IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Navigation of Robotic Swarmalators with Dynamics and Constraints

Xinyue Xu[1], Wei Xiao[2], and Steven Ceron[1]

Abstract— **Swarmalator studies have enabled self-organized collective behaviors that emerge from dual spatial and temporal coupling, without relying on external inputs. These behaviors arise solely from attractive and repulsive interactions modulated by a few global parameters. Here, we treat the swarmalator model as a planner and study how several of the collective behaviors change in terms of space-phase organization when the agents are robots with vehicle dynamics and constraints: including omnidirectional, unicycle, and bicycle dynamics. Furthermore, we use the control barrier function method to guarantee that the collective can navigate around objects, through cluttered environments, and transport objects in between obstacles by exploiting global and local control methods. This work brings us closer to realizing large groups of robotic swarmalators, of heterogeneous dynamics, that can enable shape formation, navigation, and object manipulation in cluttered environments.**

Swarm Robotics, Distributed Robot Systems, Optimization and Optimal Control.

I. Introduction

Natural collective systems, like fish schools [1], bird flocks [2], and bacterial colonies [3], exhibit spatial self-organization, which results from agents coupling to each other through spatial variables like cohesion and alignment, and lead to swarming behaviors [4]. Other systems, like flashing fireflies [5], clapping audiences [6], and firing neurons [7] exhibit temporal self-organization, which results from temporal coupling between agents; here, agents couple to each other through their internal oscillating states and enable synchronization-like behavior [8]. By considering the interplay between these two types of self-organization, we can better understand and replicate the dual synchronization and swarming behaviors exhibited by diverse collective systems. Swarmalators were initially introduced to study emergent behaviors that result from dual spatial and temporal self-organization [9]; it is a framework that models point agents whose motions depend on their relative internal state (phase) interactions, and whose phase interactions depend on their relative positions and motions.

Throughout the past few years, researchers from physics, biology, and mathematics have dived into this model to understand the fundamental mechanisms that lead to the diverse collective behaviors it enables, as well as to find its intrinsic connection to real natural collective systems [10], [11], [12]. The model may also be valuable to swarm roboticists because it enables large groups of agents to self-organize into very specific formations or exhibit predictable collective motions even though each constituent has little knowledge about others' actions. This would be useful for robot swarms at many different length scales, in which there are restrictions on computational capabilities or access to information about other agents' behaviors; it would enable robots to tune their spatial attraction / repulsion to enable diverse formations and motions. Roboticists have realized some of the swarmalator behaviors on groups of macro-scale and micro-scale robots [13], [14], [15], and have applied optimal control methods in computational studies to optimize the coupling between agents to enable desired formations [16].

Throughout these studies, they assume omnidirectional point robots and obstacle-free workspaces. To bring these collective behaviors closer to real-world applications, it is necessary to consider how the collective behaviors change when the agents have constraints on their vehicle dynamics and when the environment becomes cluttered.

In parallel, existing multi-agent systems [17], [18], [19] usually refer to a number of agents with external inputs, and we usually need to formulate centralized / decentralized optimizations in order to find controls for all the agents such that they can achieve certain formations, which is computationally expensive and hard to scale to large multi-agent systems. Existing works employ control barrier function (CBF) and control Lyapunov function (CLF) [20], [21] to ensure safety and stability of those multi-agent systems, and it may be hard for multi-agent systems to switch formations to adapt to varying environments.

By contrast, the swarmalator model enables self-organized systems without external inputs, where diverse collective behaviors arise solely from attractive and repulsive interactions between the agents that can be tuned through global coupling parameters; thus making it computationally efficient and scalable. The model can also quickly switch formations to adapt to varying environments. There are limited studies on how to incorporate guarantees in those self-organized systems, which we aim to address in this paper.

Our contributions are as follows:

- We study emergent collective behaviors enabled by the swarmalator model when the agents are constrained by unicycle and bicycle dynamics and compare to omnidirectional dynamics (no dynamic constraints).
- We use the control barrier function method to ensure that the collective can move toward a desired target while avoiding obstacles and transporting an object.
- We study how the dynamical constraints affect the collective behaviors when agents are moving around a single obstacle, in between two large obstacles, through a cluttered environment, and transporting an object.

[1]Univ. of Michigan, Ann Arbor, MI, USA; [2]Worcester Polytechnic Institute, Worcester, MA, USA. xxinyue@umich.edu, wxiao3@wpi.edu, sceron@umich.edu

We use the CBF method [22], [23] to control the group to avoid obstacles and exploit global control (treat the group as a single controllable unit) and local control (each agent computes its control input) to enable motion around obstacles when the agents are exhibiting certain dynamical constraints. CBFs are Lyapunov-like functions for control that are rooted in optimization [24], [25]; they have been used for multi-objective control and to prove set invariance. CBFs can be used for mapping a constraint defined over system states to a control constraint whose satisfaction implies the satisfaction of the original state constraint. By optimizing the collective behaviors according to the surrounding environment and the vehicle dynamics, we can make the swarmalator model useful for controlling self-organizing and task-driven robot swarms in cluttered settings.

II. Swarmalator Model

Each agent i is modeled as a *swarmalator*, possessing both a spatial state and an internal phase. Specifically, for each agent, the position is $\mathbf{x}_i(t) = (x_i(t), y_i(t)) \in \mathbb{R}^2$ and its phase is $\theta_i(t) \in [0, 2\pi)$. The swarmalators we study here have their spatial dynamics governed by an equation of motion (1) which dictates how agents move depending on the spatial attraction / repulsion enabled by other agents' relative phases. To study the swarmalator behaviors from a robotics perspective, we adjust the swarmalator model's equation of motion so that it considers agents of finite-size that cannot get infinitesimally close to each other. $\dot{\mathbf{x}}_i^{\text{swm}}$ is the velocity induced by interactions with other agents:

$$\dot{\mathbf{x}}_i^{\text{swm}} = \frac{1}{N}\sum_{j\neq i}^{N}\left[\frac{\mathbf{r}_{ji}}{\|\mathbf{r}_{ji}\|}\left(A + J\cos(\theta_j - \theta_i)\right) - B\frac{\mathbf{r}_{ji}}{\|\mathbf{r}_{ji}\|\left(\mathbf{r}_{ji} - d\right)}\right] \tag{1}$$

Here, N is the total number of agents, $\|\mathbf{r}_{ji}\| = \|\mathbf{x}_j - \mathbf{x}_i\|$, which represents the distance between the global positions, $\mathbf{x}_j$ and $\mathbf{x}_i$, of agents j and i, and their phases are given by θ_j and θ_i, respectively. The parameter d defines the minimum allowable distance between two agents, which is set to twice an individual agent's radius to prevent collisions. The attraction strength is controlled by parameter A, whereas B regulates the repulsive force. Increasing the ratio of $\frac{B}{A}$ increases the equilibrium spacing between agents. The global parameter J quantifies the influence of phase similarity on spatial attraction, and is kept between -1 and 1 throughout this study; when $J > 0$, agents with similar phases attract, while for $J < 0$, agents with similar phases repel.

