij} = 1$ if $(i, j) \in E$, and 0 otherwise. For robot i, the set of neighbors is denoted as $\mathcal{N}_i = \{j | a_{ij} = 1\}$. The formation is defined by the relative positions of the robots in the target formation, $f_{ij} = p_{d_i} - p_{d_j}$, which is constant and known as a prior.

The robots are assigned two roles: leader and follower. The leader directs the formation towards the goal, while the followers coordinate their relative positions to maintain the desired geometric shape. Based on the communication topology, the control errors for the leader, e_i^l, and the follower, e_i^f, can be derived as:

$$e_i^l = (p_i - p_{d_i}) + \sum_{j \in \mathcal{N}_i} (p_i - p_j - f_{ij}) \tag{47}$$

$$e_i^f = \sum_{j \in \mathcal{N}_i} (p_i - p_j - f_{ij}). \tag{48}$$

Therefore, the CLF can be constructed as $V_i = \|e_i\|^2$. To ensure the connectivity of the communication topology, the distance between each robot and its neighbors must be maintained within the communication radius R_c, which is achieved by constructing a CBF $\bar{h}_{ij} = R_c^2 - \|(p_i - p_j)\|^2$ for each edge $(i, j) \in E$. In addition, collisions between the



(a) (b)

Fig. 9. Examples of weak deadlocks in multi-robot formations. (a) Weak deadlocks dominated by a combination of collision avoidance constraints and connectivity constraints. (b) Weak deadlocks caused by collision avoidance constraints or connectivity constraints alone.

robot and its neighbors and obstacles have to be considered. For robot i, the set of surrounding obstacles is denoted as $\mathcal{O}_i$, and the distance from robot i to obstacle o_j is denoted as $D(p_i, o_j)$, where $o_j \in \mathcal{O}_i$. Hence, the CLF-MCBF-QP controller for robot i is designed as:

$$u_i^* = \arg\min \frac{1}{2}\|u_i\| + \frac{1}{2}p\delta^2$$
$$s.t. \quad e_i^T u_i \leq -\gamma(V_i) + \delta \tag{49a}$$
$$-(p_i - p_j)^T u_i \geq \alpha(\bar{h}_{ij}), j \in \mathcal{N}_i \tag{49b}$$
$$\nabla D(p_i, o_j)^T u_i \geq \alpha(D(p_i, o_j)), o_j \in \mathcal{O}_i. \tag{49c}$$

The deadlock condition for robot i can be obtained by substituting (49) into (44):

$$e_i = \sum_{j^o} k_j^o \nabla D(p_i, o_{j^o}) - \sum_{j^c} k_j^c (p_i - p_{j^c}) \tag{50}$$

where $k_j^o \geq 0$, $k_j^c \geq 0$, $D(p_i, o_{j^o}) = 0$ and $j_c \in \{j|\bar{h}_{ij} = 0\}$.

Fig. 9 illustrates deadlock configurations of formation task resulting from (50). In 2D scenarios, with at most two boundaries for conical hulls, there are at most two critical constraints causing deadlocks. By combining the types of constraints that form the boundary, weak deadlocks can be further categorized into three types. One type involves a combination of collision and connectivity constraints. In Fig. 9 (a), an example scenario is given where a robot formation approaches a large obstacle. The collision avoidance constraints require the robots to maneuver around the obstacle's boundary, causing the formation to split. This splitting subsequently triggers connectivity constraints between robots, creating a conical-shaped deadlock region. Two other types of deadlocks are caused solely by collision avoidance or communication connectivity. A typical scenario is when the formation navigates through a narrow passage, as in Fig. 9 (b). When robots are congested at the entrance, the collision avoidance constraints between the robots and between the robots and obstacles jointly create deadlocks, preventing the robots from moving forward. Even if some robots have already passed through, they can still encounter deadlock if the rest of the formation is stuck at the entrance. This occurs because the deadlock region generated by the connectivity constraints aligns with the moving direction, satisfying the deadlock condition (50).

D. Validation in Simulation

We use ablation tests to validate the identified deadlock configurations and the effectiveness of the proposed method.

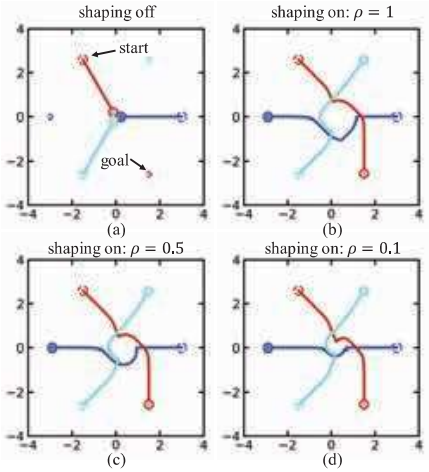


Fig. 10. Simulation results for multi-robot navigation. The label "shaping off" indicates the CLF-MCBF-QP controller, while "shaping on" indicates that the CLF shaping module is employed. (a) Three robots stop moving at the intersection. (b)-(d) All robots navigate to the target location. One can see a large lookahead value ρ is employed, the sooner deadlock is circumvented.

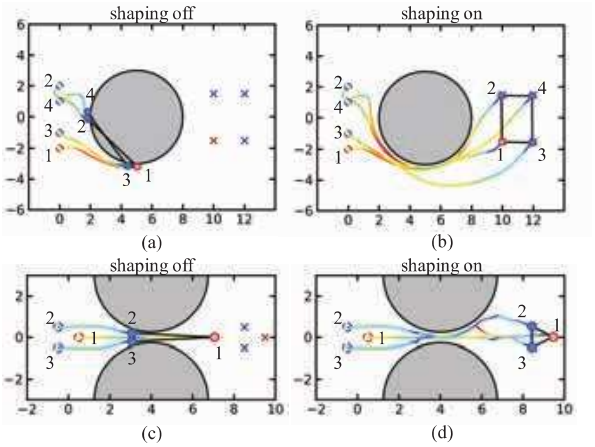


Fig. 11. Simulation results of multi-robot formation navigating through a large obstacle and a narrow corridor. The leader robot is highlighted in red, while the followers are in blue. Black lines represent the communication topology, and cross symbols indicate the goals. Trajectories are color-coded from red for faster speeds to blue for slower speeds.

For different tasks, deadlock can be solved by plugging the CLF shaping module into the CLF-MCBF-QP controller without theoretical modifications.

Fig. 10 illustrates a scenario consisting of three robots exchanging positions starting from a perfectly symmetric configuration. Under the benchmark controller, robots move closer to the center, gradually decelerate, and eventually stabilize to a weak deadlock configuration as shown in Fig. 10 (a). In our method, we use different lookahead values to predict deadlocks and the reference direction ζ is along the normal direction of the plane pointing outward, requiring the stabilizing force to prioritize clockwise change. As shown in Fig. 10 (b-d), all robots successfully navigate to the target location. Additionally, increasing the lookahead value allows the system to detect potential deadlocks earlier, enabling a more conservative strategy to avoid them.

