sciendo

VOLUME 19, N° 1 2025
Journal of Automation, Mobile Robotics and Intelligent Systems

Formation Control of Multi-agent Nonlinear Systems using The State-Dependent Riccati Equation

Submitted: 10th January 2024; accepted: 23rd July 2024

Saeed Rafee Nekoo

DOI: 10.14313/jamris-2025-003

Abstract:
This work investigates the formation control of a multi-agent robotic system via the state-dependent Riccati equation (SDRE). The system's agents interact with each other and follow a leader in point-to-point motion control (regulation). The number of agents is unlimited in conventional multi-agent formation control considering a complex dynamic for each agent, though the complexities of the algorithms usually result in small-scale simulations. Here a formalism is proposed that considers fully coupled nonlinear dynamics for robotic systems in multi-agent system formation control with a large number of agents. The interaction of the agents with each other and obstacle avoidance are embedded in the design through the weighting matrix of states in the SDRE. The input constraint also limits the actuators to create a more realistic scenario. Two dynamical systems have been modeled and simulated in this work: wheeled mobile robots (WMR) and multirotor unmanned aerial vehicles (UAVs). The simulation results show success in the implementation of a total of 1,089 agents in the desired square formation shape in the UAV case study, and a figure of 45 agents and 1,050 differential wheeled mobile robots in the circular desired shape, considering obstacle avoidance and also collision avoidance between the agents.

Keywords: *Multi-agent, Formation, Control, SDRE, Robotics.*

1. Introduction

This paper presents research on controlling a highly-populated multi-agent system within the framework of the state-dependent Riccati equation (SDRE), which is a nonlinear optimal closed-loop control policy. A multi-agent system consists of a leader and a population of followers, playing the role of agents. The agents in a swarm system show a collective behavior, with interactions achieving a common goal [1]. The previous reports on swarm robotics have been quite diverse in subject matter: technological aspects of swarm robotics [2, 3]; arrangement, formation, and behavior [4–6]; and bio-inspiration [7, 8], among others.

The application of swarm robotics has been reported in many works, including micro-robotics: the interaction and physical contact between the agents [9–11], nano-swarms for delivering medicine or performing an operation in human body [12], search and rescue [13–15], monitoring and data collection [16, 17], etc.

Multiple decision-making agents with auto—nomous operation, social ability interaction, reactivity perception, and pro-activeness exhibition, were motivated by complex inherently distributed systems to increase robustness. Multi-agent and swarm systems are similar in the sense that they require multiple agents that communicate and cooperate. Each agent in multi-agent systems can do some meaningful part of a task; however, in swarm systems, we would expect each agent to be too unaware of its environment to possibly even know what is going on around it. This emphasizes the existence of a large number of agents, and promotes scalability, soft collisions and physical contacts of the swarm in migration. This might not be the case for swarms in nature, i.e., in birds, which do not collide, and also do not receive a command from leader [18]. Here, in this current work, the modeling is not based on bio-inspiration, since we are far away from such sophisticated design in mother nature.

Multi-agent systems, or cooperative robots, can usually be fit into these two categories: systems that considered complex dynamics for each agent, though the number of agents is small [19–24], and systems with simple dynamics or only kinematics for each agent when the number of agents is large [25–32]. For example, Yang et al. presented a formation control strategy based on a stable control law to guide the agents towards a constraint region and simulated it for a total of 155 agents [32]. Collision avoidance and obstacle avoidance between the agents are also two important characteristics of the formation methods. Wang and Xin presented an optimal control approach for formation control taking into consideration obstacle avoidance for multiple unmanned aerial vehicle (UAV) systems [33]. Kumar et al. presented a velocity controller for a swarm of UAVs with obstacle avoidance capability [34]. Park and Yoo proposed a connectivity-based formation control with collision-avoidance properties for a swarm of unmanned surface vessels [35, 36].

Table 1. A detailed report on the population of agents in SDRE in previous, as literature compared to this work.

context	Ref.	No. of agents, type of agent
multi-agent leader-follower formation	[37]	5, single-integrator
	[38]	2, mobile robot
	[40]	2, spacecraft
	[41]	5, single-integrator
	[42]	2, spacecraft
	[43]	3, mobile robot
	[44]	2, aircraft
	[45]	3, spacecraft
	[46]	2, spacecraft
	[47]	2, satellite
	[39]	1024, satellite
consensus control	[22]	10, crane
cooperative multi-agents	[48]	2, manipulator
	[49]	2, manipulator
	[50]	3, spacecraft
	[19]	4, manipulator
	[51]	4, UAV
	[52]	3, mobile robot
multi-agents leader-follower	this work	**1,089, UAV**
	this work	**45, mobile robot**
	this work	**1,050, mobile robot**

The application of the swarm regulation control is devoted to multi-copter UAV formation control for a large number of agents–1,089–to show the capabilities of the proposed approach. Swarm control using augmented methods such as graph theory and multi-layer network design is a useful approach; however, in this work, the pure capacity of the SDRE is used to control a highly populated multi-agent system. This will simplify the design and limit the formulation to pure SDRE. The advantages of this point of view result in controlling large-scale systems with complex dynamics. A fully coupled six-degree-of-freedom (DoF) nonlinear model of the system is considered (12 state variables). The second case study is a leader-follower system of 45-wheeled mobile robots (WMRs) with non-holonomic and holonomic constraints of wheels. A smaller number of robots was considered to highlight the effectiveness of obstacle and collision avoidance. SDRE has been used in both multi-agent systems [37–39], and leader-follower control [22, 40–47]. The details of the number of agents using SDRE are reported on Table 1. The majority of these studies focused on the control and behavior of the leader-follower system, first considering two agents and then sometimes increasing the number of agents to 5 or 10. The only large populated system using SDRE control included 1,024 satellites; obstacle avoidance and collision avoidance between agents were not considered [39]. The large distance between the agents and the pattern of the satellites did not necessitate collision/obstacle avoidance. In this work, a more mature version of multi-agent SDRE [39] is presented to add those characteristics, since the agents are working close together in both solved examples.

The swarm formulation in control methods could be solved by graph theory for the formation design of the whole swarm; in that case, another controller must control the agents, individually by checking the connection for the agents through multiple layers [53–55]. The design of the formation using graph theory, and adapting a controller for handling the multi-agent system, might pose difficulty in implementation because a controller must include other layers of designs through external methods.

The reported number of agents was 16 satellites [53], 6 agents [54], and 90 agents [55]. The use of the Kronecker delta function for the selection of information from a particular agent limits the application of the swarm multi-agent formulation to a limited number. In the SDRE method particularly, the solution to the Riccati equation might become overly complex if the SDRE should be solved for the entire swarm. In this approach, the gain of the targeted agent would be collected from the overall matrix of the swarm. Again, this approach would limit the implementation of the formulation for a large number of agents. The reported SDRE agents, using consensus control and graph theory, were 10 [22] and 2 [38], respectively.

The focus of this work is large-scale multi-agent systems that need simplicity in their design. This simplicity does not imply a simple linear controller; on the contrary, the SDRE is a nonlinear sub-optimal controller with a relatively complex solution. However, the practical approach to handle the multi-agent system relies solely on the SDRE's capabilities itself, and uses the weighting matrix of the states to handle distance constraints and obstacle avoidance. As a result, there would not be further complexity for practical implementation raised by the interaction of the SDRE with another layer or graph theory. This can be viewed as independent SDRE controllers working with neighbor components and their leader through a weighting matrix, which is inside the umbrella of the SDRE, not another system augmented by the multi-agents or other complementary methods. This is the key to large-scale modeling and simulation–which is simplicity in the communication of the multi-agent system and reliance on the capacities of the SDRE itself.

The main contribution of this work is the presentation of a control structure for forming a multi-agent system with complex dynamics with obstacle and collision avoidance between the agents. Complex systems with a large number of agents are seldom reported; this work presents a suitable structure for that. This work adds collision and obstacle avoidance to SDRE multi-agent system control, motivated by the example set by [39], which reported solely multi-agent system SDRE without contact prevention. For very crowded systems, contact between the agents has been both expected and considered in the literature [30, 32], but this work proposes a method that adds distance between the agents to reduce collisions during the control task.

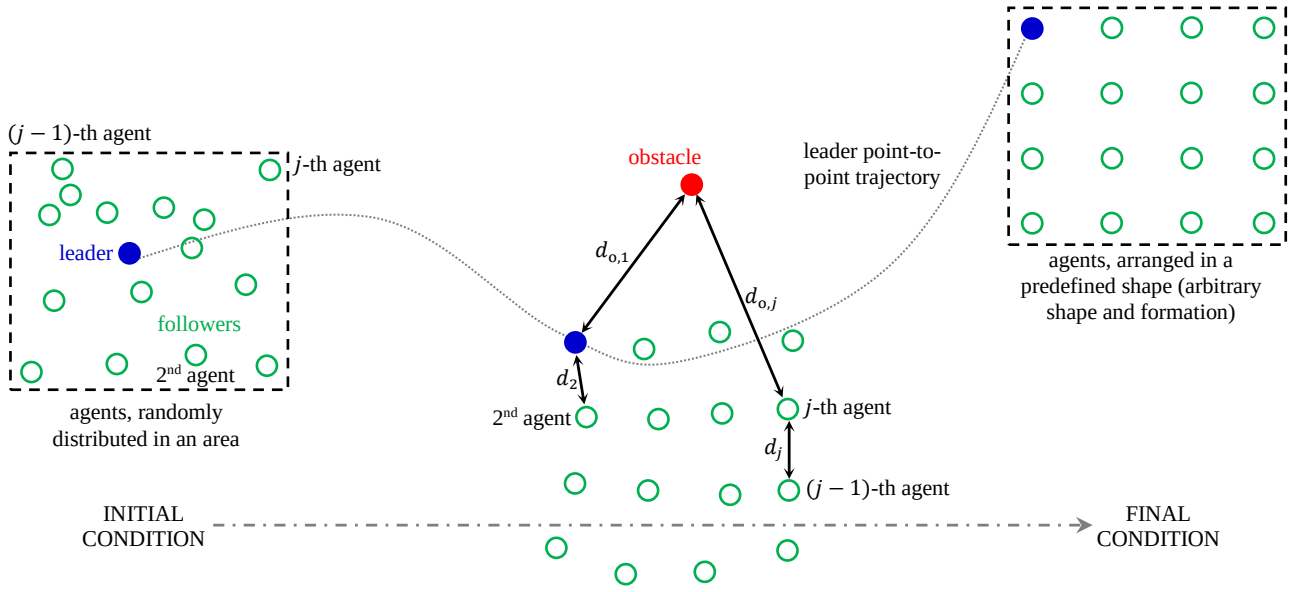

Figure 1. A schematic view of the multi-agent system, moving from an initial randomly-distributed form to a final desired shape with arranged agents in an environment with an obstacle. The 2D representation is intended to show more detail; its formation is general and could be applied to 3D formation as well.

The distance-based arrangement of agents, and avoiding collision between them, is basically the idea of multi-agent system control methods that use graph theory, potential field method, multi-layer designs, and the like; here in this work, the distribution, as well as obstacle, and collision avoidance, are structured within the SDRE formulation–specifically, the weighting matrix of states.

Section 2 presents the control structure of the SDRE and multi-agent system modeling. Section 3 describes the dynamics of the two case studies of this work, UAV, and WMR. The simulation results are presented in Section 4 and the concluding remarks are summarized in Section 5.