Swarmalators' phase dynamics follow a distance-dependent variant of the Kuramoto model [26], which dictates how each agent's phase evolves over time depending on how it is coupling / anti-coupling to other agents' phases:

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum_{j\neq i}^{N}\frac{\sin(\theta_j - \theta_i)}{\|\mathbf{r}_{ji}\|} \tag{2}$$

Here, K represents the phase coupling strength; when $K > 0$, agents tend to synchronize by minimizing their phase differences, whereas for $K < 0$, they strive to maximize their phase differences, leading to asynchrony. Intuitively, one can think of the phase θ_i as a changeable internal state exhibited by each agent. The parameter K governs whether the tend to align their states ($K > 0$) or diversify ($K < 0$). The spatial attraction in (1) is modulated by phase similarity through J, the phase dynamics indirectly shape how agents position themselves such that agents with similar phases are more likely to cluster together when $J > 0$, while dissimilar phases promote separation.

III. Problem formulation

Based on the swarmalator model in (1) and (2), consider each agent i has a planar position $\mathbf{x}_i \in \mathbb{R}^2$ and a phase $\theta_i \in [0, 2\pi]$. The swarmalator dynamics yields a velocity $\dot{\mathbf{x}}_i^{\text{swm}}$ that depends on interactions with neighboring agents and their phases. Formally, we write (1) as a variant:

$$\dot{\mathbf{x}}_i^{\text{swm}} = f(\mathbf{x}_i, \theta_i, \{\mathbf{x}_j, \theta_j\}_{j\neq i}) \tag{3}$$

Where $f(\cdot)$ encodes the intrinsic swarmalator behavior. To steer the system beyond its natural swarmalator behavior, we introduce an additional control. We consider two distinct paradigms: Global Control and Local Control. In the Global Control scheme, a unified control signal $\mathbf{u} \in \mathbb{R}^2$ is computed centrally and applied uniformly to all agents (i.e., $\mathbf{u}_i = \mathbf{u}$ for all i). In the Local Control scheme, each agent i is assigned its own control input $\mathbf{u}_i = \left(u_{x_i},\, u_{y_i}\right) \in \mathbb{R}^2$. The resulting agent dynamics become:

$$\dot{\mathbf{x}}_i \;=\; f(\mathbf{x}_i, \theta_i, \ldots) \;+\; \underbrace{g(\mathbf{x}_i)\,\mathbf{u}_i}_{\text{2D control in the plane}} \tag{4}$$

Where $g(\mathbf{x}_i) = \mathbf{I}_2$ if we assume the control acts directly in the plane without further constraints. Thus, $\dot{\mathbf{x}}_i = \left(\dot{x}_i, \dot{y}_i\right)$ is effectively the commanded velocity for the i-th agent in 2D.

In a fully omnidirectional robot, this velocity can be directly commanded; the agent truly moves at $\dot{x}_i, \dot{y}_i$ in the plane. However, many physical platforms (e.g., differential-drive or car-like robots) impose non-holonomic constraints. Hence, while the model in (4) still generates a 2D velocity numerically, the robot cannot just set $(\dot{x}_i, \dot{y}_i)$ arbitrarily. Instead, it interprets $\dot{\mathbf{x}}_i$ as a desired or target velocity, then attempts to realize it subject to its own kinematic constraints.

Throughout this study, we only use the control variable $u_i = [u_{xi}, u_{yi}]$. Our results with non-omnidirectional motion (unicycle / bicycle) demonstrate how we can execute the desired speed command expressed by $\dot{\mathbf{x}}_i$ on a real platform with dynamical constraints, rather than designing a new feedback control or new input. Therefore, $\dot{\mathbf{x}}_i$ is an ideal desired velocity; it is what the non-holonomic robots throughout the remainder of this study are trying to map this velocity to based on their own dynamical constraints. In what follows, we show how unicycle or bicycle robots map $\dot{\mathbf{x}}_i$ to their feasible control signals.

As a result, our framework naturally separates into three layers. At the swarmalator layer, we recover the intrinsic collective motion simply by setting the extra control input $u_i = 0$ as in (3). At the CBF-augmentation layer, we compute a correction u_i via the global control (10) or local

control (13) to obtain the desired velocity as shown in (4). Finally, at the execution layer, each robot runs a low-level PID controller that does not design the desired velocity but only tracks it—so that the real robot velocity matches the swarmalator-inspired command.

IV. Fusion of Swarmalator and Robot Dynamics

We now illustrate the mapping from the planar velocity $\dot{\mathbf{x}}_i = (\dot{x}_i, \dot{y}_i)$ to the inputs of classical non-holonomic robot models. There is no additional control input need to be applied, only a kinematic transformation.

Notably, although we do not provide a full safety proof here, recent work on model-free safety-critical control where single-integrator velocity commands are formally tracked by higher-order robotic models – provides a rigorous foundation for our mapping [27], [28], [29]. These results imply that velocity commands generated by (4) can be safely realized on unicycle and bicycle platforms under appropriate tuning.

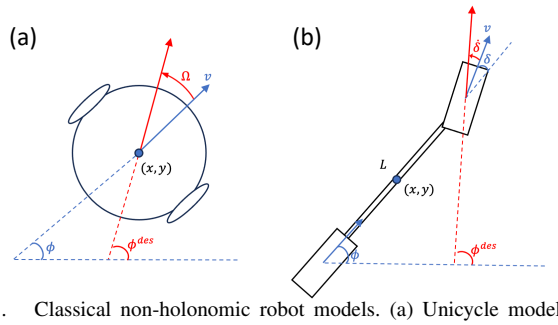

Fig. 1. Classical non-holonomic robot models. (a) Unicycle model. (b) Bicycle model.