Fig. 11 illustrates two scenarios of a multi-robot formation

TABLE I
THEORETICAL COMPARISON OF DIFFERENT STATE-OF-THE-ART METHODS FOR DEADLOCK RESOLUTION

Methods		KKT Conclusion	Deadlock Flag	Deadlock Avoidance Strategy	Potential Deadlock	Reactive	Obstacle dense Environment
CBF-based	Ours	conical hull	conical hull	force modulation	✓	✓	✓
	[22], [44]	collinearity	collinearity	force modulation	✓	✓	–
	[14]	point	geometric features	global planner	✓	✗	✓
	[40]	point	feasible set size	consistent disturbance	✗	✓	✓
	[52]	point	velocity threshold	geometric strategy	✗	✓	–
BVCs-based	[33]	point	terminal overlap	force modulation	✓	✓	–
	[35] [42]	–	geometric features	geometric strategy	✓	✓	–
Trajectory Optimization	[32], [37]	–	terminal overlap	priority assignment	✓	✗	✓

(The symbol '–' denotes features that are not involved.)

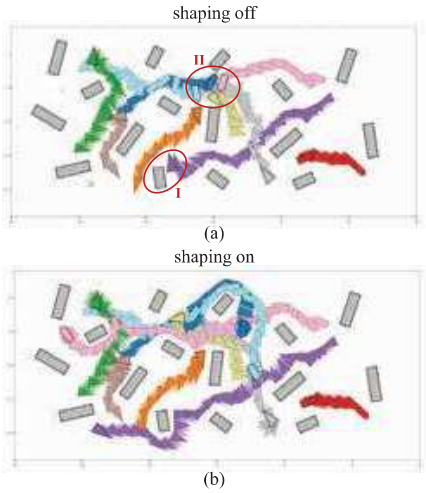


Fig. 12. Simulation results of heterogeneous polygonal robots navigating in an environment with polygonal obstacles.

task, where the robots get stuck in different deadlock configurations. In the first scenario, a formation of four robots navigates around a large obstacle. As shown in Fig. 11 (a), under the benchmark controller, the robots will fall into weak deadlocks due to formation splitting. In the second scenario, the formation needs to pass through a narrow passage. Fig. 11 (c) depicts deadlocks occurring at the passage entrance under the benchmark controller, with the follower and leader both getting stuck. In contrast, using our proposed method, the formation successfully avoids deadlocks and reaches its destination, as shown in Fig. 11 (b) and (d).

Fig. 12 presents the simulation results of a team of heterogeneous, polygon-shaped robots navigating among uniformly distributed polygonal obstacles. Both robot start and goal positions are uniformly sampled within the workspace. With the original controller, two deadlocks occur (marked by red circles in Fig. 12(a)): one collinear deadlock between a triangular robot and an obstacle, and one conic deadlock involving three robots in a narrow choke point. By contrast, with our proposed method, all robots successfully reach their target poses, as shown in Fig. 12(b).

VII. COMPARISON ALGORITHMS

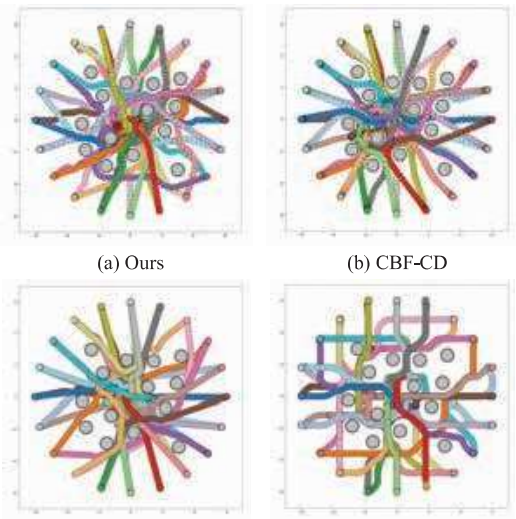
A. Qualitative Comparison

We have compared several deadlock resolution methods and summarized their key features in Table I. For work involving equilibrium analysis, most focus only on the force balance properties at the deadlock point, which can only be used to check the current state. To detect potential deadlocks, these methods require additional flags, such as terminal overlaps in the robot's future waypoints. When it comes to the coexistence of collision avoidance and connectivity maintenance constraints, geometric features can only describe typical deadlock configurations. Work [14] empirically identifies areas where deadlocks are likely to occur and uses the global planner to make the robot avoid these areas. Buffered Voronoi cell (BVC)-based methods use cell boundaries and detour points to avoid deadlocks, but they require robots to follow the same rules and have not been validated in obstacle-dense scenarios. Trajectory-optimization methods can incorporate more complex dynamic models and generate long-horizon trajectories, achieving excellent motion performance. However, in the context of deadlock avoidance, these methods rely on heuristic strategies that require robots to share high-level information—such as priorities and future waypoints—and may still become trapped in local optima.

Deadlock flags derived from abstract safety constraints encoded by CBF have better generalization. Works [22], [44] have used collinearity conditions but are limited to systems with a single CBF constraint. In contrast, our method introduces conic hull analysis, which not only accommodates multiple safety constraints encoded by CBFs but also enables detection of potential deadlocks. Moreover, since deadlock avoidance strategy is built on a generic controller, it is applicable to various tasks, like navigation in obstacle-dense environments and formation control with connectivity requirements.

B. Quantitative Comparison

For quantitative comparisons, we evaluated methods capable of multi-robot navigation in obstacle-dense environments. For clarity, we refer to the methods from [32] as LSC (Linear Safety Corridor), [37] as RSC (Recursive Safety Corridor), and [40] as CBF-CD (Consistent Disturbance). In the tests, 20 robots were initially distributed along a 6 m radius circle, tasked with swapping positions in an environment with 15



(a) Ours (b) CBF-CD

(c) RSC (d) LSC

Fig. 13. Comparison of four methods in a multi-robot position swap task.

TABLE II
COMPARISON RESULTS ON NAVIGATION PERFORMANCE METRICS

Methods	SR	T_f	L_t	T_c
Ours	100%	33.07 ± 3.79	14.75 ± 0.39	1.1
CBF-CD [40]	100%	39.36 ± 9.66	16.47 ± 1.74	4.7
RSC [37]	98%	34.20 ± 5.12	14.51 ± 0.59	8.5
LSC [32]	94%	45.26 ± 5.95	15.76 ± 0.73	10.1

(SR: success rate; T_f[s]: mean task completion time; L_t[m]: mean trajectory length of each robot; T_c[ms]: mean computation time per step. The number following ± represents the standard deviation.)

obstacles. We ran 50 Monte Carlo trials, randomly varying obstacle positions. All trials were conducted on the same computer with an Intel Core i9-13900H @ 2.6 GHz. Except for LSC method, which is implemented in C++, all other methods are implemented in Python.[1]

The sampled trajectories and quantitative metrics from successful trials are shown in Fig. 13 and Table II, respectively. While both LSC and RSC methods achieve relatively high success rates, their heuristic-based planners occasionally fail to resolve navigation conflicts. Our method and the CBF-CD method can accurately categorize deadlock types and respond accordingly, achieving a 100% success rate. However, since the CBF-CD method only addresses deadlocks after they occur, robots need to repeatedly attempt to resolve the issue, leading to longer trajectories. In contrast, our method can detect potential deadlocks and respond promptly, significantly improving efficiency in avoiding deadlock.