2. Control Structure: SDRE for Multi-agent Systems

2.1. Conventional Controller: A Brief Preliminary Formulation

Consider N number of agents, randomly distributed in a predefined area, with continuous-time nonlinear systems and affine-in-control properties. The j-th agent's dynamic is represented in state-space form as:

$$\dot{\mathbf{x}}_j(t) = \mathbf{f}_j(\mathbf{x}_j(t)) + \mathbf{g}_j(\mathbf{x}_j(t), \mathbf{u}_j(t)), \qquad (1)$$

where $\mathbf{x}_j(t) \in \mathbb{R}^n$ is the state vector and $\mathbf{u}_j(t) \in \mathbb{R}^m$ is the input vector of the j-th agent. $\mathbf{f}_j(\mathbf{x}_j(t)) : \mathbb{R}^n \to \mathbb{R}^n$ and $\mathbf{g}_j(\mathbf{x}_j(t), \mathbf{u}_j(t)) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ in (1) are vector-valued smooth functions that satisfy the local Lipschitz condition, with equilibrium point $\mathbf{f}_j(\mathbf{0}) = \mathbf{0}$. The nonlinear system (1) is transformed into the state-dependent coefficient (SDC) parameterization:

$$\dot{\mathbf{x}}_j(t) = \mathbf{A}_j(\mathbf{x}_j(t))\mathbf{x}_j(t) + \mathbf{B}_j(\mathbf{x}_j(t))\mathbf{u}_j(t), \qquad (2)$$

where $\mathbf{A}_j(\mathbf{x}_j(t)) : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ and $\mathbf{B}_j(\mathbf{x}_j(t)) : \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are held.

The SDC parameterization (2) is referred to as apparent linearization; however, it preserves the nonlinearity of the system. For example, consider an over-actuated single-degree-of-freedom system in state-space form with $n = 2$ and $m = 2$, then the system vectors

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} x_2 \\ x_1 x_2^2 \cos x_2 + x_1 e^{x_1} \end{bmatrix},$$
$$\mathbf{g}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} 0 \\ x_2 u_1 + x_1 x_2 u_2 \end{bmatrix},$$

can present a set of SDC matrices as

$$\mathbf{f}(\mathbf{x}) = \underbrace{\begin{bmatrix} 0 & 1 \\ e^{x_1} & x_1 x_2 \cos x_2 \end{bmatrix}}_{\mathbf{A}(\mathbf{x})} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$
$$\mathbf{g}(\mathbf{x}, \mathbf{u}) = \underbrace{\begin{bmatrix} 0 & 0 \\ x_2 & x_1 x_2 \end{bmatrix}}_{\mathbf{B}(\mathbf{x})} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

The result of multiplication of $\mathbf{A}(\mathbf{x})\mathbf{x}$ and $\mathbf{B}(\mathbf{x})\mathbf{u}$ will be the same original values of $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x}, \mathbf{u})$; hence, the SDRE method is a nonlinear optimal control that keeps the nonlinearities in the control structure. Another point on SDC matrices is that they are not unique representations, and can show numerous versions, such as:

$$\mathbf{f}(\mathbf{x}) = \underbrace{\begin{bmatrix} 0 & 1 \\ e^{x_1} + x_2^2 \cos x_2 & 0 \end{bmatrix}}_{\mathbf{A}(\mathbf{x})} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$
$$\mathbf{g}(\mathbf{x}, \mathbf{u}) = \underbrace{\begin{bmatrix} 0 & 0 \\ x_2 + x_1 x_2 \frac{u_2}{u_1} & 0 \end{bmatrix}}_{\mathbf{B}(\mathbf{x},\mathbf{u})} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}.$$

$\mathbf{B}(\mathbf{x},\mathbf{u})$ in the second set of the SDC matrices is not ideal because it changes the problem into one with non-affine parameterization and singularity in the case of $u_1 = 0$. The limitation and proper selection of SDC matrices can be summarized in controllability assumption, as follows.

Assumption 1 *The pair of* $\{\mathbf{A}_j(\mathbf{x}_j(t)), \mathbf{B}_j(\mathbf{x}_j(t))\}$ *is a completely controllable parameterization of nonlinear affine-in-control system (1) for the states of the j-th agent* $\mathbf{x}_j(t) \in \mathbb{R}^n$ *in* $t \in [0,\infty)$ *[56].*

The control objective is to minimize the cost function:

$$J = \frac{1}{2}\int_0^\infty \{\mathbf{x}_j^\top(t)\mathbf{Q}_j(\mathbf{x}_j(t))\mathbf{x}_j(t) + \mathbf{u}_j^\top(t)\mathbf{R}_j(\mathbf{x}_j(t))\mathbf{u}_j(t)\}\,\mathrm{d}t, \quad (3)$$

where $\mathbf{Q}_j(\mathbf{x}_j(t)) : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ and $\mathbf{R}_j(\mathbf{x}_j(t)) : \mathbb{R}^n \to \mathbb{R}^{m\times m}$ are weighting matrices of states and inputs of j-th agent, respectively. $\mathbf{Q}_j$ is symmetric positive semi-definite and $\mathbf{R}_j$ is symmetric positive definite.

Assumption 2 *The pair of* $\{\mathbf{A}_j(\mathbf{x}_j(t)), \mathbf{Q}_j^{1/2}(\mathbf{x}_j(t))\}$ *is a completely observable parameterization of the nonlinear affine-in-control system* (1) *for the states of j-th agent* $\mathbf{x}_j(t) \in \mathbb{R}^n$ *in* $t \in [0,\infty)$*, where* $\mathbf{Q}_j^{1/2}$*, in Eq. (3), is Cholesky decomposition of* $\mathbf{Q}_j$ *[57].*

The control law of the SDRE is [56]:

$$\mathbf{u}_j(t) = -\mathbf{R}_j^{-1}(\mathbf{x}_j(t))\mathbf{B}_j^\top(\mathbf{x}_j(t))\mathbf{K}_j(\mathbf{x}_j(t))\mathbf{x}_j(t), \quad (4)$$

where $\mathbf{K}_j(\mathbf{x}_j(t))$ is the positive-definite symmetric gain of the control law. The gain $\mathbf{K}_j(\mathbf{x}_j(t))$ is the solution to the state-dependent Riccati equation:

$$\mathbf{A}_j^\top(\mathbf{x}_j(t))\mathbf{K}_j(\mathbf{x}_j(t)) + \mathbf{K}_j(\mathbf{x}_j(t))\mathbf{A}_j(\mathbf{x}_j(t)) + \mathbf{Q}_j(\mathbf{x}_j(t)) - \mathbf{K}_j(\mathbf{x}_j(t))\mathbf{B}_j(\mathbf{x}_j(t)) \mathbf{R}_j^{-1}(\mathbf{x}_j(t))\mathbf{B}_j^\top(\mathbf{x}_j(t))\mathbf{K}_j(\mathbf{x}_j(t)) = \mathbf{0}. \quad (5)$$

Without loss of generality, the input law (4) is transformed to:

$$\mathbf{u}_j(t) = -\mathbf{R}_j^{-1}(\mathbf{x}_j(t))\mathbf{B}_j^\top(\mathbf{x}_j(t))\mathbf{K}_j(\mathbf{x}_j(t))\mathbf{e}_j(t),$$

for controlling the system to a desired condition in which $\mathbf{e}_j(t) = \mathbf{x}_j(t) - \mathbf{x}_{\mathrm{des},j}(t)$ where $\mathbf{x}_{\mathrm{des},j}(t)$ is the desired state vector of j-th agent.

2.2. Formation Control

The definition of formation control is introduced by a set of time-varying desired conditions for the agents to follow the leader of the group. The first agent, $j = 1$, is the leader and the rest of the agents are followers, $j = 2,...,N$. The leader is regulated (point-to-point motion) from an initial condition, $\mathbf{x}_1(0)$, to the desired condition, $\mathbf{x}_{\mathrm{des},1}(t_\mathrm{f})$, in which t_f is the final time of the control task. The schematic of the distribution of the agents, and the migration from the initial to the final condition, are illustrated in Fig. 1.

The rest of the agents follow the leader while they try to keep the predefined geometric formation during the task and sit at the exact predefined geometric formation at the end of the task:

$$\mathbf{x}_{\mathrm{des},j}(t) = \mathbf{x}_1(t) - \mathbf{x}_{\mathrm{f},1} + \mathbf{x}_{\mathrm{f},j}, \qquad j = 2,...,N, \quad (6)$$

where $\mathbf{x}_{\mathrm{f},j}$ represents the desired coordinates (constant values) for the j-th agent in the predefined formation shape. Only the leader has a constant (time-invariant) desired condition. Equation (6) shows that the time-varying desired conditions of the followers are fed by the online position of the leader and the constant desired formation. That means the followers try to keep the formation of the overall point-to-point trajectory of the leader from the initial to the final condition.

Remark 1 *It should be pointed out that* $\mathbf{x}_{\mathrm{f},j}$*, is a state vector, and only the first two or three states are different from zero (those that represent the configuration space). The rest of the desired states could be set to zero. Common dynamics of multirotor UAVs usually have six degrees of freedom that provide 12 states for the system. For the UAVs, only three position states of Cartesian coordinate* (x, y, z) *must be considered in the formation. The orientation and velocity states might be independently controlled.*

2.3. Collision Avoidance and Obstacle Avoidance Between Agents

The weighting matrix for the states of the SDRE controller can include nonlinear functions of state variables in its components. This property provides obstacle avoidance in regulation control based on artificial potential field method [52, 58, 59]. The distance between an obstacle and an agent is:

$$d_{\mathrm{o},j}(t) = \|\mathbf{x}_{\mathrm{p},j}(t) - \mathbf{x}_\mathrm{o}\|_2, \quad (7)$$

for $j = 1,...,N$ where $\mathbf{x}_{\mathrm{p},j}(t)$ is a part of state vector, which includes the position coordinates, $\mathbf{x}_{\mathrm{p},j}(t) \in \mathbf{x}_j(t)$, and has $\mathbf{x}_\mathrm{o}$ as the position of the obstacle.

The next step is to define the distance between two consecutive agents for collision avoidance. The distance between the j-th and $(j-1)$-th agent is:

$$d_j(t) = \|\mathbf{x}_{\mathrm{p},j}(t) - \mathbf{x}_{\mathrm{p},j-1}(t)\|_2, \quad (8)$$

for $j = 2,...,N$, it should be noted that the leader does not need to include collision avoidance terms. The collision avoidance approach in Eq. (8) only avoids the impact between two consecutive agents. The topology of the agents is defined through the distributed pattern (6) where the main character is the time-varying position of the leader. Evidently, this pattern and leader information does not provide enough safety in the migration of the multi-agent system. Hence, the collision avoidance term was introduced (8) to add another safety condition for migration without impact. This feature creates communication between two consecutive agents for reading and sending the position data.

The index is assigned to a given agent randomly; as a result, two consecutive agents might not be neighbors at the initial condition and could even be far away from each other. This initialization might raise issues. The collision avoidance between neighbor agents (8) might not be effective and collision might happen between j-th agent and other members. So, it is not assumed that agents j and $j-1$ are closed; however, starting the regulation control, they follow the leader and the pattern, and they merge to find their neighbor agents in a short period. To avoid impact and collision in a global scheme, the following terms will be considered as well.

To include the collision avoidance between the j-th agent and all previous members, the following condition is introduced:

$$d_{s,j}(t) = \begin{cases} d_{\text{tot},k,j}(t), & d_{\text{tot},k,j}(t) < r_{\min}, \\ 1, & d_{\text{tot},k,j}(t) \geq r_{\min}, \end{cases} \tag{9}$$

for $j = 3, ..., N$, and $k = 1, ..., j-1$ where $d_{\text{tot},k,j}(t) = \|\mathbf{x}_{p,j}(t) - \mathbf{x}_{p,k}(t)\|_2$, is the distance between the j-th agent and the k-th agent, and $r_{\min}$ is the minimum collision avoidance safety distance. The pseudocode of Eq. (9) is presented as

for $j = 3, ..., N$,
  for $k = 1, ..., j-1$,
    $d_{\text{tot},k,j}(t) = \|\mathbf{x}_{p,j}(t) - \mathbf{x}_{p,k}(t)\|_2$,
    if $d_{\text{tot},k,j}(t) < r_{\min}$,
      $d_{s,j}(t) = d_{\text{tot},k,j}(t)$,
    else,
      $d_{s,j}(t) = 1$,
    end,
  end,
end.

Equation (9) is updated at each time step of simulation when all the agents are moving from the initial to the final condition. In cases in which the j-th agent is close to one agent, the term $d_{s,j}(t)$ is updated in (9); otherwise $d_{s,j}(t) = 1$. The different distance terms ($d_{o,j}(t)$, $d_j(t)$, and $d_{s,j}(t)$) will be used in Section 2.4 to set the obstacle and collision avoidance strategy based on the modification of $\mathbf{Q}_j(\mathbf{x}_j(t))$ matrix.