A. Unicycle dynamics

The unicycle model introduces constraints that require explicit orientation control as shown in Fig. 1a. Each agent's state is characterized by planar coordinates (x_i^{uni}, y_i^{uni}) and orientation ϕ_i. In particular, the planar position evolves via $\dot{x}_i^{uni} = v_i \cos\phi_i$ and $\dot{y}_i^{uni} = v_i \sin\phi_i$, while the heading angle satisfies $\dot{\phi}_i = \Omega_i$. Here, $v_i \in \mathbb{R}^+$ and $\Omega_i \in \mathbb{R}$ represent linear and angular velocities, respectively.

To map the swarmalator plus control spatial commands from (4) onto this constrained system, we interpret the output $\dot{\mathbf{x}}_i$ as a desired velocity vector. Specifically, we set $v_i = \|\dot{\mathbf{x}}_i\| = \sqrt{\dot{x_i}^2 + \dot{y_i}^2}$. The desired orientation ϕ_i^{des} is derived by $\phi_i^{\text{des}} = \arctan(\dot{y_i}, \dot{x_i})$, and the orientation error is $e = \phi_i^{\text{des}} - \phi_i$. A proportional control law yields the angular velocity $\Omega_i = K_p \cdot e$, where K_p is a proportional gain that determines how quickly the agent corrects its heading error. Under the proportional law, the error dynamics $\dot{e} = -K_p e$ are exponentially stable, guaranteeing $\phi_i \to \phi_{\text{des},i}$ and hence asymptotic tracking of the desired velocity $\dot{\mathbf{x}}_i$ [27].

The phase dynamics remain governed by (2), preserving the swarmalator synchronization properties. Thus, this formulation enables physical realization of swarmalator behaviors on platforms with non-holonomic constraints, while maintaining the original phase-space coupling.

B. Bicycle dynamics

The bicycle model extends the unicycle framework by incorporating steering kinematics, thereby introducing additional constraints on motion control as demonstrated in Fig. 1b. Unlike the unicycle model, where orientation ϕ_i can be directly controlled through an angular velocity Ω_i, the bicycle model's orientation depends on both the linear velocity v_i and steering angle δ_i. Concretely, the position coordinates satisfy $\dot{x}_i^{bi} = v_i \cos\phi_i$ and $\dot{y}_i^{bi} = v_i \sin\phi_i$, while the heading angle evolves via $\dot{\phi}_i = \frac{v_i}{L_i} \tan(\delta_i)$. Here, $\delta_i \in [-\delta_{\max}, \delta_{\max}]$ is the steering angle (mechanically constrained), and L_i is the wheelbase length. The orientation dynamics depend on both linear velocity and steering angle, introducing velocity-dependent curvature constraints.

Given a desired planar velocity $\dot{\mathbf{x}}_i$ as computed in (4), we follow a strategy similar to the unicycle case: the magnitude of $\dot{\mathbf{x}}_i$ defines the linear speed v_i, and its direction determines the desired orientation ϕ_i^{des}. In the bicycle model, steering control builds on the unicycle's orientation control by incorporating a derivative term. Since the steering angle δ_i does not immediately match the orientation, applying only proportional control can result in heavy oscillations. To alleviate this overshoot, we employ a proportional-derivative (PD) control law of the form $\dot{\delta}_i = K_p \cdot e + K_d \cdot \dot{e}$, where $\dot{e}$ denotes the time derivative of the orientation error and K_d is a gain that modulates the damping effect, thereby reducing oscillations. The PD control law is exponentially stable for $K_p, K_d > 0$. This ensures bounded tracking error and practical realization of the commanded velocity [29]. The phase synchronization still follows (2), but the additional coupling between translational motion and rotational steering produces richer collective dynamics in steering-based platforms.

V. Constraint-Driven Collective Behaviors

We conducted comparative experiments between non-holonomic implementations (unicycle and bicycle models) and the idealized omnidirectional swarmalator system. Despite kinematic constraints, both non-holonomic models successfully reproduced characteristic swarmalator states observed in the omnidirectional case, as shown in Fig. 2. Each system comprised of 100 space-filling agents, each with a radius of 0.1 units, colored by the phase ($\theta \in [0, 2\pi]$). Experimental domains measured 10×10 units for omnidirectional / unicycle models and 30×30 units for bicycle implementations to accommodate their distinct maneuvering characteristics.

To maintain collision-free operation under real-world motion constraints, we established safety margins through parameter adaptation. The repulsion coefficient B was incrementally increased until collision avoidance was achieved in the most compact configuration ($K = 1$, $J = 1$). Final parameters were $B = 3$ for omnidirectional / unicycle models and $B = 10$ for bicycle models, with consistent attraction $A = 1$ across all implementations.

Fig. 2 demonstrates key emergent behaviors of swarmalators for five different values of K, all with $J = 1$. When $K = 1$, agents converge into a compact circle with fully

synchronized phases, referred to as Static Sync. For $K = 0$, agents assume a Static Phase Wave, retaining their initial phases to form a phase gradient along a ring-like shape. As K becomes increasingly negative, the phase interactions become anti-coupled. At $K = -0.05$, a Splintered Phase Wave arises, where agents form several sub-clusters with partial phase segregation. Decreasing further to $K = -0.5$ induces persistent circular motion in clusters, producing an Active Phase Wave, as highlighted by black trajectory lines. Finally, at $K = -1$, agents move in a disordered manner and become an asynchronous group.

It is noteworthy that due to the velocity-dependent turning constraints in the bicycle model, agents do not achieve a completely static state; rather, they exhibit small residual motions around their nominal positions. Nevertheless, on a macroscopic scale, the agents form the intended patterns that are characteristic of the swarmalator dynamics.