The LSC method requires dividing the map into grids no smaller than the robot size and adopts a more conservative strategy, such as waiting for other robots to pass, resulting in the longest task completion time. The RSC method iteratively optimizes trajectories by exchanging future waypoints with neighboring robots and using dynamic priorities for deadlock resolution, achieving the shortest average trajectory length.

[1]The source code for this implementation is available online at https://github.com/Parker-Zhang/Shaping-CLF-MCBF

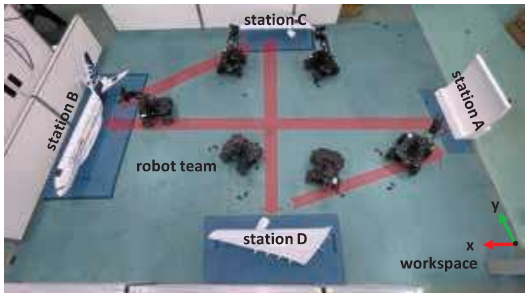


Fig. 14. Hypothetical manufacturing scenario, where six robots collaborate to perform a sequential task in parallel. Each robot is required to follow a fixed sequence of stations A-B-C-D.

Overall, our method outperforms the others in terms of efficiency, recording the shortest task completion time and a competitive trajectory length. Among all methods, our approach exhibits the smallest standard deviation across all metrics, demonstrating its consistency. With regard to computational efficiency, LSC and RSC plan long-horizon trajectories online and therefore incur higher percycle computation times. By contrast, our reactive framework achieves a computation time of only 1.1 ms per control cycle, underscoring its suitability for real-time applications in dynamic environments.

VIII. REAL ROBOT EXPERIMENTS

In this section, we conduct physical experiments to test the performance of our methods in a distributed multi-robot system with asynchronous control and imperfect motion ability. The platform consists of a team of Mecanum robotic cars, each equipped with an onboard computer, Raspberry Pi, for locally running the proposed controller, and a wireless module for communication. The robots obtain their own poses through a motion capture system and track input velocity using a low-level PID controller. The poses of neighboring robots are acquired by processing data packets broadcast by other robots, which are transmitted via a UDP-based protocol implemented using the LCM communication library [57]. The communication range is limited to an ideal value by discarding information outside the set range. A remote computer is used solely for collecting the robots' log data.

A. Multi-Robot Parallel Operation Task

In the first test, we demonstrate the application of our proposed controller in a hypothetical manufacturing scenario, where six robots collaborate in a shared workspace to perform a sequential task in parallel. The simulated factory, shown in Fig. 14, occupies an area of around 2.5 m × 3.5 m and is distributed over four workstations. Each robot is required to follow a fixed sequence of stations A-B-C-D. Lanes A-B and C-D intersect, potentially causing congestion. Moreover, potential deadlock situations arise at the entry and exit of each workstation when one robot leaves after completing its work and another robot requests entry. To verify the proposed method's ability to resolve deadlocks, no high-level coordination is imposed.

The pose of robot i is denoted as $x_i = [p_i, \theta_i]^T$, where $p_i = [p_{i,x}, p_{i,y}]$ is its position and θ_i is its heading angle.

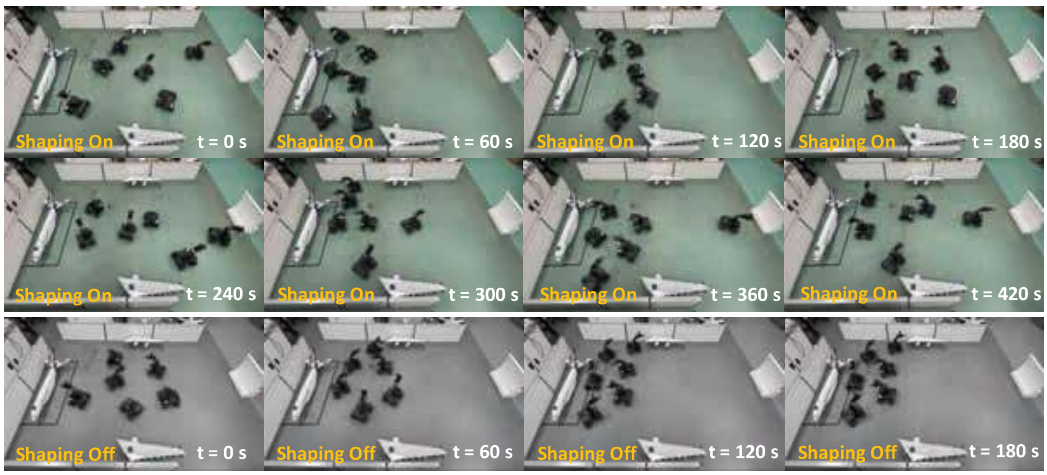


Fig. 15. Snapshots of the multi-robot parallel operation experiment comparing the performance under the CLF-MCBF-QP controller with and without CLF shaping module. The team of robots becomes deadlocked shortly after starting under the "shaping off" control. In contrast, the team of robots can work continuously when using the CLF shaping module.

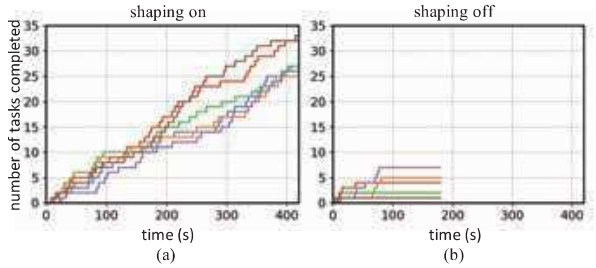


Fig. 16. The number of tasks completed by each robot with (a) and without (b) the CLF shaping module.

The control input of robot i is given by $u_i = [v_{i,x}, v_{i,y}, \omega_i]$, with $\max(|v_{i,x}|, |v_{i,y}|) \leq 0.3$ m/s and $|\omega_i| \leq 1$ rad/s. To increase the efficiency of collaboration, robots are allowed to work simultaneously around the workpiece. Therefore, a virtual stand-off circle with radius R_k and center p_k is designed for each workstation, where $k \in \{A, B, C, D\}$ corresponds to different workstations. The position error for robot i to station k is $e_{i,k} = (\|p_{i,k}\| - R_k)\hat{p}_{i,k}$, where $p_{i,k} = p_i - p_k$. The CLF used for converging to the stand-off circle is defined as $V_{i,k} = e_{i,k}^T P e_{i,k}$, where P is the shape matrix of the CLF. In addition to reaching the stand-off circle, the robot needs to orient towards the workpiece, with its target heading angle given by $\theta_{i,r} = \arctan(\frac{p_{k,y} - p_{i,y}}{p_{k,x} - p_{i,x}})$. A nominal controller $\bar{u}$ is provided to adjust the robot's heading angle, where $\bar{u}_i = [0, 0, -0.8(\theta_i - \theta_{i,r})]^T$. For collision avoidance, we use signed distance functions [58] to measure the distance between robots and obstacles, and employ the differential approximation method to compute the gradient.