2.4. Definition of Weighting Matrix

The state-dependent weighting matrices of the SDRE are one of the advantages of this method over a linear quadratic regulator (LQR) controller. The weighting matrices of the LQR are constant and cannot include parameters related to trajectories such as–the introduced distance terms in Section 2.3–for obstacle avoidance, collision avoidance, or adaptive designs. The definitions of the weighting matrices for the leader and followers are different in order to gain different motion speeds. The leader must be slower and the followers must be faster to track the leader properly. The following definition of the weighting matrix will provide for this condition.

The weighting matrix of states for the leader only includes obstacle avoidance:

$$Q_{m,m,1}(\mathbf{x}_1(t)) = 1 + \frac{1}{d_{o,1}(t)}, \tag{10}$$

where $Q_{m,m,1}(\mathbf{x}_1(t))$ is the m-th diagonal component of $\mathbf{Q}_1(\mathbf{x}_1(t))$ related to m-th position state; m could be a state variable of leader's dynamics, i.e. the second state, generated by $Q_{2,2,1}(\mathbf{x}_1(t))$. It should be noted that only a few diagonal components of $\mathbf{Q}_1(\mathbf{x}_j(t))$ are responsible for obstacle avoidance. For the example presented in Remark 1, only the first two diagonal components of the weighting matrix include (10), and the rest of the 10 components are tuned conventionally. The UAV position variables are indeed defined in X, Y, Z, though the obstacle avoidance terms are set only for X, Y; the results were satisfactory. The choice is with the designer, and in some cases, it can also be assigned for all three axes in 3D platforms.

Remark 2 *If we notice the term* $1 + \frac{1}{d_{o,1}(t)}$ *in* (10), $\frac{1}{d_{o,1}(t)}$ *might gain a very small value if the distance between the obstacle and the leader is far. The constant term "1" in* $Q_{m,m,1}(\mathbf{x}_1(t))$ *makes sure that the velocity of the leader is properly defined when* $\frac{1}{d_{o,1}(t)} \to 0$.

The followers j-th must avoid obstacles, as well as the collision between the other agents, through the matrix $\mathbf{Q}_j(\mathbf{x}_j(t))$:

$$Q_{m,m,j}(\mathbf{x}_j(t)) = \frac{1}{d_{o,j}(t)} + \frac{1}{d_j(t)} + \frac{1}{d_{s,j}(t)}, \tag{11}$$

for $j = 2, ..., N$. If the distance terms become small, which shows there is a probability of collision between agents or obstacles and an agent, the diagonal component of $\mathbf{Q}$ gets bigger and that state deviates the trajectory of the system (in other words, it avoids the collision). Assigning all the position states for an obstacle, or between collision avoidance speeds up all the states simultaneously and the collision avoidance fails. More details could be revisited in [52]. The following is helpful to clarify this point. Imagine a mobile robot moves forward in a straight line and there is an obstacle on the way. The actuators are two motors on the wheels. If the obstacle avoidance mechanism commands both wheels to speed up at the same time, the robot just moves faster in a straight line. In order to move the robot to one side, only one wheel must rotate faster! Therefore, avoidance functions are introduced only in one or two components related to the Cartesian coordinates.

In this section, three different distance terms were introduced: $d_{o,j}(t)$, $d_j(t)$, and $d_{s,j}(t)$. It must be noted that these distance terms will be applied in the weighting matrix of states, using the artificial potential field method. The distance terms increase the value of the weighting matrix, leading the agent to avoid the collision.

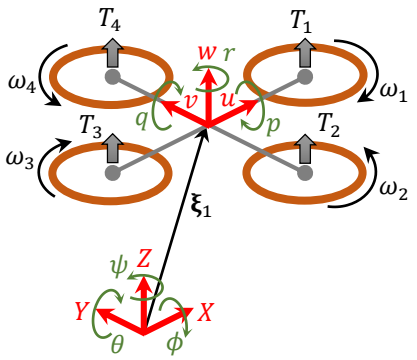

Figure 2. The schematic view and axis definition of a quadrotor.

Each distance term plays its own role in the migration of the multi-agent system. $d_{o,j}(t)$ avoids collision of the agent with the obstacle in the environment; $d_j(t)$ avoids collision of the two neighbor agents; and in case of the proximity of any previous agents to current agent, $d_{s,j}(t)$ will be activated to avoid the collision. In summary, the three components will contribute to the safety in case of collision of the agents with the obstacle and with other agents.

The definition of the rest of the components of the weighting matrix of states, as well as the inputs, could be done based on conventional tuning methods and rules [60].

3. System Dynamics and SDC Parameterization

3.1. Unmanned Aerial Vehicle

The dynamic equation of one quadrotor unmanned aerial vehicle (UAV) is presented in this section. The schematic view of the system and the configuration of the rotors are illustrated in Fig. 2. The agents are identical and share the same dynamics. The generalized coordinates of one system are $\{x_{c,j}(t), y_{c,j}(t), z_{c,j}(t), \phi_j(t), \theta_j(t), \psi_j(t)\}$, where $\xi_{1,j}(t) = [x_{c,j}(t), y_{c,j}(t), z_{c,j}(t)]^\top$(m) defines the center-of-mass (CoM) of the j-th UAV in the Cartesian coordinate system, and $\xi_{2,j}(t) = [\phi_j(t), \theta_j(t), \psi_j(t)]^\top$(rad) are its Euler angles. The linear and angular velocities of the drone are presented $\upsilon_{1,j}(t) = [u_j(t), v_j(t), w_j(t)]^\top$(m/s) and $\upsilon_{2,j}(t) = [p_j(t), q_j(t), r_j(t)]^\top$(rad/s), respectively. The state vector of the system is defined as:

$$\mathbf{x}_j(t) = [\xi_{1,j}^\top(t), \xi_{2,j}^\top(t), \upsilon_{1,j}^\top(t), \upsilon_{2,j}^\top(t)]^\top. \tag{12}$$

The kinematics relations

$$\begin{aligned} \dot{\xi}_{1,j}(t) &= \mathbf{R}_{ZYX,j}(\xi_{2,j}(t))\upsilon_{1,j}(t), \\ \dot{\xi}_{2,j}(t) &= \mathbf{T}_j(\xi_{2,j}(t))\upsilon_{2,j}(t), \end{aligned} \tag{13}$$

are held for the quadrotor system. Considering that the quadrotor system flies based on small changes in the Euler angles, it is not supposed to perform aggressive and aerobatic maneuvers, and the hovering state is assumed for the flight condition.

Therefore, the derivative of state-vector (12) generates

$$\dot{\mathbf{x}}_j(t) = [\dot{\xi}_{1,j}^\top(t), \dot{\xi}_{2,j}^\top(t), \dot{\upsilon}_{1,j}^\top(t), \dot{\upsilon}_{2,j}^\top(t)]^\top.$$

Assuming small changes in Euler angels during flight control, the derivative of the kinematics equation (13):

$$\begin{aligned} \ddot{\xi}_{1,j}(t) =& \underbrace{\dot{\mathbf{R}}_{ZYX,j}(\xi_{2,j}(t))}_{\approx \mathbf{0}} \upsilon_{1,j}(t) + \\ & \underbrace{\mathbf{R}_{ZYX,j}(\xi_{2,j}(t))}_{\approx \mathbf{I}} \dot{\upsilon}_{1,j}(t), \\ \ddot{\xi}_{2,j}(t) =& \underbrace{\dot{\mathbf{T}}_j(\xi_{2,j}(t))}_{\approx \mathbf{0}} \upsilon_{2,j}(t) + \underbrace{\mathbf{T}_j(\xi_{2,j}(t))}_{\approx \mathbf{I}} \dot{\upsilon}_{2,j}(t), \end{aligned}$$

shows that in a hovering state, $\ddot{\xi}_{1,j}(t) \approx \dot{\upsilon}_{1,j}(t)$ and $\ddot{\xi}_{2,j}(t) \approx \dot{\upsilon}_{2,j}(t)$ are valid. As a result, the state-space representation of one agent is [61]:

$$\dot{\mathbf{x}}_j = \begin{bmatrix} \mathbf{R}_{ZYX,j}(\xi_{2,j})\upsilon_{1,j} \\ \mathbf{T}_j(\xi_{2,j})\upsilon_{2,j} \\ \frac{1}{m_j}\{\mathbf{R}_{ZYX,3,j}(\xi_{2,j})T_{\mathrm{B},j} - m_j g\mathbf{e}_3 - \mathbf{D}_{\mathrm{f},j}\dot{\xi}_{1,j}\} \\ \mathbf{J}_j^{-1}(\xi_{2,j})\{\tau_{\mathrm{B},j} - \mathbf{C}_j(\xi_{1,j}, \xi_{2,j})\dot{\xi}_{2,j}\} \end{bmatrix}, \tag{14}$$

where m_j(kg) is the mass of the drone, $T_{\mathrm{B},j}(t)$(N) is the thrust force in z_c direction (upward), $g = 9.81$(m/s^2) is gravity constant, $\mathbf{e}_3 = [0, 0, 1]^\top$, $\mathbf{D}_{\mathrm{f},j} = \mathrm{diag}(D_x, D_y, D_z)$(kg/s) includes drag, aerodynamic effects, etc., and $\tau_{\mathrm{B},j}(t) = [\tau_{\phi,j}(t), \tau_{\theta,j}(t), \tau_{\psi,j}(t)]^\top$(Nm) is the input torque vector. Moreover, the details of $\mathbf{R}_{ZYX,j}(\xi_{2,j}(t))$, $\mathbf{T}_j(\xi_{2,j}(t))$, $\mathbf{C}_j(\xi_{1,j}(t), \xi_{2,j}(t))$, and $\mathbf{J}_j(\xi_{2,j}(t))$ can be found in Section III of Ref. [61], and $\mathbf{R}_{ZYX,3,j}(\xi_{2,j}(t))$ is the third column of the rotation matrix $\mathbf{R}_{ZYX,j}(\xi_{2,j}(t))$.

The quadrotor UAV is underactuated, and the control structure uses a cascade design, which controls the translation part of dynamics (14) in the first layer and the orientation in the second layer. The translation dynamics includes the states $\mathbf{x}_{\mathrm{t},j}(t) = [x_{c,j}(t), y_{c,j}(t), z_{c,j}(t), u_j(t), v_j(t), w_j(t)]^\top$, expressed in rows 1-3 and 7-9 of Eq. (14). The SDC parameterization considers full actuation for translation control:

$$\mathbf{A}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}(t)) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{R}_{ZYX,j}(\xi_{2,j}(t)) \\ \mathbf{0}_{3\times3} & -\frac{1}{m_j}\mathbf{D}_{\mathrm{f},j}\mathbf{R}_{ZYX,j}(\xi_{2,j}(t)) \end{bmatrix},$$

$$\mathbf{B}_{\mathrm{t},j} = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \frac{1}{m_j}\mathbf{I}_{3\times3} \end{bmatrix}.$$

The component $[\mathbf{R}_{ZYX,3,j}(\xi_{2,j})T_{\mathrm{B},j}]_{3\times1}$ in Eq. (14) is a vector of thrust, and is distributed in the translation dynamics by the third component of the rotation matrix. Besides the SDRE design, there is a cascade controller as well, Eq. (17), to address the underactuation condition of the quadrotor.

In order to implement the cascade design and SDRE control, in the first step, the SDRE controller must be operated with the assumption of full actuation condition, which demands a $\mathbf{B}_{\mathrm{t},j} = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \frac{1}{m_j}\mathbf{I}_{3\times3} \end{bmatrix}$ matrix of 6×3 dimension. If the third component of the rotation matrix enters the SDC parameterization, a scalar thrust term is obtained as the single input to the problem, which creates an obstacle in the design. This topic has been presented in previous implementations and exercised with different conditions, platforms, and controllers [61–63], but it was originally proposed as cascade control design; details can be seen in Ref. [64].