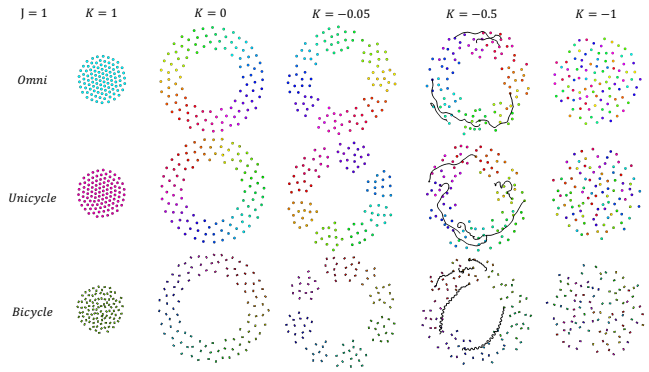

Fig. 2. Final configurations of 100 agents under the three dynamic models with $J = 1$. From left to right: $K = 1$ (Static Sync), $K = 0$ (Static Phase Wave), $K = -0.05$ (Splintered Phase Wave), $K = -0.5$ (Active Phase Wave), and $K = -1$ (Fully Disordered Phase). The black lines highlight sample trajectories in the Active Phase Wave.

Fig. 3 illustrates how the collective radii varies with the phase coupling strength J under the static phase wave regime ($K = 0$). We plot the average inner and outer radii of the ring formed by the agents for the three different dynamic models. As J increases from 0.1 to 1.0, the ring expands in all models, reflecting a broader spatial distribution of agents. The bicycle model consistently yields the largest radii, owing in part to the larger repulsive parameter used to prevent collisions under its non-holonomic constraints. In contrast, the omnidirectional and unicycle models have almost the same radii, which indicates the unicycle model can perform similar to the ideal situation.

VI. Controllable Robotic Swarmalators

Our control framework for robotic swarmalators focuses on two primary objectives, namely (1) trajectory tracking and (2) obstacle avoidance. To achieve these goals, we adopt a hybrid control strategy that combines CLFs [30] for convergence guarantees with CBFs for safety-critical constraints. As presented in Sec. III, our approach incorporates both Global Control, where a centralized optimization treats the entire swarm as a unified system under common constraints,

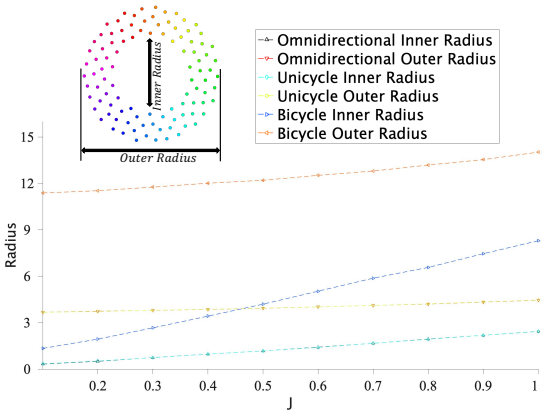

Fig. 3. Line plot illustrating how the average inner and outer radii vary with the phase coupling strength J across different dynamic models (omnidirectional, unicycle, and bicycle).

and Local Control, in which each agent computes its control input based on local neighborhood information.

We assume that each agent's position is available at every control step. The control loop operates with a fixed timestep of $\Delta t = 0.1\,\text{s}$, which provides sufficient margin for both computation and communication. Control inputs are broadcast to all agents, and the positions of static obstacles are assumed to be known. To reflect realistic operating conditions, we impose a velocity bound on the control inputs of $\|\mathbf{u}_i\| \leq 1\,\text{m/s}$.

A. Control Lyapunov Function Formulation

To ensure trajectory tracking for the swarm, we define a swarm-level Lyapunov function $V(\mathbf{x})$ based on the collective centroid dynamics. Let $\mathbf{p}_{\text{target}} \in \mathbb{R}^2$ denote the swarm's desired target position. The Lyapunov function is $V(\mathbf{x}) = \|\bar{\mathbf{x}} - \mathbf{p}_{\text{target}}\|^2$, where $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x_i}$ represents the centroid of the swarm, and N is the total number of agents. The Lyapunov function $V(\mathbf{x})$ quantifies the squared Euclidean distance between the swarm centroid and the target position, providing a measure of the collective tracking error.

To guarantee exponential convergence of the swarm centroid to the target position, we enforce the CLF condition:

$$\inf_{u \in \mathcal{U}} \left[L_f V(\mathbf{x}) + L_g V(\mathbf{x}) u + \lambda V(\mathbf{x}) \right] \leq 0 \tag{5}$$

Where $L_f V(\mathbf{x})$ and $L_g V(\mathbf{x})$ are the Lie derivatives of $V(\mathbf{x})$ along the vector fields $f(\mathbf{x})$ and $g(\mathbf{x})$, respectively. The parameter $\lambda > 0$ controls the rate of exponential convergence. The CLF condition ensures that the control input u drives the swarm centroid toward the target position while maintaining stability. The Lie derivatives are computed as $L_f V(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x})$, $L_g V(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} g(\mathbf{x})$, where $\frac{\partial V}{\partial \mathbf{x}} = 2(\bar{\mathbf{x}} - \mathbf{p}_{\text{target}})^\top$ is the gradient of the Lyapunov function with respect to the swarm state. This formulation ensures that the control input u is designed to minimize the tracking error while adhering to the system dynamics.

B. Control Barrier Functions Formulation

To ensure collision-free navigation while preserving swarm coherence, we employ CBFs with formally guaran-

teed safety properties. Consider the swarm's configuration space $\mathcal{X} \subset \mathbb{R}^{2N}$ and a set of safety constraints $\mathcal{S} \subset \mathcal{X}$ defined through continuously differentiable barrier functions $b\colon \mathcal{X} \to \mathbb{R}$. The safe set is characterized by $\mathcal{S} := \{\mathbf{x} \in \mathcal{X}\colon b(\mathbf{x}) \geq 0\}$.