Each robot is initially assigned a random target workstation and visits workstations sequentially in the order A-B-C-D. We conducted identical experiments to compare team performance with and without the CLF shaping module in the controller. Each experiment ran for a sufficient duration to minimize deviations caused by initial configuration randomness. Snapshots of the experiments are shown in Fig. 15. Under the benchmark controller, the robot team entered deadlock shortly after operation began, whereas the proposed method

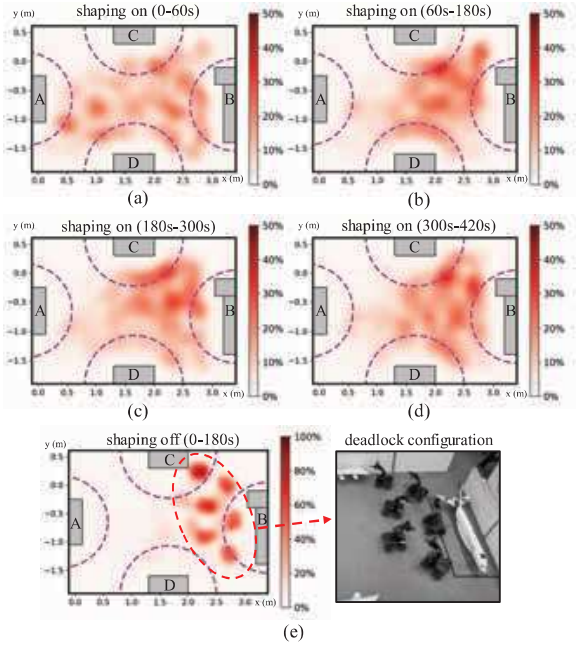


Fig. 17. Trajectory heatmaps, depicting the probability distribution of robot occupancy in the environment over time. Magenta circles mark standoff circles. (a)-(d) show the results using the proposed method. The maximum probability presented is 50% to emphasize distribution differentiation. (e) shows the result under the CLF-MCBF-QP controller. The high probability locations in the heatmap correspond to the deadlock configuration.

maintained continuous team activity throughout. The number of tasks completed by each robot is shown in Fig. 16. A task is considered completed when a robot reaches the standoff circle and completes the heading angle adjustment. With the benchmark controller, one robot experienced immediate deadlock (evidenced by a stagnant task count), which progressively caused team congestion and ultimately system-wide deadlock at approximately 90 seconds. In contrast, our approach demonstrated higher task completion efficiency, with consistently increasing task counts across all robots.

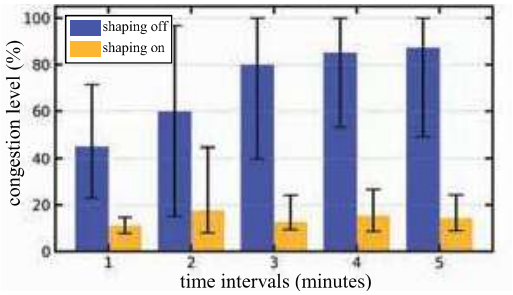


Fig. 18. Average congestion level for each 1-minute time interval across ten independent tests. The bars represent the mean congestion levels, and the error bars indicate the range from the minimum to the maximum.

To measure congestion in the environment during parallel operations, we conduct statistical analysis of the robot team's trajectories. The workspace is divided into 0.05-meter grids, and we calculate the occupancy probability of each grid over specified time intervals. Fig. 17 presents these statistical results as heatmaps, where darker colors indicate higher probabilities of robot presence within the grid during the time interval, and lighter colors denote lower probabilities. Fig. 17 (a-d) show trajectory heatmaps generated using the proposed method. The heatmaps are capped at a maximum displayed probability of 50% to enhance color differentiation. Analysis reveals that during the initial 60-second period, the trajectory distribution remains uniform, which indicates smooth robot motion. In subsequent intervals, the area between workstations B and C exhibits more severe congestion compared to other regions. This congestion arises because the narrow space between these workstations serves as a junction for entry and exit points. Despite the absence of higher-level coordination, the proposed method prevents system-wide deadlocks and maintains orderly robot operations. For comparison, Fig. 17(e) displays the trajectory heatmap from a controller without the CLF shaping module, where six distinct high-probability regions correspond to robot deadlock configurations.

To quantify the likelihood of congestion, we define a robot as congested if its speed is very low and it has not reached its target ($\|v_i\| <= 0.05$ m/s, $\|e_i\| > 0.1$ m). The system congestion level is measured by the percentage of congested robots at each time step, where 100% congestion indicates system-wide deadlock. We conducted ten repeated trials for both the benchmark controller and our proposed method, with each trial lasting 5 minutes. Data were sampled at 1-minute intervals, and the average congestion level was calculated for each time interval, as shown in Fig. 18. Results demonstrate that the benchmark controller caused more severe congestion, reaching 80% after 2 minutes in most trials. In contrast, our method maintained the system's average congestion level below 20% throughout all trials. Even when temporary congestion spikes occurred (e.g., exceeding 40% during the second interval), the proposed method effectively mitigated congestion and prevented system-wide deadlock.

B. Formation Navigation Task

In the second test, a team of robots navigate through obstacles and transition from a trapezoidal to a rectangular

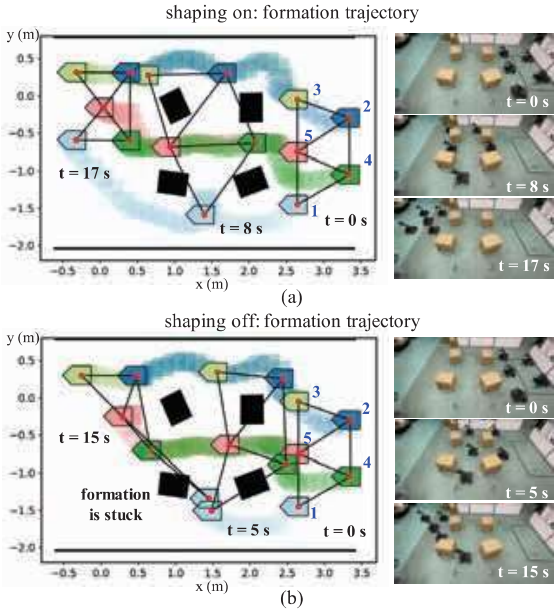


Fig. 19. Trajectories of robot formations, with the communication topology represented by a black line graph. (a) The formation reaches the target successfully under our proposed method. (b) Under the CLF-MCBF-QP controller, the entire formation is stuck because Robot 1 is deadlocked.

formation. Throughout the process, the team needs to maintain a fixed communication topology and the communication radius of the robots is set to $R_c = 1.2$ m. Fig. 19 (a) and (b) respectively show the trajectories and formation snapshots under controller (49) with and without the CLF shaping module. The trajectories are generated by overlaying robot shape markers at different time instances, where black lines depict inter-robot communication topology. Under the benchmark controller, the formation becomes stuck during the transition and fails to complete the task. In contrast, our method enables safe navigation and successful formation transition.