Assigning weighting matrices with proper dimensions and solving the SDRE for the translation part:

$$\begin{aligned}\mathbf{A}_{\mathrm{t},j}^{\top}(\mathbf{x}_{\mathrm{t},j})\mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}) + \mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j})\mathbf{A}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}) - \\ \mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j})\mathbf{B}_{\mathrm{t},j}\mathbf{R}_{\mathrm{t},j}^{-1}(\mathbf{x}_{\mathrm{t},j})\mathbf{B}_{\mathrm{t},j}^{\top}\mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}) + \\ \mathbf{Q}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}) = \mathbf{0},\end{aligned}$$

results in the sub-optimal gain $\mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}(t))$ for the control law:

$$\mathbf{u}_{\mathrm{t},j}(t) = -\mathbf{R}_{\mathrm{t},j}^{-1}(\mathbf{x}_{\mathrm{t},j}(t))\mathbf{B}_{\mathrm{t},j}^{\top}\mathbf{K}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}(t))\mathbf{e}_{\mathrm{t},j}(t), \quad (15)$$

where $\mathbf{e}_{\mathrm{t},j}(t) = \mathbf{x}_{\mathrm{t},j}(t) - \mathbf{x}_{\mathrm{t,des},j}(t)$ in which $\mathbf{x}_{\mathrm{t,des},j}(t)$ includes the desired translation variables. The three components of the control law, in (15), are substituted in the thrust equation (16) [61]:

$$T_{\mathrm{B},j}(t) = m_j\{\mathbf{R}_{ZYX,3,j}^{\top}(\xi_{2,j}(t))(\mathbf{u}_{\mathrm{t},j}(t) + g\mathbf{e}_3)\}. \quad (16)$$

Note that gravity is compensated for in (16), and for this reason, the SDC matrix $\mathbf{A}_{\mathrm{t},j}(\mathbf{x}_{\mathrm{t},j}(t))$ excluded the gravity term. The second cascade control layer regulates the orientation of the drone to help the translation part, based on two constraints [62]:

$$\begin{aligned}&\theta_{\mathrm{des},j}(t) = \\ &\tan^{-1}\frac{u_{\mathrm{t},1,j}(t)\cos(\psi_{\mathrm{des},j}) + u_{\mathrm{t},2,j}(t)\sin(\psi_{\mathrm{des},j})}{u_{\mathrm{t},3,j}(t) + g}, \\ &\phi_{\mathrm{des},j}(t) = \\ &\sin^{-1}\frac{u_{\mathrm{t},1,j}(t)\sin(\psi_{\mathrm{des},j}) - u_{\mathrm{t},2,j}(t)\cos(\psi_{\mathrm{des},j})}{\sqrt{u_{\mathrm{t},1,j}^2(t) + u_{\mathrm{t},2,j}^2(t) + (u_{\mathrm{t},3,j}(t) + g)^2}},\end{aligned} \quad (17)$$

where $\psi_{\mathrm{des},j}$(rad) is the desired constant yaw of the system and i.e. $u_{\mathrm{t},2,j}(t)$ is the second component of $\mathbf{u}_{\mathrm{t},j}(t)$. The state vector of orientation control is set as $\mathbf{x}_{\mathrm{o},j}(t) = [\phi_j(t), \theta_j(t), \psi_j(t), p_j(t), q_j(t), r_j(t)]^{\top}$, and the relevant dynamics for orientation, rows 4–6 and 10–12 of (14), provides the SDC matrices:

$$\mathbf{A}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{T}_j(\xi_{2,j}) \\ \mathbf{0}_{3\times3} & -\mathbf{J}_j^{-1}(\xi_{2,j})\mathbf{C}_j(\xi_{1,j}, \xi_{2,j}) \end{bmatrix},$$

$$\mathbf{B}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) = \begin{bmatrix} \mathbf{0}_{3\times3} \\ \mathbf{J}_j^{-1}(\xi_{2,j}) \end{bmatrix}.$$

The SDRE for the orientation control is:

$$\begin{aligned}\mathbf{A}_{\mathrm{o},j}^{\top}(\mathbf{x}_{\mathrm{o},j})\mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) + \mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j})\mathbf{A}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) - \\ \mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j})\mathbf{B}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j})\mathbf{R}_{\mathrm{o},j}^{-1}(\mathbf{x}_{\mathrm{o},j})\mathbf{B}_{\mathrm{o},j}^{\top}(\mathbf{x}_{\mathrm{o},j}) \\ \mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) + \mathbf{Q}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}) = \mathbf{0},\end{aligned}$$

which generates the sub-optimal gain $\mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}(t))$ for the control law:

$$\mathbf{u}_{\mathrm{o},j}(t) = -\mathbf{R}_{\mathrm{o},j}^{-1}(\mathbf{x}_{\mathrm{o},j}(t))\mathbf{B}_{\mathrm{o},j}^{\top}(\mathbf{x}_{\mathrm{o},j}(t))\mathbf{K}_{\mathrm{o},j}(\mathbf{x}_{\mathrm{o},j}(t))\mathbf{e}_{\mathrm{o},j}(t),$$

where $\mathbf{e}_{\mathrm{o},j}(t) = \mathbf{x}_{\mathrm{o},j}(t) - \mathbf{x}_{\mathrm{o,des},j}(t)$ and $\mathbf{x}_{\mathrm{o,des},j}(t)$ includes the desired orientation variables. Note that $\phi_{\mathrm{des},j}(t)$ and $\theta_{\mathrm{des},j}(t)$ in $\mathbf{x}_{\mathrm{o,des},j}(t)$ are updated in each time-step of the simulation or experiment by (17).

The input thrust and torque vector to the system are generated by four rotating propellers:

$$\begin{bmatrix} T_{\mathrm{B},j}(t) \\ \tau_{\phi,j}(t) \\ \tau_{\theta,j}(t) \\ \tau_{\psi,j}(t) \end{bmatrix} = \mathbf{M}_{\mathrm{x},j}\omega_j^2(t), \quad (18)$$

where $\omega_j(t) \in \mathbb{R}^4$ is the angular velocity vector of the rotors and

$$\mathbf{M}_{\mathrm{x},j} = \begin{bmatrix} k & k & k & k \\ 0 & -lk & 0 & lk \\ -lk & 0 & lk & 0 \\ d & -d & d & -d \end{bmatrix},$$

is the mixer matrix of the quadrotor, which is defined based on the "plus" configuration of the rotors in Fig. 2, in which $k(\mathrm{Ns}^2/\mathrm{rad}^2)$ is the lift constant or thrust factor, $l(\mathrm{m})$ is the distance between motor and CoM of the quadrotor, and $d(\mathrm{Nms}^2/\mathrm{rad}^2)$ is the drag constant.

Remark 3 *To regulate the system (2) to the desired condition, the equilibrium point of the system should be zero. Many systems, such as robotic manipulators, multirotor UAVs, etc., work under the effect of gravity, where* $\mathbf{f}_j(\mathbf{0}) \neq \mathbf{0}$*. For those systems, the gravity compensation should be considered as shifting the equilibrium to zero [65]. Here in Section 3.1, the gravity will be compensated for through the cascade control design of the UAVs, which performs translation control in the first layer and orientation control in the second one [62].*

3.2. Wheeled Mobile Robot

Consider j-th wheeled mobile robot of a multi-agent system with differential wheels working on $\{X, Y\}$ planar Cartesian coordinates. The coordinates, and global and local axes, are shown in Fig. 3. The center of mass of the robot is defined by $\{x_{\mathrm{c},j}(t), y_{\mathrm{c},j}(t)\}$(m), and the rotation of the robot around Z axis, measured from X axis (counter-clockwise indicates positive direction), is called $\phi_j(t)$(rad). The system is subjected to two constraints showing that the robot cannot move alongside the axis of the wheels and that the wheels only have a pure rolling motion (rolling without slipping).

The first auxiliary relation is found by combining the constraints on the left and right wheels, $\dot{\phi}_j(t) = \frac{r}{2b}(\dot{\theta}_{\mathrm{r},j}(t) - \dot{\theta}_{\mathrm{l},j}(t))$. This integration generates a holonomic constraint [66]:

$$\phi_j(t) = \frac{r}{2b}(\theta_{\mathrm{r},j}(t) - \theta_{\mathrm{l},j}(t)). \quad (19)$$

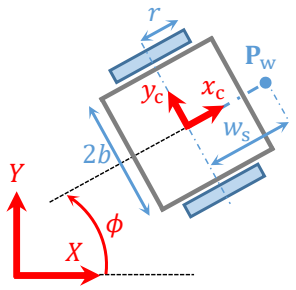

Figure 3. The schematic view and axis definition of a differential-wheel mobile robot.

The non-holonomic constraint, derived from the rolling condition of the wheels, is obtained [67]:

$$\dot{x}_{c,j}(t)\cos\phi_j(t)+\dot{y}_{c,j}(t)\sin\phi_j(t)=\frac{r}{2}(\dot{\theta}_{r,j}(t)+\dot{\theta}_{l,j}(t)). \tag{20}$$

Equation (19) is the holonomic constraint of the system and equations ($\dot{y}_{c,j}(t)\cos\phi_j(t) - \dot{x}_{c,j}(t)\sin\phi_j(t) = 0$) and (20) are non-holonomic constraints, which could be represented as $\mathbf{A}_j(\mathbf{q}_j(t))\dot{\mathbf{q}}_j(t)=\mathbf{0}$, where

$$\mathbf{A}_j(\mathbf{q}_j(t))=\begin{bmatrix}-\sin\phi_j(t) & \cos\phi_j(t) & 0 & 0\\ -\cos\phi_j(t) & -\sin\phi_j(t) & \frac{r}{2} & \frac{r}{2}\end{bmatrix}, \tag{21}$$

in which generalized coordinate vector of the j-th system is:

$$\mathbf{q}_j(t)=[x_{c,j}(t),y_{c,j}(t),\theta_{r,j}(t),\theta_{l,j}(t)]^\top. \tag{22}$$

The dynamics equation of the mobile robot is the common form of a second-order differential equation:

$$\begin{aligned}\mathbf{M}_j(\mathbf{q}_j(t))\ddot{\mathbf{q}}_j(t)+\mathbf{c}_j(\mathbf{q}_j(t),\dot{\mathbf{q}}_j(t)) =\\ \mathbf{E}\tau_j(t)-\mathbf{A}_j^\top(\mathbf{q}_j(t))\lambda_j(t),\end{aligned} \tag{23}$$

where $\mathbf{M}_j(\mathbf{q}_j(t)) : \mathbb{R}^4 \to \mathbb{R}^{4\times4}$ is the inertia matrix; the Coriolis-centrifugal vector is $\mathbf{c}_j(\mathbf{q}_j(t),\dot{\mathbf{q}}_j(t)) : \mathbb{R}^4 \times \mathbb{R}^4 \to \mathbb{R}^4$; $\mathbf{E} = [\mathbf{0}_{2\times2},\mathbf{I}_{2\times2}]^\top$, $\tau_j(t) = [\tau_{r,j}(t),\tau_{l,j}(t)]^\top$ is the input torque vector to the wheels; $\mathbf{A}_j(\mathbf{q}_j(t)) : \mathbb{R}^4 \to \mathbb{R}^{2\times4}$ is constraint matrix (21); and $\lambda_j(t) = [\lambda_{1,j}(t),\lambda_{2,j}(t)]^\top$ is the vector of the Lagrange multipliers. The details of the inertia matrix and Coriolis vector can be found in Ref. [67]. One way to omitting unknown Lagrange multipliers from Eq. (23) is to find the null space of $\mathbf{A}_j(\mathbf{q}_j(t))$:

$$\mathbf{S}_j(\mathbf{q}_j(t))=\begin{bmatrix}\frac{r}{2}\cos(\phi_j(t)) & \frac{r}{2}\cos(\phi_j(t))\\ \frac{r}{2}\sin(\phi_j(t)) & \frac{r}{2}\sin(\phi_j(t))\\ 1 & 0\\ 0 & 1\end{bmatrix},$$

and pre-multiply the transpose of $\mathbf{S}_j(\mathbf{q}_j(t))$ to (23) to obtain [66]:

$$\begin{aligned}\mathbf{S}_j^\top(\mathbf{q}_j(t))\mathbf{M}_j(\mathbf{q}_j(t))\ddot{\mathbf{q}}_j(t)+\\ \mathbf{S}_j^\top(\mathbf{q}_j(t))\mathbf{c}_j(\mathbf{q}_j(t),\dot{\mathbf{q}}_j(t))=\tau_j(t).\end{aligned} \tag{24}$$