For obstacle avoidance, we model obstacles as superellipses, which is a generalization of standard geometric primitives that provides shape flexibility through a single parameter. The m-th obstacle is defined by:

$$\left(\frac{\|x - c_x^{(m)}\|}{a^{(m)}}\right)^{n^{(m)}} + \left(\frac{\|y - c_y^{(m)}\|}{b^{(m)}}\right)^{n^{(m)}} \leq 1 \quad (6)$$

Here, $(c_x^{(m)}, c_y^{(m)})$ is the obstacle centroid, $a^{(m)}, b^{(m)} > 0$ are the semi-axes lengths, and $n^{(m)} \geq 0.3$ controls the convexity. Then we define a barrier function for each obstacle:

$$b_k(\mathbf{x}) = \left[\left(\frac{\|x - c_x^{(m)}\|}{a^{(m)} + r}\right)^{n^{(m)}} + \left(\frac{\|y - c_y^{(m)}\|}{b^{(m)} + r}\right)^{n^{(m)}}\right] - 1 \quad (7)$$

Here, r denotes the safety radius, generally equivalent to the agent's radius. This formulation ensures $b_k(\mathbf{x}) \geq 0$ implies collision avoidance between agent i and obstacle m. To guarantee forward invariance of the safe set $\mathcal{S}$, we enforce the CBF condition through Lie derivatives of the barrier functions along the system dynamics:

$$\sup_{u \in \mathcal{U}} [L_f b(\mathbf{x}) + L_g b(\mathbf{x}) u] + \alpha(b(\mathbf{x})) \geq 0 \quad (8)$$

Where $\alpha\colon \mathbb{R} \to \mathbb{R}$ is a Lipschitz continuous extended class $\mathcal{K}$ function. The Lie derivatives are computed as:

$$L_f b(\mathbf{x}) = \frac{\partial b}{\partial \mathbf{x}} f(\mathbf{x}), \quad L_g b(\mathbf{x}) = \frac{\partial b}{\partial \mathbf{x}} g(\mathbf{x}) \quad (9)$$

This formulation ensures that the control input u is designed to ensure the agent stays in the safety area while satisfying the system dynamics.

1) Global Control Formulation: The global control paradigm treats the swarm as a cohesive entity with collective dynamics governed by a unified control input $u \in \mathbb{R}^2$. This approach maintains swarm coherence by ensuring that all agents remain within a compact formation. In other words, the formation radius is defined as the maximum distance from any agent's position to the swarm's centroid, such that the group is tightly clustered.

To address potential feasibility issues in constrained environments, we formulate a relaxed CLF-CBF Quadratic Program (QP) with slack variable $\delta \geq 0$, under the assumption of a total of M obstacles:

$$\begin{aligned} \min_{u \in \mathcal{U}, \delta \geq 0} \quad & \|u\|^2 + \rho\delta^2 \\ \text{s.t.} \quad & L_f V + L_g V u + \varepsilon V \leq \delta \\ & L_f b_m + L_g b_m u + \alpha(b_m) \geq 0, \ \forall m \in \{1, ..., M\} \end{aligned} \quad (10)$$

As demonstrated in Fig. 4a-b and Fig. 5a-b on both the unicycle and bicycle models in the Static Sync formation, this centralized approach successfully navigates sparse obstacle configurations but exhibits a limitation: The cohesive formation cannot pass through narrow passages between adjacent obstacles due to the fixed safety margin.

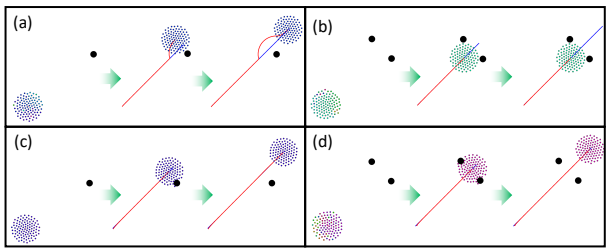

Fig. 4. Global and local control of unicycle swarmalators among obstacle-laden paths. (a) Global control enables the swarmalators to move around a single obstacle to a desired target. (b) Global control prohibits the collective from passing in between two close obstacles. (c-d) Local control enables the group to move in a straight line around one obstacle (c) and between two obstacles (d). The snapshots in each subfigure correspond to 2 s, 14 s, and 20 s.

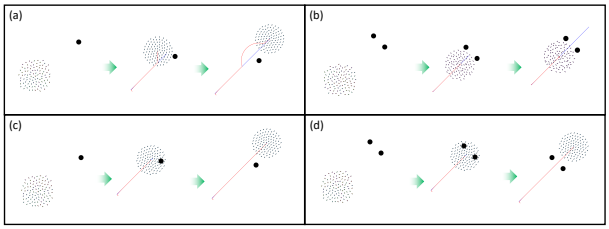

Fig. 5. Global and local control of bicycle swarmalators among obstacle-laden paths. (a) Global control enables the swarmalators to move around a single obstacle to a desired target. (b) Global control prohibits the collective from passing in between two close obstacles. (c-d) Local control enables the group to move in a straight line around one obstacle (c) and between two obstacles (d). The snapshots in each subfigure correspond to 2 s, 18 s, and 30 s.

2) Local Control Formulation: The local control strategy adopts a decentralized approach, where each agent i computes its control input u_i based on local neighborhood information. This requires two complementary safety constraints:

For each agent-obstacle pair (i, m), we define the pairwise barrier function:

$$b_{ik}(\mathbf{x}_i) = \left[\left(\frac{\|x_i - c_x^{(m)}\|}{a^{(m)} + r_i}\right)^{n^{(m)}} + \left(\frac{\|y_i - c_y^{(m)}\|}{b^{(m)} + r_i}\right)^{n^{(m)}}\right] - 1 \quad (11)$$

In continuous time with infinite updates, the swarmalator repulsion alone would prevent collisions, but in practice discrete timesteps and uncertain local velocity corrections can still permit overlaps, especially in crowded scenarios under local control. To address this, we introduce inter-agent CBFs, which enforce a hard, forward-invariant safety margin at each finite update, thereby converting the soft emergent spacing into a strict collision-avoidance guarantee under real-world control uncertainties. For each agent pair (i, j), we enforce:

$$b_{ij}(\mathbf{x}_i) = (x_i - x_j)^2 + (y_i - y_j)^2 - (r_i + r_j)^2, \quad \forall i, j \neq i \quad (12)$$

Each agent solves the decentralized QP:

$$\begin{aligned} \min_{u_i \in \mathcal{U}_i, \delta_i \geq 0} \quad & \|u_i\|^2 + \rho \delta_i^2 \\ \text{s.t.} \quad & L_f V_i + L_g V_i u_i + \varepsilon V_i \leq \delta_i \\ & L_f b_{im} + L_g b_{im} u_i + \alpha_1(b_{im}) \geq 0, \ \forall m \in \{1, ..., M\} \\ & L_f b_{ij} + L_g b_{ij} u_i + \alpha_2(b_{ij}) \geq 0, \ \forall j \neq i \end{aligned} \quad (13)$$

This formulation enables agents to independently negotiate complex obstacles while maintaining swarm connectivity, as shown in Fig. 4c-d and Fig. 5c-d. However, the constraint count per agent grows quadratically, creating scalability challenges for dense swarms in obstacle-rich environments. The computation complexity increased from $\mathcal{O}(\mathcal{N})$ to $\mathcal{O}(\mathcal{N}^2)$.