Fig. 20 presents the quantitative results of each robot during formation navigation, where Figs. 20 (a)-(d) correspond to the controller (49) with CLF shaping module, and Figs. 20 (e)-(h) correspond to the benchmark controller. Under the benchmark controller, the formation prematurely stops moving due to deadlock, maintaining a steady error from the target formation. In contrast, all robots successfully converge to their target positions using our method. The deadlock mechanism can be analyzed from Figs. 20 (e)-(h). At approximately 4 s, Robot 1 approaches the obstacle and decelerates due to collision avoidance constraints. By 7.5 s, Robot 1 reaches the boundary of the collision avoidance CBF and becomes deadlocked. During this period, other robots continue moving unaffected by this local deadlock. It is not until around 12.5 s that the connectivity CBF constraint between Robot 1 and Robot 5 also approaches its boundary, ultimately causing deadlock for the entire formation. Our method resolves this issue by applying lateral velocity to Robot 1 at 7.5 s, as shown in Fig. 20 (a). Consequently, Robot 1 swiftly moves away from the obstacle after 7.5 s, while the connectivity CBF maintains a safe margin throughout the process, as shown in Fig. 20 (d).

Fig. 21 illustrates the evolution of CLF shape parameters for

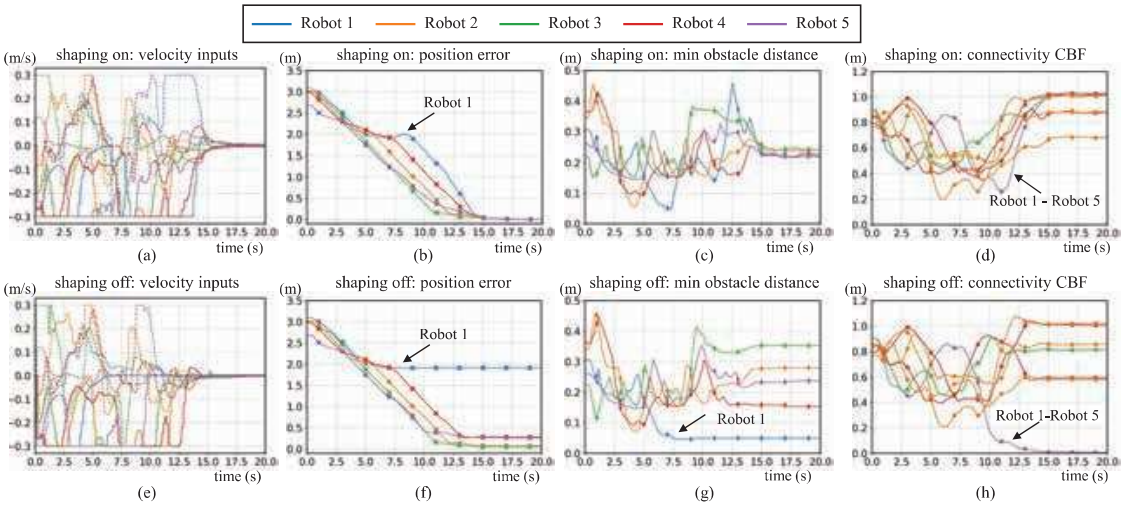


Fig. 20. Experimental results of formation navigation with (a)-(d) and without (e)-(h) the CLF shaping module. (a) and (e) record the velocity inputs of each robot. (b) and (f) show the position error of each robot relative to the target position. (c) and (g) record the minimum distance from each robot to the obstacle. (d) and (h) display the CBF values for the connectivity constraints between robots.

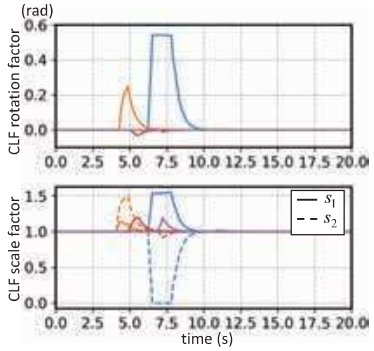


Fig. 21. Variation of CLF shape parameters with time.

each robot over time under the CLF shaping module. Initially, each robot's CLF shape is configured as a sphere with a scaling factor of 1 and a rotation angle of 0 rad. Notably, at approximately 7.5 s, an aggressive deformation strategy is applied to Robot 1, causing its scaling factor s_2 to approach zero. This occurs because Robot 1 enters a weak deadlock state, as shown in Fig. 19 (b), which requires lateral movement for resolution. When the system transitions to the *restricted deadlock-free* state, the virtual CBF constraint (43b) activates to adjust the CLF shape, thereby maintaining a safe distance from potential deadlock regions. Additionally, in the *completely deadlock-free* state, the robots' CLF parameters revert to user-predefined values to preserve motion performance.

IX. Conclusion

In this article, we propose a deadlock-aware control framework that allows robots to detect and avoid deadlocks while adhering to multiple safety constraints. Deadlocks are detected by analyzing undesired equilibria in robot dynamics, overcoming the limitations of geometric interpretation in physical space. A reactive deadlock avoidance method is designed to help robots escape potential deadlock regions when inside and maintain a safe distance using a virtual CBF constraint when outside. Our method offers two advantages: first, it establishes a general analytical framework that considers the entire continuous state space, enabling unified handling of various deadlock types; second, it achieves distributed implementation relying solely on local controller information, making it adaptable to both cooperative and non-cooperative environments.

Hardware experiments with robots performing parallel operation and formation tasks demonstrate the effectiveness of our method in avoiding deadlocks compared to benchmark controllers. It is worth noting that local congestion still occurs as our method is reactive and there is no global coordination. Moreover, infeasible scenarios may arise under arbitrary tasks, environments, and robot configurations, as discussed in Appendix A, which implies that global convergence cannot be guaranteed. However, the proposed method guarantees that all deadlocks can be detected, and nearly all weak deadlocks can be avoided. Future work will explore integrating deadlock awareness into long-horizon optimization to reduce global congestion and enhance motion performance. Another valuable direction is to incorporate acceleration limits and uncertainty sensing into the CBF construction [59], [60], enabling the proposed method to achieve reliable engineering applications.