The angular velocities of the wheels are arranged in a vector $\mathbf{v}_j(t) = [\dot{\theta}_{r,j}(t),\dot{\theta}_{l,j}(t)]^\top$; $\dot{\mathbf{q}}_j(t) = \mathbf{S}_j(\mathbf{q}_j(t))\mathbf{v}_j(t)$ is introduced; the second time derivative of the generalized coordinates results in:

$$\ddot{\mathbf{q}}_j(t)=\dot{\mathbf{S}}_j(\mathbf{q}_j(t))\mathbf{v}_j(t)+\mathbf{S}_j(\mathbf{q}_j(t))\dot{\mathbf{v}}_j(t). \tag{25}$$

The state-vector of the system is chosen as $\mathbf{x}_j = [\mathbf{q}_j^\top(t),\mathbf{v}_j^\top(t)]^\top$. Substitution of $\ddot{\mathbf{q}}_j(t)$ from (24) into (25) will result in a state-space representation of the system:

$$\begin{aligned}\dot{\mathbf{x}}_j=&\begin{bmatrix}\mathbf{S}_j\mathbf{v}_j\\ -(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}[\mathbf{S}_j^\top\mathbf{M}_j\dot{\mathbf{S}}_j\mathbf{v}_j+\mathbf{S}_j^\top\mathbf{c}_j]\end{bmatrix}+\\ &\begin{bmatrix}\mathbf{0}_{4\times2}\\ (\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}\end{bmatrix}\tau_j.\end{aligned} \tag{26}$$

In the control scheme of WMR, the controlled output of the system is a point $\mathbf{P}_w(t)$, ahead of the CoM of the mobile robot; w_s is the distance between the CoM of the robot; $\mathbf{P}_w(t)$ (see Fig. 3). The point is related to the state variables of the WMR through a kinematics relation. To control this point, the output- and state-dependent Riccati equation (OSDRE) approach is needed [57]. The output of the j-th WMR is defined as

$$\mathbf{y}_j(t)=\begin{bmatrix}x_{c,j}(t)+w_s\cos\phi_j(t)\\ y_{c,j}(t)+w_s\sin\phi_j(t)\end{bmatrix},$$

and the first two components of $\mathbf{S}_j(\mathbf{q}_j(t))\mathbf{v}_j(t)$ results in

$$\begin{aligned}\dot{x}_{c,j}(t)&=\frac{r}{2}(\dot{\theta}_{r,j}(t)+\dot{\theta}_{l,j}(t))\cos\phi_j(t),\\ \dot{y}_{c,j}(t)&=\frac{r}{2}(\dot{\theta}_{r,j}(t)+\dot{\theta}_{l,j}(t))\sin\phi_j(t).\end{aligned} \tag{27}$$

Substituting (27) into $\dot{\mathbf{y}}_j(t)$, and another time derivative $\ddot{\mathbf{y}}_j(t)$ results in:

$$\ddot{\mathbf{y}}_j(t)=\alpha_j(\mathbf{x}_j(t))\dot{\mathbf{v}}_j(t)+\beta_j(\mathbf{x}_j(t)), \tag{28}$$

where $\alpha_j(\mathbf{x}_j(t)) : \mathbb{R}^6 \to \mathbb{R}^{2\times2}$ and $\beta_j(\mathbf{x}_j(t)) : \mathbb{R}^6 \to \mathbb{R}^2$. $\alpha_j(\mathbf{x}_j(t))$ and $\beta_j(\mathbf{x}_j(t))$ are a nonlinear matrix and vector, respectively, that have been generated during the computation of the derivative of the output. The symbolic representation of them can be revisited in [67]. $\alpha_j(\mathbf{x}_j(t))$ and $\beta_j(\mathbf{x}_j(t))$ are kinematic transformations that change the dynamics of the mobile robot from a *state* to an *output* representation. Substituting $\dot{\mathbf{v}}_j(t)$ from the lower set of (26) into (28) provides:

$$\begin{aligned}\ddot{\mathbf{y}}_j=&\alpha_j\{-(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}[\mathbf{S}_j^\top\mathbf{M}_j\dot{\mathbf{S}}_j\mathbf{v}_j+\mathbf{S}_j^\top\mathbf{c}_j]\}+\\ &\beta_j+\alpha_j(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}\tau_j.\end{aligned} \tag{29}$$

By defining a new state-vector for the output dynamics (29), $\mathbf{z}_j(t) = [\mathbf{y}_j^\top(t),\dot{\mathbf{y}}_j^\top(t)]^\top$, one could find the state-space representation of the output dynamic:

$$\begin{aligned}\dot{\mathbf{z}}_j=&\begin{bmatrix}\dot{\mathbf{y}}_j\\ \alpha_j\{-(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}[\mathbf{S}_j^\top\mathbf{M}_j\dot{\mathbf{S}}_j\mathbf{v}_j+\mathbf{S}_j^\top\mathbf{c}_j]\}+\beta_j\end{bmatrix}\\ &+\begin{bmatrix}\mathbf{0}_{2\times2}\\ \alpha_j(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}\end{bmatrix}\tau_j.\end{aligned} \tag{30}$$

The output- and state-dependent coefficient parameterization of system (30) is expressed as [57]:

$$\mathbf{A}_j = \begin{bmatrix} \mathbf{0}_{2\times2} & \mathbf{I}_{2\times2} \\ \mathbf{0}_{2\times2} & [\alpha_j\{-(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1}[\mathbf{S}_j^\top\mathbf{M}_j\dot{\mathbf{S}}_j\mathbf{v}_j + \mathbf{S}_j^\top\mathbf{c}_j\} + \beta_j]\dot{\mathbf{y}}_j^\dagger \end{bmatrix},$$

$$\mathbf{B}_j = \begin{bmatrix} \mathbf{0}_{2\times2} \\ \alpha_j(\mathbf{S}_j^\top\mathbf{M}_j\mathbf{S}_j)^{-1} \end{bmatrix},$$

where $\dot{\mathbf{y}}_j(t)^\dagger$ is pseudo-inverse of the velocity of the output $\dot{\mathbf{y}}_j(t)$. Finally, the control law is set in the form of:

$$\mathbf{u}_j(t) = -\mathbf{R}_j^{-1}(\mathbf{x}_j(t))\mathbf{B}_j^\top(\mathbf{x}_j(t))\mathbf{K}_j(\mathbf{x}_j(t))[\mathbf{z}_j(t) - \mathbf{z}_{\mathrm{des},j}(t)].$$

The necessary conditions of the output, controllability, and observability of the OSDRE, and condition on $\dot{\mathbf{y}}_j(t)^\dagger$ to avoid a singularity issue, were reported in [57].

4. Simulations

4.1. Multi-agent systems of UAV

A multi-agent system of $N = 1,089$ quadrotor agents is considered for a simulation that represents a regulation case from an initial to a final condition in $t_\mathrm{f} = 20(\mathrm{s})$. The agents must form a square shape when they are launched and keep the shape during the regulation. As they approach the final condition, the square shape and arrangement of the agents get more precise, and the error in regulation also converges to zero. The pseudocode of the square pattern of agents for the simulation case of 1,089 drones. The following loop will define the local arrangement of the agents in a square shape during the regulation, presented in the pseudocode:

for $k = 1, ..., N_\mathrm{n}$,
  for $j = 1 + (k-1)N_\mathrm{n}, ..., N_\mathrm{n} + (k-1)N_\mathrm{n}$,
    $x_{\mathrm{c,f},j} = \dfrac{-(L + D_\mathrm{a})}{2} + jD_\mathrm{a} - (k-1)L$,
    $y_{\mathrm{c,f},j} = \dfrac{-(L + D_\mathrm{a})}{2} + kD_\mathrm{a}$,
    $z_{\mathrm{c,f},j} = 0$,
  end,
end,

where length of square side is denoted by $L = 30(\mathrm{m})$, $N_\mathrm{n} = 33$ is number of rows and columns, and $D_\mathrm{a} = \frac{L}{N_\mathrm{n}} = 0.9091(\mathrm{m})$.

The leader is the first agent of the system and guides the multi-agent system in point-to-point regulation from an arbitrary start to an endpoint. The 1,089 agents are initially scattered randomly inside a $10 \times 10(\mathrm{m})$ square at the height of $10(\mathrm{m})$. The initial orientation angle and velocities of the translation and orientation states of the agents are zero. The three components of the pseudocode $(x_{\mathrm{c,f},j}, y_{\mathrm{c,f},j}, z_{\mathrm{c,f},j})$ are the first set of vector $\mathbf{x}_{\mathrm{f},j} \in \mathbb{R}^{12}$ in (6) for the quadrotor multi-agent system.

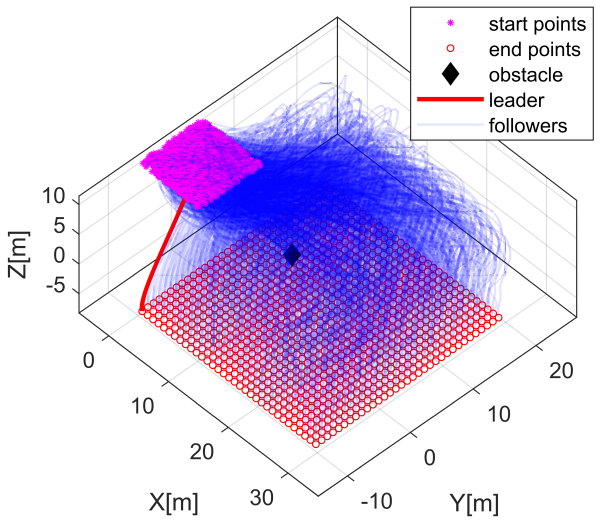

Figure 4. The trajectories of the multi-agent system of 1,089 quadrotor UAVs.

The position of the obstacle is $(x_\mathrm{o}, y_\mathrm{o}, z_\mathrm{o}) = (15, 0.25, 6)(\mathrm{m})$, with a safety radius of 0.5 for collision avoidance. The final position coordinates for the leader are:

$$\mathbf{x}_{\mathrm{f},1} = \begin{bmatrix} x_{\mathrm{c,des},1} \\ y_{\mathrm{c,des},1} \\ z_{\mathrm{c,des},1} \\ \mathbf{0}_{9\times1} \end{bmatrix} = \begin{bmatrix} 20 + x_{\mathrm{c,f},1} \\ y_{\mathrm{c,f},1} \\ z_{\mathrm{c,f},1} \\ \mathbf{0}_{9\times1} \end{bmatrix},$$

which is constant though the value for the rest of the agents is time-varying, presented in Eq. (6). This is a result of following the leader.

The physical characteristics of the agents are similar in the following way. The distance between the motor and the CoM of the quadrotor is $l = 0.225(\mathrm{m})$; the radius of the propeller is $R = 0.075(\mathrm{m})$; the lift constant is $k = 2.98 \times 10^{-6}(\mathrm{Ns^2rad^{-2}})$; the drag constant is $d = 1.14 \times 10^{-7}(\mathrm{Nms^2rad^{-2}})$; additionally $I_{\mathrm{xx}} = 4.856 \times 10^{-3}(\mathrm{kgm^2})$, $I_{\mathrm{yy}} = 4.856 \times 10^{-3}(\mathrm{kgm^2})$, and $I_{\mathrm{zz}} = 8.801 \times 10^{-3}(\mathrm{kgm^2})$ are moments of inertia around x, y, z axes. The mass of the drone is $m = 0.468(\mathrm{kg})$, the aerodynamics matrix is $\mathbf{D}_\mathrm{f} = \mathrm{diag}(0.25, 0.25, 0.25)(\mathrm{kg/s})$, the minimum and maximum rotor velocities are also $\omega_{\mathrm{min,max}} = 496.48, 744.73(\mathrm{rad/s})$.