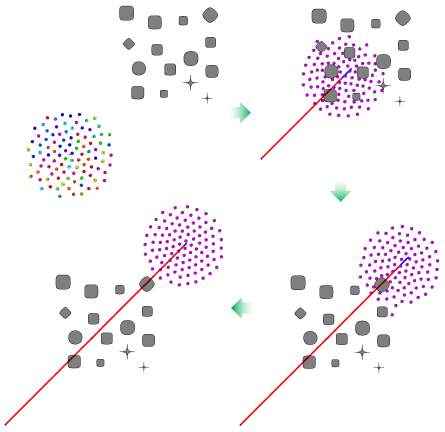

Fig. 6. Time evolution of the unicycle swarmalator system at different time instances with various obstacles in local control. The snapshots correspond to when 2 s, 10 s, 16 s, and 20 s have passed.

As illustrated in Fig. 6, we demonstrate how a swarm of unicycle swarmalators can navigate through an environment densely populated with diverse obstacles. By leveraging the combined capabilities of CLFs and CBFs, the system ensures collision-free trajectories with both obstacles and neighboring agents, all while maintaining the desired collective path. This fluid-like maneuverability highlights the effectiveness of the integrated CLF-CBF framework in guiding robotic swarms through complex terrains. In addition to obstacle avoidance, Fig. 6 shows that all agent phases converge to the same value even under local control and non-holonomic constraints.

C. Control Performance Evaluation

In this subsection, we evaluate the swarmalators' obstacle avoidance performance by examining the barrier function $b(\mathbf{x})$, which reflects each agent's proximity to obstacles. Fig. 7 shows how $b(\mathbf{x})$ varies over time for a single-obstacle scenario under both global and local control. Specifically, we test the omnidirectional and unicycle models in the same configuration as in Fig. 4a/c, and the bicycle model as depicted in Fig. 5a/c. Because the bicycle simulation is

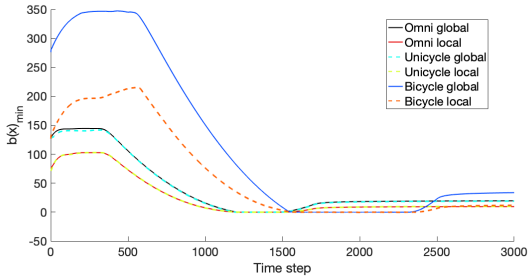

Fig. 7. Temporal evolution of the barrier function $b(\mathbf{x})$ for a single-obstacle scenario under both global and local control, comparing omnidirectional, unicycle, and bicycle models. $b(x)_{min} \geq 0$ denotes safety guarantees.

conducted in a larger domain, its $b(\mathbf{x})$ curve is generally higher. For the local control setting (detailed in Sec. VI-B.2), we record, at each time step, the minimum value of $b_{im}(\mathbf{x}_i)$ among all agents. During the initial few hundred time steps, the swarmalators gather and begin to follow the trajectory; as they approach the obstacle, $b(\mathbf{x})$ drops close to zero. Once the swarmalators cohesively bypass the obstacle (global control) or independently move around it (local control), $b(\mathbf{x})$ rises again until the agents reach their target position.

TABLE I
CONTROL PERFORMANCE METRICS

	Global Control		Local Control	
Dynamic	$b(\mathbf{x})_{\min}$	$d_{\min}$	$b(\mathbf{x})_{\min}$	$d_{\min}$
Omni	0.0012	0.0002	0.0001	0.0001
Unicycle	-0.1058	0.0820	-0.0424	0.0690
Bicycle	-0.3080	0.1765	-0.3481	0.0563

In Table I, the entry $b(\mathbf{x})_{\min}$ represents the lowest value of the barrier function $b(\mathbf{x})$ observed across the entire simulation, while $d_{\min}$ denotes the smallest true edge-to-edge distance between any agent and the obstacle. In our implementation, the obstacle radius used inside the control barrier function is defined slightly larger than the actual obstacle radius, so that even if $b(\mathbf{x})$ dips below zero due to dynamics coupling, the actual clearance $d(\mathbf{x})$ remains strictly positive. This ensures that collisions are avoided in all tested scenarios. We perform five trials for each setting and list the average values. From a theoretical standpoint, CBFs guarantee safety by maintaining $b(\mathbf{x}) \geq 0$. However, the negative $b(\mathbf{x})_{\min}$ values in Table I stem from how we couple the swarmalator model with robot dynamics in Section VI-B.2. One possible remedy is to employ an additional "safety filter," such as BarrierNet [31], which would explicitly enforce $b(\mathbf{x}) \geq 0$ at every step. While such filters can eliminate boundary violations, they typically require solving an additional optimization problem at each control step, which increases computational overhead especially for large swarms.

Nonetheless, the values of $d_{\min}$ in Table I confirm that collisions do not occur in practice. For the omnidirectional model, $b(\mathbf{x})$ remains strictly positive, indicating consistent satisfaction of the barrier constraints. In contrast, the unicycle model exhibits slight dips below zero due to orientation lag

when aligning with the desired heading; to mitigate this effect, we set the safety radius r to twice the agent's physical radius. Similarly, for the bicycle model, a larger turning radius led us to triple the agent's radius when defining r. These results validate that, despite brief negative values in the barrier function, $d_{\min} > 0$ at all times, ensuring that agents safely circumvent obstacles in all tested scenarios.

D. Object Transport

Ring-like collective behaviors in swarmalators can serve a practical purpose by enabling the swarm to encapsulate and transport an object. Specifically, when the agents form a closed circular formation around the target, they effectively "trap" the object within the swarm, allowing coordinated motion to move it from one location to another.