Appendix

A. Discussion of Infeasible Cases

1) Degenerate Case: When $Cone(\nabla \mathcal{H}_\rho)$ exactly equals a half-space $H(n_c)$, whose normal n_c aligns with the control error x, the CLF-shaping module cannot generate a shape that drives the robot out of weak deadlock. This degenerate case is shown in Fig. 22(a). On one hand, the robot lies on the threshold between a weak and strong deadlock, and a slight perturbation of the deadlock boundary may push the system into a strong deadlock. On the other hand, the robot reaches the limit of the CLF-shaping module's modulation capability. However, if x is perturbed slightly away from alignment with n_c, the module can find shape parameters that enable escape from the weak deadlock, as shown in Fig. 22(b).

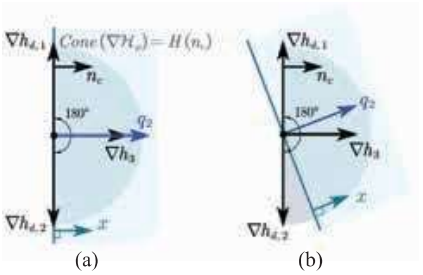


Fig. 22. (a) Degenerate case where the CLF-shaping module fails to resolve the weak deadlock. (b) Near-degenerate case with feasible escape.

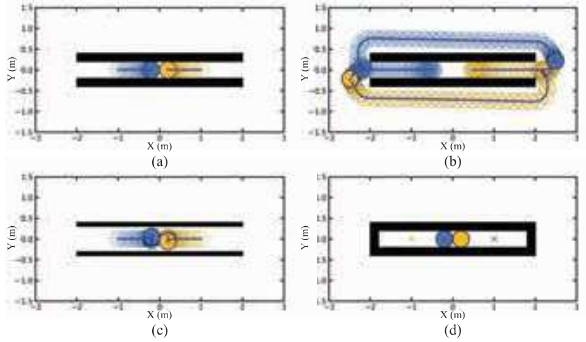


Fig. 23. Two robots in a narrow passage. (a) Degenerate case. (b) Successful case. (c) Livelock case. (d) Inherently infeasible case.

An example is shown in Fig. 23(a). Two disk-shaped robots, each of radius 0.2 m, swap positions in a narrow passage of width 0.4 m. Although the robots can detect the weak deadlock, they ultimately fail to escape due to the degenerate condition shown in Fig. 22(a) and eventually come into contact with each other. When the targets are slightly offset from the direction of the passage, the robots detect a weak deadlock configuration, as shown in Fig. 22(b). By Lemma 1, the CLF-shaping module can then generate shape parameters that break this deadlock. The corresponding simulation in Fig. 23(b) shows that the robots move relative to one another toward the exit of the narrow passage and successfully reach their respective targets.

2) Livelock Case: When the passage is widened but remains too narrow for two robots to pass, and the lookahead value is small, at most one CBF constraint is active for each robot-wall pair at any given time. As shown in Fig. 23(c), the robots exhibit livelock, characterized by repeated, patterned motion between multiple deadlock points over time. Livelock may occur in environments with static, non-cooperative obstacles, but dynamic changes typically disrupt such repetitive motion. Furthermore, it does not undermine our method's ability to resolve weak deadlocks and can be mitigated by combining our reactive module with a higher-level planner.

3) Inherently Infeasible Case: As shown in Fig. 23(d), the robots can never reach their targets, as any coordination strategy ultimately leads to either deadlock or livelock. This demonstrates that global convergence cannot be guaranteed.

B. Proof of Lemma 1

Since $f = 0$ and $\bar{u} = 0$ imply $q_1 = 0$, the CLF-shaping module can modulate q_2 into the half-space $\{q_2' \mid x^T q_2' \geq 0\}$. Under *weak deadlock* and without losing generality, assume $x \in Cone(\nabla\mathcal{H}_\rho) \subseteq H(n_c)$, which implies $x^T n_c \geq 0$. Since $n_c \nparallel x$ and every boundary satisfies $n_c^T \nabla h_d \geq 0$, there exists at least one ∇h_{d^*} for which $x^T \nabla h_{d^*} > 0$. By appropriately choosing D_0, we can find a new stabilizing force q_2' obtained from (36) such that $x^T q_2' \geq 0$. Thus, we can find a suitable set of shape parameters to avoid *weak deadlock*. ■

C. Proof of Lemma 2

We use the KKT conditions to calculate the control input u^*, following the symbols defined in theorem 1. Consequently, we have:

$$u - \bar{u} + \lambda_0 L_g V^T - \sum_{i=1} \lambda_i L_g h_i^T = 0 \tag{51a}$$

$$p\delta - \lambda_0 = 0 \tag{51b}$$

$$\lambda_0 (L_g V u + \gamma(V) - \delta) = 0 \tag{51c}$$

$$\lambda_i L_g h_i u = 0, i \in \{1, 2, \cdots, M\}. \tag{51d}$$

Case 1: Consider the solution $\lambda_i = 0, i \in \{1, 2, \cdots, M\}$ and $u^* \neq 0$. From stationary condition (51a), we have $\bar{u} = u^* + p\delta g^T \nabla V$. Substituting this into conical combination condition, we have $\sum_j k_j g^T \nabla h_j = p(\gamma(V) - \delta) g^T \nabla V - u^*$. Left-multiplying u^{*T}, we have $\sum_j k_j \nabla h_j^T g u^* = p(\gamma(V) - \delta) \nabla V^T g u^* - u^{*T} u^*$. The condition (51c) implies we have $\lambda_0 = 0$ or $L_g V u + \gamma(V) - \delta = 0$. Since the CLF constraint requires $\nabla V^T g u^* \leq \delta - \gamma(V)$, clearly for both solutions, we have $\sum_j k_j \nabla h_j^T g u^* < 0$. However, the CBF constraints require $\nabla h_i^T g u^* \geq 0$, and we have $\sum_j k_j \nabla h_j^T g u^* \geq 0$, which contradicts the previous conclusion. Therefore, this solution does not hold.

Case 2: Consider the solution $u^* = 0$. Solve the KKT conditions (51b) and (51c), we have $\delta = \gamma(V)$ and $\lambda_0 = p\gamma(V)$. Then the stationary condition (51a) becomes $-\bar{u} + p\gamma(V) g^T \nabla V - \sum_j \lambda_j g^T \nabla h_j = 0$. Observe that the conical combination condition offers solutions that $\lambda_j = k_j$. Hence, $u^* = 0$ is the only solution. So far, we prove that the robot is in a deadlock state. ■

References

[1] K. Zhang, P. Chermprayong, F. Xiao, D. Tzoumanikas, B. Dams, S. Kay, B. B. Kocer, A. Burns, L. Orr, T. Alhinai *et al.*, "Aerial additive manufacturing with multiple autonomous robots," *Nature*, vol. 609, no. 7928, pp. 709–717, 2022.

[2] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu *et al.*, "Swarm of micro flying robots in the wild," *Sci. Robot.*, vol. 7, no. 66, p. eabm5954, 2022.

[3] Z. Chen, J. Alonso-Mora, X. Bai, D. D. Harabor, and P. J. Stuckey, "Integrated task assignment and path planning for capacitated multi-agent pickup and delivery," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5816–5823, 2021.