The motion of the multi-agent system is represented in Fig. 4. Because the agents are very close to each other at the beginning, and the collision avoidance scheme in tuning increases the $\mathbf{Q}$ matrix drastically, we see a rapid divergence at the commencement of the simulation. The beginning of the regulation is like an "explosion" of multiple agents escaping from collisions with each other. The Cartesian-coordinate errors of the agents in 3D positioning are illustrated in Fig. 5 for all agents, followed by a zoomed view that shows the leader first converging towards zero, then the followers catching up with the leader.

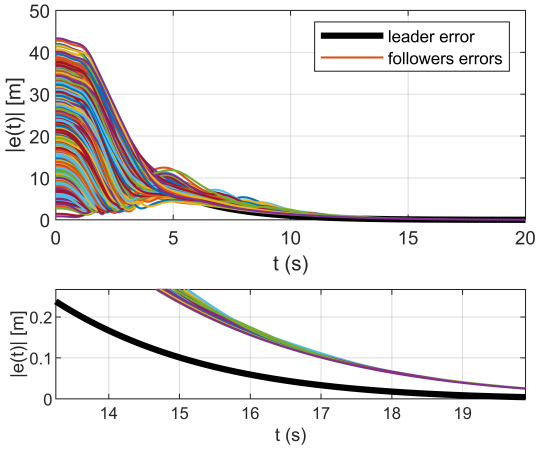

Figure 5. The error convergence of the multi-agent system of 1,089 quadrotor UAVs.

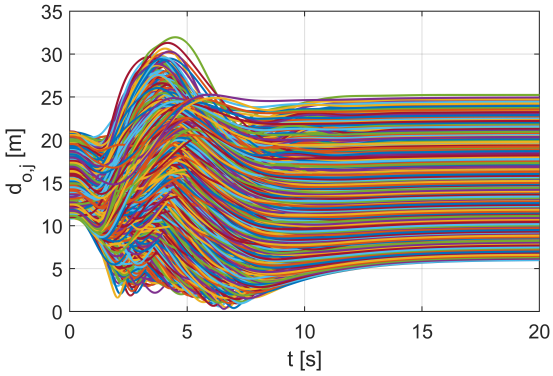

Figure 6. The distance between the drones and the obstacle for the multi-agent system of 1,089 quadrotor UAVs.

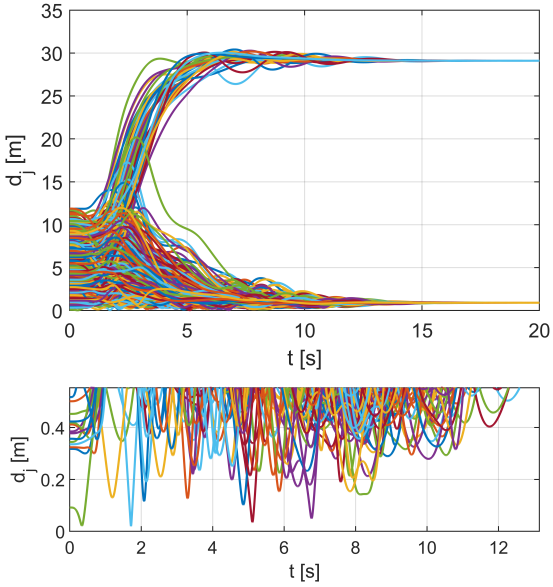

Figure 7. The collision distance between the drones for the multi-agent system of 1,089 quadrotor UAVs.

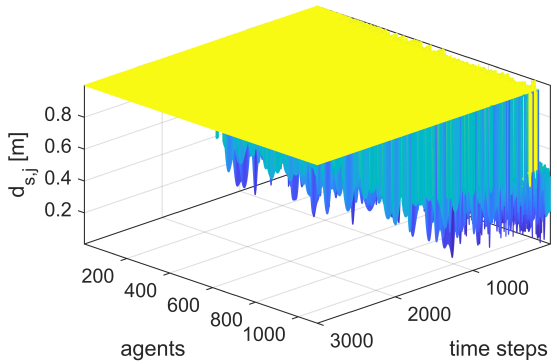

Figure 8. The $d_{s,j}(t)$ parameter for the multi-agent system of 1,089 quadrotor UAVs. Samples are shown the axes instead of time as a result of the mesh presentation.

The weighting matrices of the leader are selected as:

$$\mathbf{R}_{t,1} = 10 \times \mathbf{I}_{3\times3},\ \ \mathbf{R}_{o,1} = \mathbf{I}_{3\times3},$$
$$\mathbf{Q}_{t,1}(\mathbf{x}) = 0.5 \times \mathrm{diag}\left(1 + \frac{1}{d_{o,1}(\mathbf{x})}, 1 + \frac{1}{d_{o,1}(\mathbf{x})}, 1, 2, 2, 5\right),$$
$$\mathbf{Q}_{o,1} = \mathrm{diag}(\mathbf{2}_{1\times3}, \mathbf{0.5}_{1\times3})_{6\times6}.$$

The weighting matrices of the followers are:

$$\mathbf{R}_{t,j} = \mathbf{I}_{3\times3},\ \ \mathbf{R}_{o,j} = \mathbf{I}_{3\times3},$$
$$\mathbf{Q}_{t,j}(\mathbf{x}) = \mathrm{diag}\,(1 + Q_D(\mathbf{x}), 1 + Q_D(\mathbf{x}), 1, 2, 2, 5)$$
$$\mathbf{Q}_{o,j} = \mathrm{diag}(\mathbf{2}_{1\times3}, \mathbf{0.5}_{1\times3})_{6\times6},$$

where $Q_D(\mathbf{x}(t)) = \frac{1}{d_j(\mathbf{x}(t))} + \frac{1}{d_{o,j}(\mathbf{x}(t))} + \frac{1}{d_{s,j}(\mathbf{x}(t))}$, with $d_{s,j}(\mathbf{x}(t))$ set by (9).

Obstacle avoidance by embedding an artificial potential field into the SDRE has been reported on and analyzed in the literature. It has been stated that this method does not guarantee absolute obstacle avoidance, though it reduces the chance of impact significantly [52]. Figure 6 shows that the minimum distance between some agents and the obstacle was 30(cm). The collision avoidance between the agents is reported in Fig. 7. A group of drones in Fig. 7 merged to almost a 30(cm) distance; those are the ones at the beginning of rows, with the previous agents located at the end of previous rows. For the rest of the agents, the collision avoidance term, $d_j(t)$, shows a collision when it is lower than 0.48(m). The reason for the collision between the agents is the accumulation of 1,089 drones in a 10×10(m) at the initial condition of the simulation. After the commencement of the simulation, the collision avoidance scheme guided the drones and increased the distance between them after 8 seconds.

Checking the collision between two consecutive agents does not necessarily show all the collisions. Other agents might also have collisions that are demonstrated in Fig. 8.

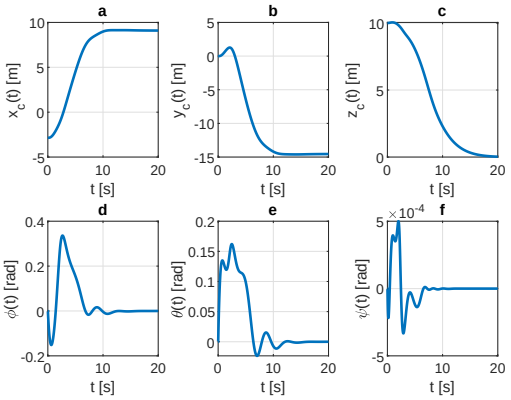

Figure 9. The position and orientation states for the 5th quadrotor UAV.

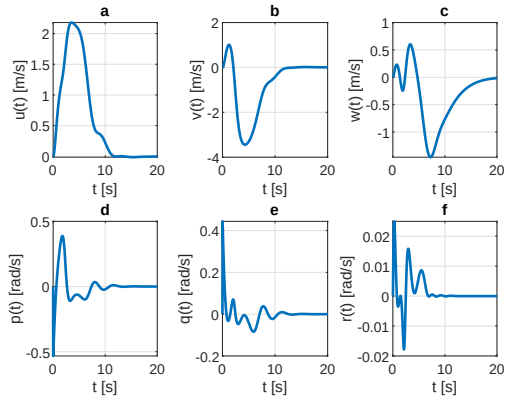

Figure 10. The linear and angular velocity states for the 5th quadrotor UAV of the multi-agent system.

Contrary to $d_j(t)$, the parameter $d_{s,j}(t)$ checks the collision of j-th agent with all previous agents. The behavior of $d_{s,j}(t)$ is similar to $d_j(t)$; at the beginning, the number of collisions is higher, and moving towards the final pattern, the distance between the agents gets bigger, starting from first time-step and ending 30,000-th time-steps, discretized into 20 seconds of total simulation time (samples are shown in the axes instead of time as a result of the mesh presentation). The collision of the agents is obvious since their arrangement is quite ambitious, leaving only 42(cm) between two agents in the final square shape. It can be observed in Fig. 7 when the distance for some drones is less than 0.48(m).

The simulation of 1,089 agents, presented in Fig. 4 might not reveal the complexity of the multi-agent system dynamics, therefore, the data of the simulation for the 5th agent is presented. The position and velocity states of the 5th drone are presented in Figs. 9 and 10, respectively. The equation of the mixer matrix (18) presents the input angular velocities of the rotors, limited to the lower and upper bounds, Fig. 11.

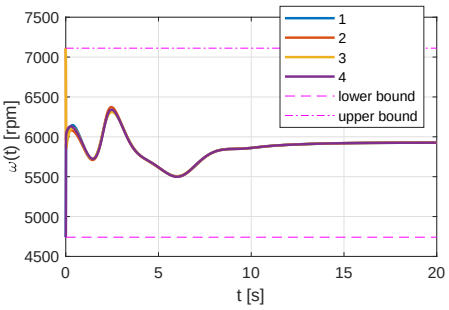

Figure 11. The angular velocities of the rotors for the 5th quadrotor UAV.

4.2. WMR

A group $N = 45$, agents was chosen for a regulation simulation over $t_f = 15(s)$ with circular final arrangements of the agents. The number of agents was selected smaller in comparison with Section 4.1 to highlight the role of obstacle avoidance in the results. The following algorithm defines the arrangement of the agents in a circular pattern:

for $j = 1, ..., N$,
$r_p = 1$,
$c_o = r_p$,
$x_{c,f,j} = x_{shift} + r_p \cos\frac{(j-1)2\pi}{N_c}$,
$y_{c,f,j} = y_{shift} + r_p \sin\frac{(j-1)2\pi}{N_c}$,
for $k = 1, ..., N_r$,
if $j > \frac{k(k+1)N_c}{2}$
$c_o = c_o + r_p$,
$x_{c,f,j} = x_{shift} + r_p \cos\frac{(j-N_c-1)2\pi r_p}{N_c c_o}$,
$y_{c,f,j} = y_{shift} + r_p \sin\frac{(j-N_c-1)2\pi r_p}{N_c c_o}$,
end,
end,
end,

where $(x_{shift}, y_{shift}) = (40, 20)(m)$ defines the center of the circular pattern; $N_c = 3$ is the number of agents in the first circle; $N_r = 5$ is number of rings; $N = \frac{N_r(N_r+1)N_c}{2} = 45$ is total number of agents; and $r_p = 1(m)$ is the distance between the rings.

The position of the obstacle is set as $(x_o, y_o) = (25, 10)(m)$ with a safety radius of $x_{min} = 0.5(m)$. The agents are randomly scattered in a $10 \times 10(m)$ square at the origin. These random scattered agents are almost 40(m) away from the desired position of the pattern.

Table 2. The error of the system under different w_s values.

w_s(mm)	error leader (mm)	error 5th follower (mm)
20	144.3912	911.3158
50	76.4762	345.8130
80	104.3379	388.0787
100	124.3148	427.7242
200	224.7471	630.6206
300	324.6205	827.9986
400	424.4897	1025.9861
500	524.4070	1223.8401

The physical characteristics of the mobile robots were chosen as follows: the radius of the wheels is denoted by $r = 0.08(\text{m})$; the length of the wheels' axis by $b = 0.145(\text{m})$; mass of base is $m_\text{b} = 6(\text{kg})$; mass of wheels is $m_\text{w} = 0.32(\text{kg})$; moment of inertia components are $I_{\text{zz,b}} = 0.06363, I_{\text{zz,w}} = 0.00006, I_{\text{yy,w}} = 0.00081(\text{kgm}^2)$; and the distance of the look-ahead control term is selected as $w_\text{s} = 0.1(\text{m})$.