To illustrate this capability, we choose parameters $A = 1$, $B = 1.5$, $J = 1$, and $K = 0$. These values provide sufficient inter-agent spacing to accommodate an object, which has a radius of 0.5. As shown in Fig. 8, the object initially acts as an obstacle that the swarmalators must approach and surround via local control. Once the ring formation is established, the object is then treated as a payload that the group can push and guide along a desired trajectory. When an agent enters the object's contact radius, its velocity command is set to zero to anchor the payload, and the object is then given a kinematic velocity equal to 0.1 times the agent's velocity in the same direction, with the 10% factor modeling the object's light inertia. Here, the 0.1 scaling factor is a simplifying estimate of the weight ratio between an agent and the payload, modeling the case where an agent transfers its motion to a light object via pushing; extensions to heavier or attached payloads could incorporate more detailed dynamics (e.g., frictional contact or rigid coupling).

Both global and local control strategies can facilitate this form of collaborative transport while avoiding other obstacles along the way. In Fig. 8a, the combination of local and global control enables the swarm to bypass a circular obstacle, carrying the object safely around it. In Fig. 8b, local control is sufficient to guide the swarm through a narrow corridor, showcasing the flexible application of CBFs to achieve a smooth passage in constrained environments. Currently, the capture moment is determined by simulation timesteps; however, if the obstacle's position is known, it is also possible to automatically determine whether the object is encapsulated by comparing the obstacle's position with the centroid of the collective and applying a suitable threshold.

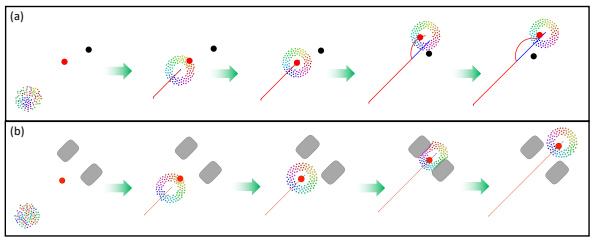

Fig. 8. Unicycle swarmalators transporting an object through global and local control. The red circle indicates the object to be transported, while the black circle and gray rectangles represent obstacles. (a) Combining Local Control and Global Control. (b) Using Local Control Only. Snapshots are shown at 2 s, 8 s, 12 s, 16 s, and 20 s.

To further quantify the swarm's transport capability, we ran a parametric study (shown in Fig. 9) varying both the number of agents and the passage width, while keeping the same object (radius = 0.5) and overall parameters as in Fig. 8b. We consider the task successful if the object's final position ends up within 0.2 units of the specified goal location. For the omnidirectional and unicycle models, the repulsion parameter is set to $B = 1.5$, whereas for the bicycle model we use $B = 5$. For the omnidirectional and unicycle cases (Fig. 9b-c), smaller swarms navigate narrow passages more successfully, since a high agent density can cause collisions with the object in very tight spaces. However, these smaller swarms fare worse when there is greater space between the obstacles, as they lack enough agents to robustly push the object along a desired direction. In contrast, the bicycle model's larger repulsion parameter $B = 5$ makes the agent ring sparser, so fewer agents can result in a lower success rate: the object is more prone to slipping out of the swarm ring. Overall, when the object is light enough that a single agent can move it, two main factors govern success: whether the swarm can physically pass through the gap, and whether the agent distribution is compact enough to keep the object enclosed without causing excessive collisions.

VII. Conclusion and Discussion

In this work, we integrated the classical swarmalator model with non-holonomic robot dynamics and demonstrated that, despite the kinematic constraints of unicycle and bicycle platforms, the system effectively reproduces the hallmark behaviors of omnidirectional swarmalators. By incorporating a two-dimensional control input into the swarmalator equations, we enabled the robots to steer and avoid collisions with obstacles, as well as encapsulate and transport objects through an obstacle-laden environment, all through global and local CLF-CBF-based strategies.

Our current validation relies on simulation, but the settings were chosen to closely reflect realistic robotic conditions. In particular, the control loop runs at a rate consistent with feasible onboard computation and communication, and control inputs are applied under a standard zero-order hold assumption (i.e., each command is held constant until the next update). Control velocities are bounded to match typical limits of small robots. Unicycle and bicycle kinematics are explicitly modeled to capture non-holonomic constraints. Furthermore, we inflate obstacle and agent radii slightly in the CBF formulation, ensuring that even under discretization, sensing noise, or localization error, the true clearance remains positive. We also confirmed that the CLF-CBF QPs solve comfortably within each control period for the swarm sizes tested. These considerations indicate that the reported behaviors are not artifacts of idealized simulation, but are representative of what could be realized in practice. We also note that, as a reactive method, the CBF controller may occasionally cause local deadlocks when the swarm becomes trapped between obstacles. Incorporating a planning

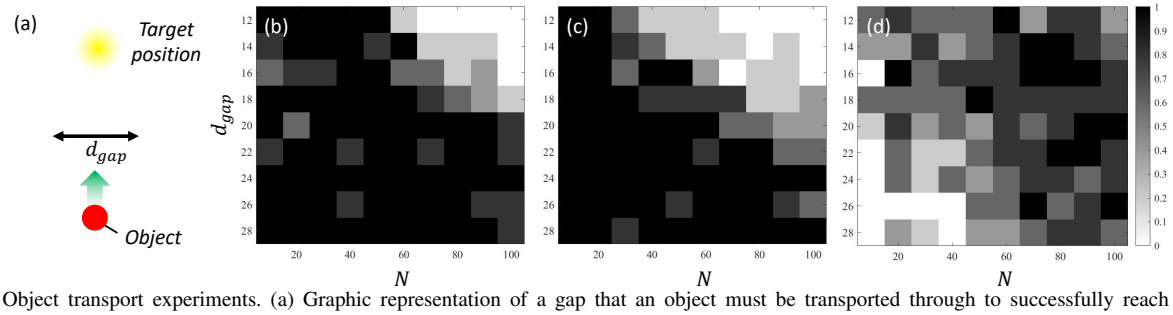

Fig. 9. Object transport experiments. (a) Graphic representation of a gap that an object must be transported through to successfully reach the target position. (b-d) Transport experiments across a parameter space varying the gap size (d_{gap}) and the number of agents (N) for the different dynamic models. (b) Omnidirectional. (c) Unicycle. (d) Bicycle. Color represents the percent of the 10 trials that resulted in successful object transport to the target position for each of the test cases.

or receding-horizon layer [32] could help mitigate this issue by adding predictive guidance, though such methods require additional planning infrastructure and higher computational cost, which we leave for future work.