[4] C. Wang, S. Zhu, B. Li, L. Song, and X. Guan, "Time-varying constraint-driven optimal task execution for multiple autonomous underwater vehicles," *IEEE Robot. Autom. Lett.*, vol. 8, no. 2, pp. 712–719, 2023.

[5] J. Lin, Z. Miao, H. Zhong, W. Peng, Y. Wang, and R. Fierro, "Adaptive image-based leader–follower formation control of mobile robots with visibility constraints," *IEEE Trans. Ind. Electron.*, vol. 68, no. 7, pp. 6010–6019, 2021.

[6] C. Street, S. Pütz, M. Mühlig, N. Hawes, and B. Lacerda, "Congestion-aware policy synthesis for multirobot systems," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 262–280, 2022.

[7] I. Draganjac, D. Miklić, Z. Kovačić, G. Vasiljević, and S. Bogdan, "Decentralized control of multi-AGV systems in autonomous warehousing applications," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 4, pp. 1433–1447, 2016.

[8] H. Lee, J. Motes, M. Morales, and N. M. Amato, "Parallel hierarchical composition conflict-based search for optimal multi-agent pathfinding," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 7001–7008, 2021.

[9] C. Toumieh and A. Lambert, "Decentralized multi-agent planning using model predictive control and time-aware safe corridors," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 11 110–11 117, 2022.

[10] Y. Zhou, H. Hu, Y. Liu, S.-W. Lin, and Z. Ding, "A distributed method to avoid higher-order deadlocks in multi-robot systems," *Automatica*, vol. 112, p. 108706, 2020.

[11] K. Okumura, M. Machida, X. Défago, and Y. Tamura, "Priority inheritance with backtracking for iterative multi-agent path finding," *Artif. Intell.*, vol. 310, p. 103752, 2022.

[12] Y. Zhou, H. Hu, Y. Liu, and Z. Ding, "Collision and deadlock avoidance in multirobot systems: A distributed approach," *IEEE Trans. Syst. Man Cybern. -Syst.*, vol. 47, no. 7, pp. 1712–1726, 2017.

[13] A. Mannucci, L. Pallottino, and F. Pecora, "On provably safe and live multirobot coordination with online goal posting," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1973–1991, 2021.

[14] M. Cavorsi, L. Sabattini, and S. Gil, "Multirobot adversarial resilience using control barrier functions," *IEEE Trans. Robot.*, vol. 40, pp. 797–815, 2024.

[15] 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, 2017.

[16] J. Liu, M. Li, J. W. Grizzle, and J.-K. Huang, "CLF-CBF constraints for real-time avoidance of multiple obstacles in bipedal locomotion and navigation," in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.* IEEE, 2023, pp. 10 497–10 504.

[17] A. D. Ames, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to adaptive cruise control," in *Proc. 53rd IEEE Conf. Decis. Control*. IEEE, 2014, pp. 6271–6278.

[18] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. 18th Eur. Control Conf.*, 2019, pp. 3420–3431.

[19] R. Herguedas, G. López-Nicolás, and C. Sagüés, "Multirobot transport of deformable objects with collision avoidance," *IEEE Syst. J.*, vol. 17, no. 2, pp. 3224–3234, 2023.

[20] A. Singletary, S. Kolathaya, and A. D. Ames, "Safety-critical kinematic control of robotic systems," *IEEE Control Syst. Lett.*, vol. 6, pp. 139–144, 2022.

[21] A. Li, L. Wang, P. Pierpaoli, and M. Egerstedt, "Formally correct composition of coordinated behaviors using control barrier certificates," in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.* IEEE, 2018, pp. 3723–3729.

[22] M. F. Reis, A. P. Aguiar, and P. Tabuada, "Control barrier function-based quadratic programs introduce undesirable asymptotically stable equilibria," *IEEE Control Syst. Lett.*, vol. 5, no. 2, pp. 731–736, 2020.

[23] M. Jankovic, M. Santillo, and Y. Wang, "Multiagent systems with CBF-based controllers: Collision avoidance and liveness from instability," *IEEE Trans. Control Syst. Technol.*, vol. 32, no. 2, pp. 705–712, 2024.

[24] V. N. Hartmann, A. Orthey, D. Driess, O. S. Oguz, and M. Toussaint, "Long-horizon multi-robot rearrangement planning for construction assembly," *IEEE Trans. Robot.*, vol. 39, no. 1, pp. 239–252, 2022.

[25] H. Li, T. Zhao, and S. Dian, "Prioritized planning algorithm for multi-robot collision avoidance based on artificial untraversable vertex," *Appl. Intell.*, vol. 52, no. 1, pp. 429–451, 2022.

[26] Z. Xing, X. Chen, X. Wang, W. Wu, and R. Hu, "Collision and deadlock avoidance in multi-robot systems based on glued nodes," *IEEE-CAA J. Automatica Sin.*, vol. 9, no. 7, pp. 1327–1330, 2022.

[27] Z. Ren, S. Rathinam, and H. Choset, "CBSS: A new approach for multiagent combinatorial path finding," *IEEE Trans. Robot.*, vol. 39, no. 4, pp. 2669–2683, 2023.

[28] J. Alonso-Mora, J. A. DeCastro, V. Raman, D. Rus, and H. Kress-Gazit, "Reactive mission and motion planning with deadlock resolution avoiding dynamic obstacles," *Auton. Robot.*, vol. 42, pp. 801–824, 2018.

[29] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1163–1177, 2016.

[30] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. Intell.*, vol. 219, pp. 1–24, 2015.

[31] Z. Mao, M. Hou, H. Li, Y. Yang, and W. Song, "Multi-UAV cooperative motion planning under global spatio-temporal path inspiration in constraint-rich dynamic environments," *IEEE T. Intell. Veh.*, pp. 1–14, 2024.

[32] J. Park, I. Jang, and H. J. Kim, "Decentralized deadlock-free trajectory planning for quadrotor swarm in obstacle-rich environments," in *2023 Proc. IEEE Int. Conf. Rob. Autom. (ICRA)*, 2023, pp. 1428–1434.

[33] Y. Chen, M. Guo, and Z. Li, "Deadlock resolution and recursive feasibility in MPC-based multi-robot trajectory generation," *IEEE Trans. Autom. Control*, pp. 1–16, 2024.

[34] B. Hu and Z. Cao, "Minimizing task completion time of prioritized motion planning in multi-robot systems," in *2019 Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC)*, 2019, pp. 1018–1023.

[35] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 1047–1054, 2017.

[36] M. Čáp, J. Gregoire, and E. Frazzoli, "Provably safe and deadlock-free execution of multi-robot plans under delaying disturbances," in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.* IEEE, 2016, pp. 5113–5118.

[37] Y. Chen, C. Wang, M. Guo, and Z. Li, "Multi-robot trajectory planning with feasibility guarantee and deadlock resolution: An obstacle-dense environment," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2197–2204, 2023.