The weighting matrices of the leader are set:

$$\mathbf{R}_1 = \mathbf{I}_{2\times 2},$$

$$\mathbf{Q}_1(\mathbf{x}(t)) = \text{diag}\left(0.5, 0.5 + \frac{1}{\sqrt{(d_{\text{o},1}(\mathbf{x}(t)) - 2)^2}}, 2, 2\right).$$

The weighting matrices of the followers are selected as:

$$\mathbf{R}_j = \mathbf{I}_{2\times 2}, \quad \mathbf{Q}_j(\mathbf{x}(t)) = \text{diag}\,(0.5, 0.5 + Q_\text{D}(\mathbf{x}(t)), 2, 2),$$

where $Q_\text{D}(\mathbf{x}(t)) = \frac{1}{d_j(\mathbf{x}(t))} + \frac{1}{d_{\text{s},j}(\mathbf{x}(t))} + \frac{1}{\sqrt{(d_{\text{o},1}(\mathbf{x}(t))-2)^2}}$,

in which $d_j = \sqrt{(x_{\text{c},j} - x_{\text{c},j-1})^2 + (y_{\text{c},j} - y_{\text{c},j-1})^2}$.

The trajectories of the leader and follower agents are illustrated in Fig. 12. The leader and the followers avoided the obstacle, which are represented in the trajectories. The error convergence of the multi-agent system is presented in Fig. 13. The obstacle and collision avoidance terms are plotted in Figs. 14 and 15, respectively. The input signals and state variables of one agent are presented in Figs. 16 and 17, respectively. The error of the leader and 5th follower is found at 122.6741(mm) and 411.7172(mm) respectively. It is important to notice that the error is a function of the look-ahead control parameter which was set $w_\text{s} = 0.1(\text{m})$. This means that the robot is regulating the look-ahead point towards the desired condition, not the CoM of the robot. If w_s is reduced, the error changes. The overshoot and the behavior of the regulation also change as the look-ahead parameter varies. A comparative table is presented to clarify this effect in Table 2. Another simulation was done for $1,050$ WMRs to show the capacity of the method for a large-population multi-agent system. For the sake of brevity, only the migration of the systems from initial to final condition, following the leader and the pattern, is illustrated in Fig. 18.

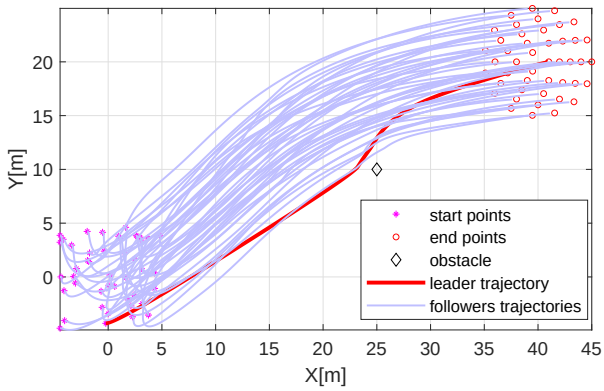

Figure 12. The trajectories of the leader and 45 follower agents.

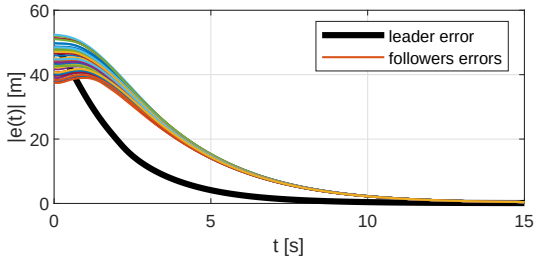

Figure 13. The errors of the leader and 45 follower agents.

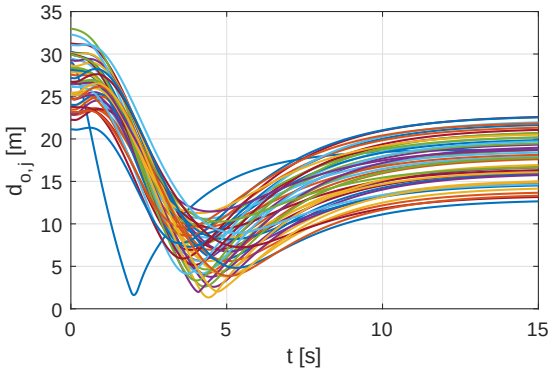

Figure 14. The obstacle avoidance performance of the leader/follower system of mobile robots.

4.3. Tuning of Weighting Matrices

There are some general rules for the selection of weighting matrices of the SDRE. **Q** penalizes the states and **R** penalizes the inputs. Since the inverse of **R** is the first term of the control law, a decrease in **R** increases the input signal, which results in more energy consumption. Increase in **Q** also increases the precision of the state regulation. In this work, the weighting matrix of states has obstacle/collision avoidance terms for multi-agent control design. The definition of the weighting matrices for the leader and followers must be different. If the leader is too fast, the followers might not follow the leader with proper accuracy.

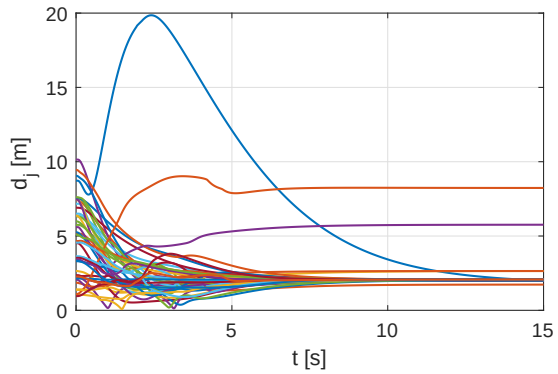

Figure 15. The collision avoidance performance of the leader/follower system of mobile robots.

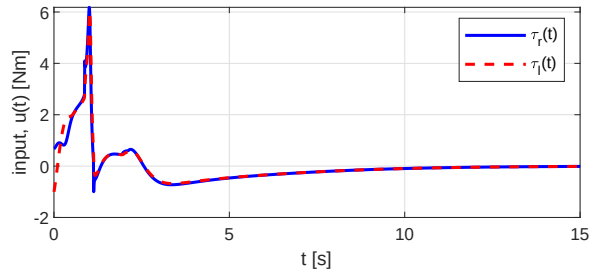

Figure 16. The input signals of the wheels for one of the agents.

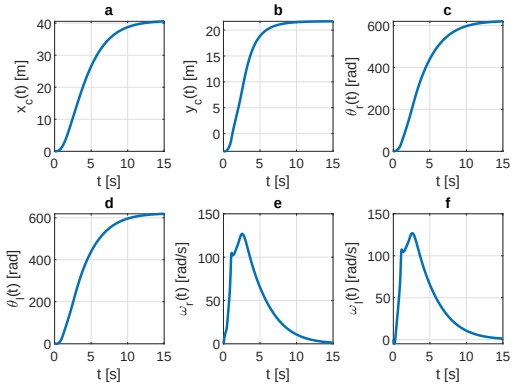

Figure 17. The state information for one of the agents.

Hence, the diagonal components of **Q** relative to the position coordinates of the leader are smaller than the ones for the followers for both simulations of the UAVs and WMRs. Another important point for tuning the robots is the contrast between the weighting matrices' value and the distance terms! If large values are selected for **Q**, the changes due to distance terms vanish, and collision/obstacle avoidance terms are not effective. These points were considered in the tuning of the matrices in the simulation sections. Considering the aforementioned point, it must be said that several trials and errors are needed to tune the controller.

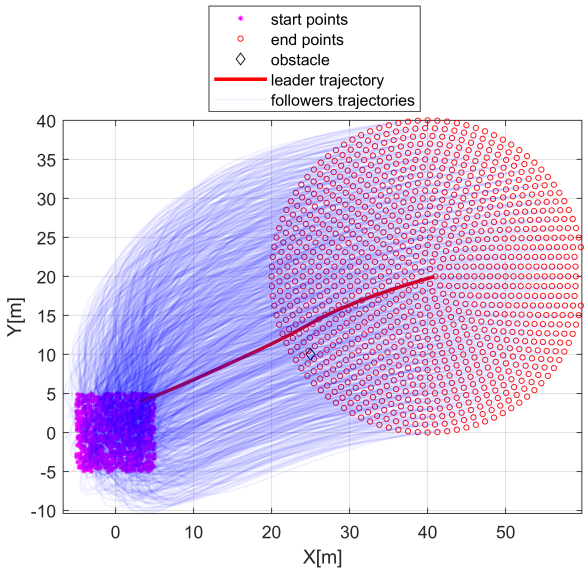

Figure 18. The migration of 1,050 mobile robots in leader-follower formation.

5. Conclusion

This work presents an SDRE control design (without augmentation of other techniques) for the formation regulation of a multi-agent system of robotic systems that takes into account the complex dynamics of the agents. The command to the multi-agent system is given to the leader, and the followers pursue the leader in accordance with the predefined pattern of the multi-agent system. Obstacle and collision avoidance among the agents was designed through the modification of the weighting matrices of states and the artificial potential field method. The application of the proposed method was devoted to the control of unmanned multi-copter systems and differential wheeled mobile robots. A highly populated multi-agent system of 1,089 agents was simulated for the UAV, and another simulation was presented for the WMRs with 45, and then 1,050 agents which were successfully regulated in the control task, preserving the expected formation shapes.

Code or data availability

The MATLAB codes of the simulations of this paper will be publicly available on the MATLAB Central, File Exchange page of the corresponding author (link).

AUTHOR

Saeed Rafee Nekoo* – Universidad de Sevilla, Spain , e-mail: saerafee@yahoo.com, ORCID: 0003-1396-5082.

*Corresponding author

References

[1] G. Beni, "The concept of cellular robotic system". *Proceedings 1988 IEEE International Symposium on Intelligent Control*, 1988, pp. 57–58.

[2] V. Genovese, P. Dario, R. Magni, and L. Odetti, "Self organizing behavior and swarm intelligence in a pack of mobile miniature robots in search of pollutants". *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 1992, pp. 1575–1582.

[3] N. Mitsumoto, T. Fukuda, K. Shimojima, and A. Ogawa, "Micro autonomous robotic system and biologically inspired immune swarm strategy as a multi agent robotic system". *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 2, 1995, pp. 2187–2192.

[4] T. Ueyama, T. Fukuda, and F. Arai, "Structure configuration using genetic algorithm for cellular robotic system". *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 1992, pp. 1542–1549.

[5] G. S. Chirikjian, "Kinematics of a metamorphic robotic system". *Proceedings of the 1994 IEEE international conference on robotics and automation*, 1994, pp. 449–455.

[6] G. Beni and P. Liang, "Pattern reconfiguration in swarms-convergence of a distributed asynchronous and bounded iterative algorithm", *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, 1996, pp. 485–490.

[7] C. R. Kube and E. Bonabeau, "Cooperative transport by ants and robots", *Robotics and Autonomous Systems*, vol. 30, no. 1-2, 2000, pp. 85–101.

[8] M. J. Krieger, J.-B. Billeter, and L. Keller, "Ant-like task allocation and recruitment in cooperative robots", *Nature*, vol. 406, no. 6799, 2000, pp. 992–995.

[9] L. Yang, J. Yu, S. Yang, B. Wang, B. J. Nelson, and L. Zhang, "A survey on swarm microrobotics", *IEEE Transactions on Robotics*, 2021.

[10] L. Yang and L. Zhang, "Motion control in magnetic microrobotics: From individual and multiple robots to swarms", *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, 2021, pp. 509–534.

[11] Y. Fu, H. Yu, X. Zhang, P. Malgaretti, V. Kishore, and W. Wang, "Microscopic swarms: From active matter physics to biomedical and environmental applications", *Micromachines*, vol. 13, no. 2, 2022, 295.

[12] A. R. Cheraghi, S. Shahzad, and K. Graffi, "Past, present, and future of swarm robotics". *Proceedings of SAI Intelligent Systems Conference*, 2021, pp. 190–233.