While our current work focuses on navigation around and in between static objects, future research will explore more challenging scenarios such as dynamic or moving obstacles, incorporation of physical interactions (e.g., gravity, friction), and extensions into three-dimensional environments. These future directions will help us advance our control framework and enhance its robustness under increasingly realistic and demanding conditions.

REFERENCES

[1] Y. Katz, K. Tunstrøm, C. C. Ioannou, C. Huepe, and I. D. Couzin, "Inferring the structure and dynamics of interactions in schooling fish," *Proceedings of the National Academy of Sciences*, vol. 108, no. 46, pp. 18 720–18 725, 2011.

[2] W. Bialek, A. Cavagna, I. Giardina, T. Mora, E. Silvestri, M. Viale, and A. M. Walczak, "Statistical mechanics for natural flocks of birds," *Proceedings of the National Academy of Sciences*, vol. 109, no. 13, pp. 4786–4791, 2012.

[3] D. B. Kearns, "A field guide to bacterial swarming motility," *Nature reviews microbiology*, vol. 8, no. 9, pp. 634–644, 2010.

[4] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet, "Novel type of phase transition in a system of self-driven particles," *Physical review letters*, vol. 75, no. 6, p. 1226, 1995.

[5] A. Moiseff and J. Copeland, "Firefly synchrony: a behavioral strategy to minimize visual clutter," *Science*, vol. 329, no. 5988, pp. 181–181, 2010.

[6] Z. Néda, E. Ravasz, Y. Brechet, T. Vicsek, and A.-L. Barabási, "The sound of many hands clapping," *Nature*, vol. 403, no. 6772, pp. 849–850, 2000.

[7] W. A. MacKay, "Synchronized neuronal oscillations and their role in motor processes," *Trends in cognitive sciences*, vol. 1, no. 5, pp. 176–183, 1997.

[8] S. H. Strogatz, "From kuramoto to crawford: exploring the onset of synchronization in populations of coupled oscillators," *Physica D: Nonlinear Phenomena*, vol. 143, no. 1-4, pp. 1–20, 2000.

[9] K. P. O'Keeffe, H. Hong, and S. H. Strogatz, "Oscillators that sync and swarm," *Nature communications*, vol. 8, no. 1, pp. 1–13, 2017.

[10] S. Ceron, K. O'Keeffe, and K. Petersen, "Diverse behaviors in non-uniform chiral and non-chiral swarmalators," *Nature Communications*, vol. 14, no. 1, p. 940, 2023.

[11] K. O'Keeffe, S. Ceron, and K. Petersen, "Collective behavior of swarmalators on a ring," *Physical Review E*, vol. 105, no. 1, p. 014211, 2022.

[12] K. O'Keeffe and C. Bettstetter, "A review of swarmalators and their potential in bio-inspired computing," *Micro-and Nanotechnology Sensors, Systems, and Applications XI*, vol. 10982, pp. 383–394, 2019.

[13] A. Barciś and C. Bettstetter, "Sandsbots: Robots that sync and swarm," *IEEE Access*, vol. 8, pp. 218 752–218 764, 2020.

[14] R. Beattie, S. Ceron, and D. Rus, "Realizing emergent collective behaviors through robotic swarmalators," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025.

[15] S. Ceron, G. Gardi, K. Petersen, and M. Sitti, "Programmable self-organization of heterogeneous microrobot collectives," *Proceedings of the National Academy of Sciences*, vol. 120, no. 24, p. e2221913120, 2023.

[16] S. Ceron, W. Xiao, and D. Rus, "Reciprocal and non-reciprocal swarmalators with programmable locomotion and formations for robot swarms," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 12 233–12 239.

[17] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *Ieee Access*, vol. 6, pp. 28 573–28 593, 2018.

[18] F. Chen, W. Ren *et al.*, "On the control of multi-agent systems: A survey," *Foundations and Trends® in Systems and Control*, vol. 6, no. 4, pp. 339–499, 2019.

[19] V. M. Gonçalves, D. Chaikalis, A. Tzes, and F. Khorrami, "Safe multi-agent drone control using control barrier functions and acceleration fields," *Robotics and Autonomous Systems*, vol. 172, p. 104601, 2024.

[20] R. Cheng, M. J. Khojasteh, A. D. Ames, and J. W. Burdick, "Safe multi-agent interaction through robust control barrier functions with learned uncertainties," in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 777–783.

[21] X. Tan and D. V. Dimarogonas, "Distributed implementation of control barrier functions for multi-agent systems," *IEEE Control Systems Letters*, vol. 6, pp. 1879–1884, 2021.

[22] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE control systems letters*, vol. 1, no. 2, pp. 310–315, 2017.

[23] W. Xiao and C. Belta, "Control barrier functions for systems with high relative degree," in *2019 IEEE 58th conference on decision and control (CDC)*. IEEE, 2019, pp. 474–479.

[24] K. P. Tee, S. S. Ge, and E. H. Tay, "Barrier lyapunov functions for the control of output-constrained nonlinear systems," *Automatica*, vol. 45, no. 4, pp. 918–927, 2009.

[25] S. Boyd, "Convex optimization," *Cambridge UP*, 2004.

[26] Y. Kuramoto, "Self-entrainment of a population of coupled non-linear oscillators," in *International symposium on mathematical problems in theoretical physics*. Springer, 1975, pp. 420–422.

[27] T. G. Molnar, R. K. Cosner, A. W. Singletary, W. Ubellacker, and A. D. Ames, "Model-free safety-critical control for robotic systems," *IEEE robotics and automation letters*, vol. 7, no. 2, pp. 944–951, 2021.

[28] A. J. Taylor, P. Ong, T. G. Molnar, and A. D. Ames, "Safe backstepping with control barrier functions," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 5775–5782.

[29] M. H. Cohen, T. G. Molnar, and A. D. Ames, "Safety-critical control for autonomous systems: Control barrier functions via reduced-order models," *Annual Reviews in Control*, vol. 57, p. 100947, 2024.

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

[31] W. Xiao, T.-H. Wang, R. Hasani, M. Chahine, A. Amini, X. Li, and D. Rus, "Barriernet: Differentiable control barrier functions for learning of safe robot control," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 2289–2307, 2023.

[32] L. Sforni, G. Notarstefano, and A. D. Ames, "Receding horizon cbf-based multi-layer controllers for safe trajectory generation," in *2024 american control conference (ACC)*. IEEE, 2024, pp. 4765–4770.