[38] D. J. Gonon, D. Paez-Granados, and A. Billard, "Reactive navigation in crowds for non-holonomic robots with convex bounding shape," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4728–4735, 2021.

[39] P. M. Rivera-Ortiz and M. Kobilarov, "Decentralized safety for aggressively maneuvering multi-robot interactions," in *Proc. Amer. Control Conf.*, 2022, pp. 4682–4688.

[40] 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, 2017.

[41] R. Zheng and S. Li, "MCCA: A decentralized method for collision and deadlock avoidance with nonholonomic robots," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2710–2717, 2024.

[42] M. Abdullhak and A. Vardy, "Deadlock prediction and recovery for distributed collision avoidance with buffered voronoi cells," in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.* IEEE, 2021, pp. 429–436.

[43] R. Funada, M. Santos, R. Maniwa, J. Yamauchi, M. Fujita, M. Sampei, and M. Egerstedt, "Distributed coverage hole prevention for visual environmental monitoring with quadcopters via nonsmooth control barrier functions," *IEEE Trans. Robot.*, vol. 40, pp. 1546–1565, 2024.

[44] M. Desai and A. Ghaffari, "Auxiliary control to avoid undesirable equilibria in constrained quadratic programs for trajectory tracking applications," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 2078–2092, 2023.

[45] W. S. Cortez and D. V. Dimarogonas, "On compatibility and region of attraction for safe, stabilizing control laws," *IEEE Trans. Autom. Control*, vol. 67, no. 9, pp. 4924–4931, 2022.

[46] P. Mestres and J. Cortés, "Optimization-based safe stabilizing feedback with guaranteed region of attraction," *IEEE Control Syst. Lett.*, vol. 7, pp. 367–372, 2023.

[47] Y. Yi, S. Koga, B. Gavrea, and N. Atanasov, "Control synthesis for stability and safety by differential complementarity problem," *IEEE Control Syst. Lett.*, vol. 7, pp. 895–900, 2023.

[48] X. Tan and D. V. Dimarogonas, "On the undesired equilibria induced by control barrier function based quadratic programs," *Automatica*, vol. 159, p. 111359, 2024.

[49] W. Shaw Cortez, D. Oetomo, C. Manzie, and P. Choong, "Control barrier functions for mechanical systems: Theory and application to robotic grasping," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 2, pp. 530–545, 2021.

[50] W. S. Cortez, X. Tan, and D. V. Dimarogonas, "A robust, multiple control barrier function framework for input constrained systems," *IEEE Control Syst. Lett.*, vol. 6, pp. 1742–1747, 2021.

[51] J. Grover, C. Liu, and K. Sycara, "Why does symmetry cause deadlocks?" *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9746–9753, 2020.

[52] J. Grover, C. Liu, and K. Sycara, "The before, during, and after of multi-robot deadlock," *Int. J. Robot. Res.*, vol. 42, no. 6, pp. 317–336, 2023.

[53] S. H. Khan and A. Ghaffari, "Multi-objective trajectory planning for unmanned aerial vehicles using CLF-CBF-based quadratic programs," in *Proc. Amer. Control Conf. (ACC)*. IEEE, 2023, pp. 726–731.

[54] M. Chi, W.-T. Zhang, Z.-W. Liu, J.-Z. Xu, H. Yan, and M.-F. Ge, "Predefined-time formation control of NMSVs with external disturbance via vector control lyapunov functions-based method," *IEEE Trans. Ind. Inform.*, vol. 20, no. 4, pp. 5160–5170, 2024.

[55] T. Palani, H. Fukushima, and S. Izuhara, "Control barrier function based decentralized UAV swarm navigation while preserving connectivity

without explicit communication," in *Proc. Amer. Control Conf.*, 2023, pp. 1767–1774.
[56] L. V. Stephen Boyd, *Convex Optimization*. Cambridge, U.K.:Cambridge Univ., 2004.
[57] A. S. Huang, E. Olson, and D. C. Moore, "LCM: Lightweight communications and marshalling," in *Proc. IEEE/RSJ Int. Conf. Intelligent Rob. Syst.* IEEE, 2010, pp. 4057–4062.
[58] B. Dai, R. Khorrambakht, P. Krishnamurthy, V. Gonçalves, A. Tzes, and F. Khorrami, "Safe navigation and obstacle avoidance using differentiable optimization based control barrier functions," *IEEE Robot. Autom. Lett.*, vol. 8, no. 9, pp. 5376–5383, 2023.
[59] D. R. Agrawal and D. Panagou, "Safe control synthesis via input constrained control barrier functions," in *Proc. 60th IEEE Conf. Decis. Control (CDC)*. IEEE, 2021, pp. 6113–6118.
[60] S. Liu, C. Liu, and J. Dolan, "Safe control under input limits with neural control barrier functions," in *Conference on Robot Learning*. PMLR, 2023, pp. 1970–1980.

Zhenwei Zhang (Member, IEEE) received the B.S. degrees in mechanical engineering from Huazhong University of Science and Technology (HUST) in 2020. He is currently working toward the Ph.D. degree in mechanical engineering with the State Key Lab of Digital Manufacturing Equipment and Technology, HUST. His research interests include networked and distributed systems, multi-robot systems, and robotic manufacture.

Yuhao Zhang (Member, IEEE) received the B.S. and Ph.D degrees in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2017 and 2024, respectively. He is currently a post-doc research fellow with the State Key Laboratory of Intelligent Manufacturing Equipment and Technology at HUST. His research interests include model predictive control, impedance control, robotic manufacture, and intelligent robot technologies.

Xingwei Zhao received B.S. and M.S. degrees in mechanical engineering from University of Duisburg-Essen, Duisburg, Germany, in 2012 and 2013, respectively. He received the Ph.D. degree in mechanical engineering from the Technical University of Berlin, Berlin, Germany, in 2017. He is currently an associate research fellow with the State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology (HUST), Wuhan, China. His research interests mainly include nonlinear dynamics, robot control and robotic manufacture.

Bo Tao (Member, IEEE) received the B.S. and Ph.D. degrees in mechanical engineering from Huazhong University of Science and Technology (HUST) in 1999 and 2007 respectively. After being a postdoctor from June 2007 to June 2009, he has been an Associate Professor in 2009 and a Professor in 2013 at the School of Mechanical Science and Engineering, HUST. From June 2013 to June 2014, he was a visiting scholar at the Mechanical Engineering Department of UC Berkeley, USA. He has published 2 monograph, more than 100 papers in international journals and conference. His research interests mainly include intelligent manufacturing and robotics technologies, IOT technologies and applications.

Han Ding (Senior Member, IEEE) received the Ph.D. degree in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1989. Supported by the Alexander von Humboldt Foundation, he was with the University of Stuttgart, Stuttgart, Germany, from 1993 to 1994. He was with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore from 1994 to 1996. He has been a Professor at HUST since 1997. He has been elected as a Member of the Chinese Academy of Sciences in 2013. His research interests include robotics, multi-axis machining, and equipment automation.