[13] G. A. Cardona, J. Ramirez-Rugeles, E. Mojica-Nava, and J. M. Calderon, "Visual victim detection and quadrotor-swarm coordination control in search and rescue environment", *International Journal of Electrical and Computer Engineering*, vol. 11, no. 3, 2021, 2079.

[14] G. Kumar, A. Anwar, A. Dikshit, A. Poddar, U. Soni, and W. K. Song, "Obstacle avoidance for a swarm of unmanned aerial vehicles operating on particle swarm optimization: A swarm intelligence approach for search and rescue missions", *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 44, no. 2, 2022, pp. 1–18.

[15] R. Krzysiak and S. Butail, "Information-based control of robots in search-and-rescue missions with human prior knowledge", *IEEE Transactions on Human-Machine Systems*, vol. 52, no. 1, 2021, pp. 52–63.

[16] H. Suresha, D. Sumithra, J. Renuka, B. Deepika, and N. Meghana, "Application of swarm robotics systems to marine environmental monitoring", *International Conference on Data Science, Machine Learning, and Applications*, 202 2, pp. 1075–1083.

[17] C. Carbone, D. Albani, F. Magistri, D. Ognibene, C. Stachniss, G. Kootstra, D. Nardi, and V. Trianni, "Monitoring and mapping of crop fields with UAV swarms based on information gain". *International Symposium Distributed Autonomous Robotic Systems*, 2021, pp. 306–319.

[18] Y. Liu and K. M. Passino, "Swarm intelligence: Literature overview", *Department of Electrical Engineering, The Ohio State University*, 2000.

[19] M. H. Korayem and S. R. Nekoo, "Controller design of cooperative manipulators using state-dependent Riccati equation", *Robotica*, vol. 36, no. 4, 2018, pp. 484–515.

[20] L. Yan, T. Stouraitis, and S. Vijayakumar, "Decentralized ability-aware adaptive control for multi-robot collaborative manipulation", *IEEE Robotics and Automation Letters*, vol. 6, no. 2, 2021, pp. 2311–2318.

[21] J. Hu, P. Bhowmick, I. Jang, F. Arvin, and A. Lanzon, "A decentralized cluster formation containment framework for multirobot systems", *IEEE Transactions on Robotics*, vol. 37, no. 6, 2021, pp. 1936–1955.

[22] B. Geranmehr and E. Khanmirza, "Optimal consensus control of multi-agent systems based on the state-dependent Riccati equation", *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 2022, 09544062221091526.

[23] Z. A. Ali and H. Zhangang, "Multi-unmanned aerial vehicle swarm formation control using hybrid strategy", *Transactions of the Institute of Measurement and Control*, vol. 43, no. 12, 2021, pp. 2689–2701.

[24] S. P. Hou and C. C. Cheah, "Multiplicative potential energy function for swarm control". *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 4363–4368.

[25] J. Cheng, W. Cheng, and R. Nagpal, "Robust and self-repairing formation control for swarms of mobile agents". *AAAI*, vol. 5, 2005.

[26] S. P. Hou, C.-C. Cheah, and J.-J. E. Slotine, "Dynamic region following formation control for a swarm of robots". *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 1929–1934.

[27] J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm". *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.

[28] E. Teruel, R. Aragues, and G. López-Nicolás, "A distributed robot swarm control for dynamic region coverage", *Robotics and Autonomous Systems*, vol. 119, 2019, pp. 51–63.

[29] G. Raja, K. Kottursamy, A. Theetharappan, K. Cengiz, A. Ganapathisubramaniyan, R. Kharel, and K. Yu, "Dynamic polygon generation for flexible pattern formation in large-scale UAV swarm networks". *2020 IEEE Globecom Workshops (GC Wkshps*, 2020, pp. 1–6.

[30] S. S. Ge, C.-H. Fua, and W.-M. Liew, "Swarm formations using the general formation potential function". *IEEE Conference on Robotics, Automation and Mechatronics, 2004.*, vol. 2, 2004, pp. 655–660.

[31] D. Grossman, I. Aranson, and E. B. Jacob, "Emergence of agent swarm migration and vortex formation through inelastic collisions", *New Journal of Physics*, vol. 10, no. 2, 2008, 023036.

[32] Z. Yang, Q. Zhang, and Z. Chen, "Formation control of multi-agent systems with region constraint", *Complexity*, vol. 2019, 2019.

[33] J. Wang and M. Xin, "Integrated optimal formation control of multiple unmanned aerial vehicles", *IEEE Transactions on Control Systems Technology*, vol. 21, no. 5, 2012, pp. 1731–1744.

[34] S. A. Kumar, J. Vanualailai, B. Sharma, and A. Prasad, "Velocity controllers for a swarm of unmanned aerial vehicles", *Journal of Industrial Information Integration*, vol. 22, 2021, 100198.

[35] B. S. Park and S. J. Yoo, "An error transformation approach for connectivity-preserving and collision-avoiding formation tracking of networked uncertain underactuated surface vessels", *IEEE Transactions on Cybernetics*, vol. 49, no. 8, 2018, pp. 2955–2966.

[36] B. S. Park and S. J. Yoo, "Connectivity-maintaining and collision-avoiding performance function approach for robust leader–follower formation control of multiple uncertain underactuated surface vessels", *Automatica*, vol. 127, 2021, 109501.

[37] R. Babazadeh and R. Selmic, "Distance-based multiagent formation control with energy constraints using SDRE", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, 2019, pp. 41–56.

[38] M. Davoodi and J. M. Velni, "Heterogeneity-aware graph partitioning for distributed deployment of multiagent systems", *IEEE Transactions on Cybernetics*, vol. 52, no. 4, 2022, pp. 2578–2588.

[39] S. R. Nekoo, J. Yao, A. Suarez, R. Tapia, and A. Ollero, "Leader-follower formation control of a large-scale swarm of satellite system using the state-dependent Riccati equation: Orbit-to-orbit and in-same-orbit regulation". *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10700–10707.

[40] H. Rouzegar, A. Khosravi, and P. Sarhadi, "Spacecraft formation flying control under orbital perturbations by state-dependent Riccati equation method in the presence of on–off actuators", *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 233, no. 8, 2019, pp. 2853–2867.

[41] R. Babazadeh and R. Selmic, "Anoptimal displacement-based leader-follower formation control for multi-agent systems with energy consumption constraints". *2018 26th Mediterranean Conference on Control and Automation (MED)*, 2018, pp. 179–184.

[42] M. Tannous, G. Franzini, and M. Innocenti, "State-dependent Riccati equation control for spacecraft formation flying in the circular restricted three-body problem". *2018 Astrodynamics Conference; American Astronautical Society: Springfield, VA, USA*, 2018.

[43] G. Rinaldo, E. Rafikova, and M. Rafikov, "Control of multiple mobile robots in dynamic formations". *International Symposium on Dynamic Problems of Mechanics*, 2017, pp. 359–369.

[44] S. Kumbasar and O. Tekinalp, "Fuzzy logic guidance of formation flight". *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2015, pp. 167–175.

[45] L. Felicetti and G. B. Palmerini, "Attitude coordination strategies in satellite constellations and formation flying". *2015 IEEE Aerospace Conference*, 2015, pp. 1–13.

[46] J.-I. Park, H.-E. Park, S.-Y. Park, and K.-H. Choi, "Hardware-in-the-loop simulations of gps-based navigation and control for satellite formation flying", *Advances in Space Research*, vol. 46, no. 11, 2010, pp. 1451–1465.

[47] D. Stansbery and J. Cloutier, "Nonlinear control of satellite formation flight". *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2000, 4436.

[48] A. A. Kabanov and V. A. Kramar, "Cooperative control of underwater manipulators based on

the SDRE method". *2021 International Russian Automation Conference (RusAutoCon)*, 2021, pp. 515–520.

[49] N. Nasiri, A. Fakharian, and M. B. Menhaj, "An uncertain optimal factorization of cooperative manipulators for robust optimal control schemes". *2022 30th International Conference on Electrical Engineering (ICEE)*, 2022, pp. 582–586.

[50] X. Lin, X. Shi, and S. Li, "Optimal cooperative control for formation flying spacecraft with collision avoidance", *Science Progress*, vol. 103, no. 1, 2020, 0036850419884432.

[51] R. Babaie and A. F. Ehyaie, "Robust optimal motion planning approach to cooperative grasping and transporting using multiple UAVs based on SDRE", *Transactions of the Institute of Measurement and Control*, vol. 39, no. 9, 2017, pp. 1391–1408.

[52] M. H. Korayem and S. R. Nekoo, "The SDRE control of mobile base cooperative manipulators: Collision free path planning and moving obstacle avoidance", *Robotics and Autonomous Systems*, vol. 86, 2016, pp. 86–105.

[53] D. Li, S. S. Ge, W. He, G. Ma, and L. Xie, "Multilayer formation control of multi-agent systems", *Automatica*, vol. 109, 2019, 108558.

[54] J. Wu, Q. Deng, T. Han, and H.-C. Yan, "Bipartite output regulation for singular heterogeneous multi-agent systems on signed graph", *Asian Journal of Control*, vol. 24, no. 5, 2022, pp. 2452–2460.

[55] Z. Pan, Z. Sun, H. Deng, and D. Li, "A multilayer graph for multiagent formation and trajectory tracking control based on MPC algorithm", *IEEE Transactions on Cybernetics*, vol. 52, no. 12, 2021, pp. 13586–13597.

[56] T. Çimen, "State-dependent Riccati equation (SDRE) control: A survey", *IFAC Proceedings Volumes*, vol. 41, no. 2, 2008, pp. 3761–3775.

[57] S. R. Nekoo, "Output-and state-dependent Riccati equation: An output feedback controller design", *Aerospace Science and Technology*, vol. 126, 2022, 107649.

[58] S. S. Mohammadi and H. Khaloozadeh, "Optimal motion planning of unmanned ground vehicle using SDRE controller in the presence of obstacles". *2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, 2016, pp. 167–171.

[59] M. Asgari and H. N. Foghahayee, "State dependent Riccati equation (SDRE) controller design for moving obstacle avoidance in mobile robot", *SN Applied Sciences*, vol. 2, no. 11, 2020, pp. 1–29.

[60] S. R. Nekoo and B. Geranmehr, "Nonlinear observer-based optimal control using the state-dependent Riccati equation for a class of non-affine control systems", *Journal of Control Engineering and Applied Informatics*, vol. 16, no. 2, 2014, pp. 5–13.

[61] S. Nekoo, J. Acosta, A. Gomez-Tamm, and A. Ollero, "Optimized thrust allocation of variable-pitch propellers quadrotor control: A comparative study on flip maneuver". *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*, 2019, pp. 86–95.

[62] S. R. Nekoo, J. Á. Acosta, and A. Ollero, "Quaternion-based state-dependent differential Riccati equation for quadrotor drones: Regulation control problem in aerobatic flight", *Robotica*, 2022, pp. 1–16.

[63] S. R. Nekoo, J. Á. Acosta, and A. Ollero, "Fully coupled six-dof nonlinear suboptimal control of a quadrotor: Application to variable-pitch rotor design". *Robot 2019: Fourth Iberian Robotics Conference: Advances in Robotics, Volume 2*, 2020, pp. 72–83.

[64] Z. Zuo, "Trajectory tracking control design with command-filtered compensation for a quadrotor", *IET Control Theory & Applications*, vol. 4, no. 11, 2010, pp. 2343–2355.

[65] S. R. Nekoo, J. Á. Acosta, and A. Ollero, "Gravity compensation and optimal control of actuated multibody system dynamics", *IET Control Theory & Applications*, vol. 16, no. 1, 2022, pp. 79–93.

[66] Y. Yamamoto and X. Yun, "Coordinating locomotion and manipulation of a mobile manipulator". *Proceedings of the 31st IEEE Conference on Decision and Control*, 1992, pp. 2643–2648.

[67] M. Korayem, S. Nekoo, and A. Korayem, "Finite time SDRE control design for mobile robots with differential wheels", *Journal of Mechanical Science and Technology*, vol. 30, no. 9, 2016, pp. 4353–4361.